UNIVERSITY OF SHEFFIELD

DOCTORAL THESIS

Structure-Preserving Matrix Methods for Computations on Univariate and Bivariate Bernstein Polynomials

---

Martin Bourne

A thesis submitted in partial fulfilment of the requirements for the degree of
Doctor of Philosophy

The University of Sheffield
Faculty of Engineering
Department of Computer Science

November 2017

Abstract

Curve and surface intersection finding is a fundamental problem in computer-aided geometric design (CAGD). This practical problem motivates the undertaken study into methods for computing the square-free factorisation of univariate and bivariate polynomials in Bernstein form. It will be shown how these two problems are intrinsically linked and how finding univariate polynomial roots and bivariate polynomial factors is equivalent to finding curve and surface intersection points.

The multiplicities of a polynomial's factors are maintained through the use of a square-free factorisation algorithm and this is analogous to the maintenance of smooth intersections between curves and surfaces, an important property in curve and surface design. Several aspects of the univariate and bivariate polynomial factorisation problem will be considered.

This thesis examines the structure of the greatest common divisor (GCD) problem within the context of the square-free factorisation problem. It is shown that an accurate approximation of the GCD can be computed from inexact polynomials even in the presence of significant levels of noise. Polynomial GCD computations are ill-posed, in that noise in the coefficients of two polynomials which have a common factor typically causes the polynomials to become coprime. Therefore, a method for determining the approximate greatest common divisor (AGCD) is developed, where the AGCD is defined to have the same degree as the GCD and its coefficients are sufficiently close to those of the exact GCD. The algorithms proposed assume no prior knowledge of the level of noise added to the exact polynomials, differentiating this method from others which require derived threshold values in the GCD computation.

The methods of polynomial factorisation devised in this thesis utilise the Sylvester matrix and a sequence of subresultant matrices for the GCD finding component. The classical definition of the Sylvester matrix is extended to compute the GCD of two and three bivariate polynomials defined in Bernstein form, and a new method of GCD computation is devised specifically for bivariate polynomials in Bernstein form which have been defined over a rectangular domain. These extensions are necessary for the computation of the factorisation of bivariate polynomials defined in the Bernstein form.

Supporting Publications

[9] BOURNE, M., WINKLER, J. R. and SU, Y. The computation of the degree of an approximate greatest common divisor of two Bernstein polynomials. *Applied Numerical Mathematics 111* (2017), 17-35.

[10] BOURNE, M., WINKLER, J. R. and SU, Y. A non-linear structure preserving matrix method for the computation of the coefficients of an approximate greatest common divisor of two Bernstein polynomials. *Journal of Computational and Applied Mathematics 320* (2017), 221-241.

Dedications

Firstly, and most importantly, I would like to thank my mother, for instilling in me the hardest of work ethics. To my wider family, thank you for your continued love and support. May you all bask in my reflected glory for many years to come.

Jessica, thank you.

I would also like to thank my supervisor, Dr Joab Winkler, for his patience, guidance and advice throughout my time as his student.

I would also like to thank the University of Sheffield and A*STAR Singapore for the A*STAR-Sheffield Research Attachment Programme which allowed me to conduct my research.

Contents

List of Tables

List of Figures

Symbols and Notation

Univariate Polynomials - Associated Values

$\dot{\sigma}_k$	The minimum singular value of the kth subresultant matrix S_k
$\sigma_{k,i}$	The ith singular value of the kth subresultant matrix S_k

Bivariate Polynomials - Associated Values

$\dot{\sigma}_{k_1,k_2}$	The minimum singular value of the (k_1, k_2)th subresultant matrix S_{k_1,k_2}
$\sigma_{k_1,k_2,i}$	The ith singular value of the (k_1, k_2)th subresultant matrix S_{k_1,k_2}

Polynomials

$\hat{f}(x)$	A polynomial with exact coefficients
$f(x)$	A polynomial with inexact coefficients
$\bar{f}(x)$	A polynomial whose coefficients have been normalised
$\tilde{f}_k(\omega)$	A polynomial which has been preprocessed and the independent variable x has been replaced by $\theta_k\omega$, where θ_k is the result of an optimisation problem and ω is the new independent variable
$\dot{f}(\omega)$	A polynomial whose coefficients have been perturbed and is given by $\tilde{f}(\omega) + z(\omega)$

Acronyms

AGCD	approximate greatest common divisor
BD	batch deconvolution
BDSTLN	batch deconvolution with STLN
BVDRGCD	bivariate dimension reducing GCD
BVGCD	bivariate GCD
CAGD	computer-aided geometric design
CAM	computer-aided manufacture
CBD	constrained batch deconvolution
CBDSTLN	constrained batch deconvolution with STLN
DC1	degree computation 1
DC2	degree computation 2
DC3	degree computation 3
DC4	degree computation 4
DC5	degree computation 5
GCD	greatest common divisor
LSE	least squares with equality
MUGCD	modified univariate GCD
NC	numerically controlled
PRS	polynomial remainder sequence
SD	separate deconvolution
SNTLN	structured non-linear total least norm
SQFF	square-free factorisation
STLN	structured total least norm
SVD	singular value decomposition
UGCD	univariate GCD
VDP	variation diminishing property

Chapter 1

Introduction

The calculation of the intersection points of curves and surfaces is one of the most fundamental problems in computer-aided geometric design (CAGD). The accurate representation of intersections is necessary in the construction of smooth, watertight surfaces. The computation of points of intersection can be reduced to computing (i) the roots of univariate polynomials and (ii) the factorisation of bivariate polynomials. This thesis develops a robust method for solving these problems where the polynomials are given in Bernstein form, since this basis is frequently used in CAGD. The method of root finding in this thesis preserves the multiplicity structure of the roots by first determining the polynomial's square-free factorisation and root multiplicity structure.

When considering intersections of Bézier curves and Bézier surface patches, the polynomials associated with these intersections are in Bernstein form. It is therefore necessary to derive robust methods native to polynomials in Bernstein form to avoid unstable conversion to the power basis [25, 26].

Smooth intersections are important in CAGD because they reduce high stresses at sharp corners, which can cause an object to fracture when in operation. Smooth or tangential intersections are associated with polynomials with roots of high multiplicity. It is this type of root finding problem where standard methods typically fail. Singular roots of high multiplicity can erroneously be computed as a cluster of roots, and this is equivalent to a loss of smoothness at a tangential intersection point. The standard root finding methods are more effectively applied by first determining the square-free factorisation of a polynomial, then computing the simple roots of its square-free factorisation.

The first part of this thesis (Chapters 3 and 4) considers the computation of the square-free factorisation and roots of a univariate polynomial in Bernstein form. Two key components in this process are (i) the computation of a sequence of iterative greatest common divisors (GCDs) and (ii) a structured polynomial deconvolution problem. Both of these problems are highly sensitive to noise in the input data, and structured methods developed in this work aim to overcome this limitation.

An approximate greatest common divisor (AGCD) of two inexact polynomials will be defined, and the degree of the AGCD will be computed from a modified version of the classical Sylvester matrix adapted for polynomials in Bernstein form. In this thesis a set of three preprocessing operations is developed and applied to the polynomials in order for the AGCD to be computed using the Sylvester matrix and the sequence of

subresultant matrices. It will be shown that the degree of the GCD is more reliably determined from the set of preprocessed subresultant matrices and that approximations of the cofactor polynomials and the AGCD can be several orders of magnitude more accurate than approximations obtained without preprocessing.

Structured perturbations are also added to the inexact polynomials, and a low rank approximation of the Sylvester matrix is determined. The coefficients of the cofactor polynomials and the AGCD are approximated, and these approximations of the exact polynomials are shown to be significantly better than those obtained by standard least squares based methods.

The second part of this thesis (Chapters 5 to 7) focuses on several new extensions to the work discussed in the first part. These extensions are necessary in the computation of the factorisation of a bivariate polynomial in Bernstein form. The factorisation of a bivariate polynomial reduces to the computation of a sequence of polynomial GCDs and a set of deconvolution problems.

In this thesis two new forms of the Sylvester matrix are defined. These are used in the computation of the GCD of three polynomials, also referred to as the three-polynomial GCD problem. Initial experiments show that polynomial ordering in these forms must be carefully considered. The definition of the Sylvester matrix in Bernstein form is generalised to accommodate bivariate polynomials defined over either a rectangular or triangular domain.

The method of computing the degree of the GCD of two or three bivariate polynomials over a triangular domain follows from the univariate problems. A relatively simple extension for the computation of the GCD of two or three bivariate polynomials defined over a rectangular domain is then described, but experiments show that this method is inefficient. A second, alternative and faster method which makes use of degree elevated polynomials will also be presented. This method gives similar results to the first extension, but with significantly reduced computational complexity.

Chapter 1 An overview of intersection finding algorithms is given. It will be shown how a curve or surface intersection problem reduces to the factorisation of a univariate or bivariate polynomial. For this problem a robust GCD finding algorithm is required and several GCD finding methods are discussed.

Chapter 2 In the second chapter, the Bézier curve and Bézier surface patch representations are defined. The univariate and bivariate Bernstein polynomial forms are also described and some basic polynomial arithmetic by structured matrix methods is considered. These methods are pertinent to the work discussed in later chapters.

Chapter 3 The third chapter considers the computation of the degree and coefficients of the GCD of two univariate polynomials $\hat{f}(x)$ and $\hat{g}(x)$ in Bernstein form. A sequence of subresultant matrices is used to determine the degree of the GCD and this is extended to the computation of the AGCD of two inexact polynomials $f(x)$ and $g(x)$.

It will be shown how preprocessing the subresultant matrices yields improved results in the computation of the degree, t, and the coefficients of the GCD. The

coefficients of the cofactor polynomials and the AGCD are approximated using a low rank approximation of the tth subresultant matrix, and the results of this new method, which will now be referred to as the univariate GCD (UGCD) method, will be compared with a standard least squares approach. It will be shown that, given the degree of the GCD is correctly determined, the low rank approximation method yields improved approximations of the coefficients of the cofactor polynomials and the GCD, particularly in the presence of noise.

Chapter 4 Having developed the univariate GCD (UGCD) method in the previous chapter, this chapter describes the square-free factorisation algorithm due to Gauss. The algorithm is used for the computation of multiple roots of a univariate polynomial in Bernstein form and has two key components. The first part is the computation of a sequence of polynomial GCDs and the second is a set of polynomial deconvolutions.

Modifications are made to the univariate GCD (UGCD) method, for use specifically in the set of GCD problems arising in the square-free factorisation algorithm. Each problem in the set of GCD problems $\{\hat{f}_{i+1}(x) = \text{GCD}(\hat{f}_i(x), \hat{g}_i(x))$ has the structure that $\hat{g}_i(x)$ is the derivative of $\hat{f}_i(x)$, and the Sylvester matrix and subresultant matrices can be structured accordingly. A lower limit for the degree of the GCD in the $(i+1)$th GCD computation can be determined from the ith GCD, so methods for a more efficient and more reliable algorithm are described. These modifications give a new GCD method called the modified univariate GCD (MUGCD) algorithm.

Polynomial division in the square-free factorisation algorithm is also discussed in this chapter. Several matrix based methods are considered for the deconvolution problem which arises specifically in the square-free factorisation algorithm. In particular, a new matrix based method which exploits the structure of this problem is described, and this gives significantly improved approximations of the set of polynomials when compared with naive deconvolution methods.

The combination of the modified univariate GCD (MUGCD) method and the batch deconvolution method gives a square-free factorisation (SQFF) algorithm which is then compared with other existing root finding methods. The new composite algorithm compares favourably with the alternative root finding methods, which fail to retain a polynomial's root multiplicity structure in the presence of noise.

Chapter 5 This chapter extends work in Chapter 3 and Chapter 4 to compute the GCD of three univariate polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ and the AGCD of three inexact polynomials $f(x)$, $g(x)$ and $h(x)$. Two variants of the three-polynomial subresultant matrix exist and these are analysed to determine the optimal form for computing the degree and coefficients of the three-polynomial GCD.

Chapter 6 The extensions necessary to compute the GCD of two or three bivariate polynomials defined over a triangular domain are considered. The definition of the Sylvester matrix and subresultant matrices is extended to the two-polynomial and three-polynomial GCD problems for bivariate polynomials. Results will again show that preprocessing is required to reliably determine the degree of the GCD particularly in the three-polynomial problem where the row and column-partitions of the

subresultant matrices must be balanced. This particular set of GCD problems arises in the computation of intersections involving triangular Bézier patches. More generally, these three-polynomial GCD problems arise in the square-free factorisation of a bivariate polynomial defined over a triangular domain.

Chapter 7 This chapter considers the extensions necessary to compute the GCD of two or three bivariate polynomials defined over a rectangular domain. These GCD problems arise in the computation of intersections between rectangular Bézier patches. A simple extension of the previously developed UGCD method gives rise to the bivariate GCD (BVGCD) method. However, this method has significant computational cost associated with it. A second, more efficient method is also considered in which the two-polynomial and three-polynomial bivariate GCD problem is solved in two one-dimensional stages. It will be shown that this new method, bivariate dimension reducing GCD (BVDRGCD), is significantly faster than the BVGCD method.

Chapter 8 This chapter concludes the work in the thesis. The key developments are summarised and ideas for future research are presented.

The remainder of this first chapter considers the background material and sets of alternative methods for (i) the curve and surface intersection problems, (ii) the polynomial root finding problem, (iii) the square-free factorisation problem and (iv) the polynomial GCD finding problem. The chapter has the following structure :

Section 1.1 This section describes the historical context of the curve and surface intersection problems within CAGD.

Section 1.2 Geometric representations of curves and surfaces are introduced.

Section 1.3 The curve and surface intersection problems are described and algorithms for solving these problems are considered. Of particular interest are intersections between Bézier curves and Bézier patches which reduce to computing the roots of a univariate or bivariate polynomial in Bernstein form.

Section 1.4 A variety of classical methods for polynomial root isolation and root finding are considered, some of which require that roots are first isolated with initial approximations. These root finding methods typically fail to accurately compute roots of high multiplicity (also referred to as multiple roots), yet it is this type of polynomial which arises in the computation of the points of intersection between curves and surfaces where the intersections are smooth.

Section 1.5 Univariate square-free factorisation methods are described. Given a square-free factorisation, the roots of a polynomial are more easily computed by conventional methods.

Section 1.6 The unstructured and structured condition numbers of a polynomial are defined in this section.

Section 1.7 The pejorative manifold of a polynomial is defined. A pejorative manifold gives a geometric interpretation of the conditions under which perturbations of the

exact polynomial $\hat{f}(x)$ cause its multiplicity structure to break down. The inexact polynomial $f(x)$ either has simple roots or has the same multiplicity as the exact form $\hat{f}(x)$.

Section 1.8 Polynomial GCD computation is one of the two fundamental components of Gauss' square-free factorisation algorithm described in Section 1.5. Several algorithms for the computation of the GCD of two polynomials are considered. Of particular interest are the matrix based methods such as the methods which utilise the Sylvester or Bézoutian matrix to compute the degree and coefficients of the GCD.

1.1 A Brief History of CAGD

The relatively modern development of CAGD has its origins in the automotive, shipbuilding and aerospace sectors. As the design of ships and motorcars became increasingly complex in scale and intricacy, the need for machine based construction techniques increased. By the 1950s, numerically controlled machinery became available, which allowed for the production of stamps and dyes for use in the manufacturing process. The introduction of CAGD and computer-aided manufacture (CAM) meant highly consistent products could be manufactured from reproducible tools. The difficulty faced by the industry at that time was in producing a numeric representation of the required shapes.

Early efforts focused on the laborious task of taking many thousands of individual measurements from blueprints or clay models which were fed into a numerically controlled (NC) machine from which the shape and surface geometry were approximated. Any changes in design meant that this process had to be repeated over again at great expense. This lead to the utilization of computers as part of the design process as well as the manufacturing process, eliminating the need for hand crafted prototypes.

It was de Casteljau, while working for the car manufacturer Citroën, who first used Bernstein polynomials to define curves and surfaces [18]. The Bernstein basis had been developed some fifty years earlier, in the early 1900s, as part of a solution to a problem in approximation theory. The work by de Casteljau was kept secret within Citroën, but work by Bézier developed around the same time for rival manufacturer Renault had similar results. This work was openly published and the resulting curves and surfaces bear his name.

The Bézier curve allowed for intuitive flexible design, and despite initial resistance from the design community, it became the industry standard. The control point based structure meant that a designer could alter the shape of a curve or surface by simply dragging and dropping control points.

Parametrically defined curves and surfaces such as Bézier curves and surfaces are used to represent large free flowing surfaces with minimal amounts of low level detail. A car body panel or aeroplane fuselage for example can be represented using Bézier patches, and this requires significantly less data than an alternative polygonal based modelling approach. Polygonal models, however, lend themselves to representing areas of low level detail. Surfaces can repeatedly be subdivided for areas of localised detail and this is much more difficult to achieve with Bézier patches. It is also difficult to produce sharp edges

using Bézier curves and surface patches. In comparison, sharp edges are easily achieved in a polygonal based model.

The accurate computation of the points of intersection of parametrically defined surfaces is a real industrial problem [19]. Holes which appear in computer generated models due to poor intersection approximations must be patched manually, and this can be a laborious manual process. Dokken and Skytt offer an in-depth analysis of the difficulties using floating point representations of curves and surfaces and the computation of points of smooth intersection between curves and surfaces. The merits of various intersection finding methods are considered in [19] such as surface triangulation, lattice evaluation, marching, refinement and recursive subdivision. The authors also consider the method used in this thesis where an intersection is reduced to a root finding problem by combining the use of parametric and algebraic representations. For instance, the intersection of two rational bicubic patches reduces to finding the zeros of a polynomial of bi-degree $(54, 54)$.

While focusing on the GCD and polynomial factorisation problem, this thesis considers methods for the computation of intersections of curves and surfaces defined either implicitly or parametrically. These curves and surfaces are now defined.

1.2 Curves and Surfaces and Their Representations

This section introduces the explicit, implicit and parametric curve definitions. Typical CAGD problems involve intersections of parametrically defined curves and surfaces, and this thesis focuses on the computation of intersections between Bézier curves and surfaces. It will be shown how these intersections reduce to the computation of the roots of a univariate polynomial or the factorisation of a bivariate polynomial in Bernstein form. The properties of different curve and surface representations and their respective applications are discussed.

Explicitly Defined Curves and Surfaces

An explicitly defined curve is of the form $y = \hat{f}(x)$, where y is dependent on the variable x. Similarly, an explicitly defined surface has the form $z = \hat{f}(x, y)$, and z is dependent on the variables x and y. These only represent a subset of all curves and surfaces, and this is due to the previously mentioned dependence between the variables x, y and z.

Implicitly Defined Curves and Surfaces

An implicit curve is defined as the set of all (x, y) coordinate pairs which satisfy the equation

$$\hat{f}(x, y) = 0.$$

Similarly, an implicitly defined surface in three-dimensional space is given by

$$\hat{f}(x, y, z) = 0.$$

The set of implicit curves in two-dimensional space can represent all possible curve shapes and the explicitly defined curves of the form $y = \hat{f}(x)$ are a subset of implicit curves, which can be written in the form $\hat{f}(x) - y = 0$. The implicit curve in three-dimensional space is the intersection of two implicitly defined surfaces and is given by $\hat{f}(x, y, z) \cap \hat{g}(x, y, z)$.

Parametrically Defined Curves and Surfaces

The most frequently used representation within CAGD is the parametric curve or surface. Occasionally trigonometric parametrizations are used, but more typically polynomial parametrizations are used. Of particular interest to this work are the Bézier curves and patches which will be defined later.

The parametric curve $\hat{g}(t)$ in two-dimensional space is defined by two functions in one variable, t, called the parameter, and is given by

$$x = \hat{g}_1(t) \quad \text{and} \quad y = \hat{g}_2(t).$$

A rational parametric curve is given by

$$x = \frac{\hat{g}_1(t)}{\hat{w}(t)} \quad \text{and} \quad y = \frac{\hat{g}_2(t)}{\hat{w}(t)},$$

where $\hat{w}(t)$ is a weighting function.
A parametrically defined surface in three-dimensional space is given by

$$x = \hat{g}_1(s, t), \quad y = \hat{g}_2(s, t), \quad \text{and} \quad z = \hat{g}_3(s, t).$$

All parametric curves have an implicit representation which can be obtained by implicitisation, but typically an approximate implicitisation is used to represent a parametric curve or surface since the exact implicit form is generally of high degree. A general triangular patch of degree n has an implicit form $\hat{f}(x, y, z) = 0$ of degree n^2, while a general tensor-product surface patch of degree (m, n) has an implicit representation $\hat{f}(x, y, z) = 0$ of degree $2mn$. For example, a bicubic tensor-product surface has an implicit equation $\hat{f}(x, y, z) = 0$ of degree 18 with $\binom{18+3}{3} = 1330$ coefficients [59]. A triangular cubic Bézier patch has an implicit representation $\hat{f}(x, y, z) = 0$ of degree 9.

Discussing the Curve and Surface Representations

The three representations have respective advantages and disadvantages when considered within CAGD. In rendering for example, an image is generated from a two or three-dimensional object. A parametric surface with parameters s and t is rendered by the evaluation for a set of s_i and t_i parameter pairs. There is no equivalent method for rendering an implicitly defined curve or surface, and generating the set of (x, y) or (x, y, z) points required for rendering is much more difficult.

Given a point (x_1, y_1) in two dimensions or (x_1, y_1, z_1) in three dimensions, it is easy to determine whether that point lies on, inside, or outside of an implicitly defined curve or surface. Simple substitution and evaluation of $\hat{f}(x_1, y_1)$ or $\hat{f}(x_1, y_1, z_1)$ will reveal the location of the point dependent on whether the evaluation is greater than, less than or

equal to zero.

Intersection problems are typically more easily solved with an implicit representation of the curves or surfaces involved and a breakdown of these problems will be given in Section 1.3.

1.3 Intersections of Curves and Surfaces

In this section methods for the computation of intersections between curves and surfaces are briefly considered. Texts such as [19] contain a much deeper analysis of the possible intersection problems. This thesis focuses on the set of problems which can either be reduced to finding the roots of a univariate polynomial in Bernstein form or finding an irreducible factorisation of a bivariate polynomial in Bernstein form.

In particular, this thesis focuses on problems where the roots of a univariate polynomial are of high multiplicity, and these root finding problems are typically associated with tangential intersection problems. Tangential intersections occur where two curves or surfaces are near parallel at their intersection and perturbations of two curves or surfaces with a tangential intersection can cause the intersection point to be lost. One possible result is that the perturbed curves no longer intersect, but instead have a distance ϵ between them. Alternatively, the perturbed curves or surfaces can overlap near to the intersection, causing a cluster of intersections to occur, rather than a unique point. This section discusses the types of intersection problems encountered when using the various curve and surface representations, and methods for their computation.

The first part of this section considers the computation of the intersections of two implicitly defined curves or surfaces. These problems most readily reduce to polynomial root finding (or factorisation finding) problems.

Secondly, the intersection of one implicitly defined object and a parametrically defined object similarly reduces to a root or factorisation problem. This problem first requires a substitution to obtain a polynomial whose roots are then computed.

Thirdly, the intersection of two parametrically defined objects (Bézier curves and surfaces, for example) are considered, but these problems require an implicitisation stage to reduce them to a problem of computing the intersection of a parametrically defined object and an implicitly defined object.

Implicit Curve and Surface Intersections

The following set of examples considers the intersection of two implicitly defined curves or surfaces. These problems reduce to the computation of the roots of a polynomial $\hat{h}(x) = 0$ or the factorisation of a bivariate polynomial $\hat{h}(x, y) = 0$. It will be shown how curves and surfaces with smooth intersections reduce to polynomial factorisation problems where the factors are of high multiplicity.

Example 1.3.1. Consider two implicitly defined curves given by

$$\hat{f}(x, y) = (x - 4)(x - 3)(x + 1)(x + 2) - y = 0$$
$$\hat{g}(x, y) = -3(x + 1)(x + 2)(7x + 11) - y = 0.$$

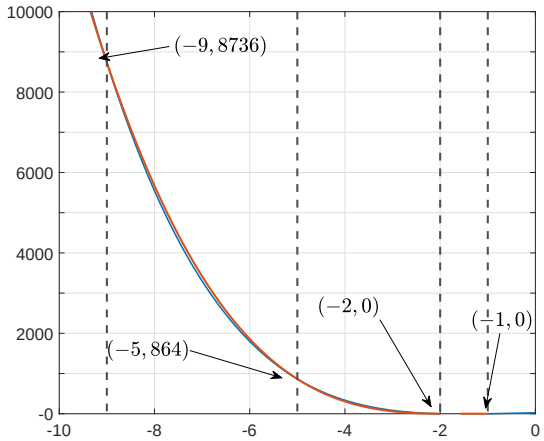
Either $\hat{f}(x,y)=0$ or $\hat{g}(x,y)=0$ can be written explicitly, so let the first curve be written in the form $y=\hat{f}(x)$, and this is substituted into $\hat{g}(x,y)=0$ such that a third polynomial $\hat{h}(x)=\hat{g}(x,\hat{f}(x))=0$ is given by

$$\begin{aligned}\hat{h}(x) &= (x+1)(x+2)\left((x-4)(x-3)-(-21x-33)\right)=0\\ &= (x+1)(x+2)(x+5)(x+9).\end{aligned}$$

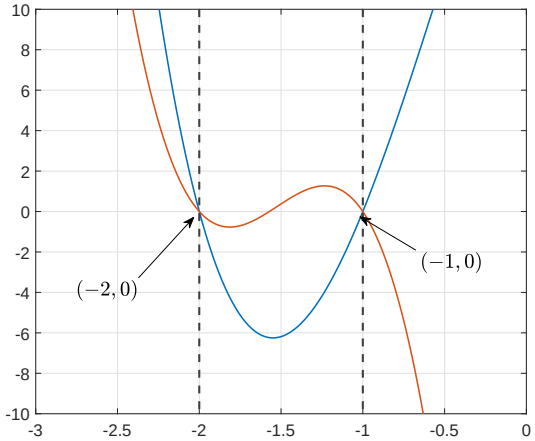
The real roots x_i of $\hat{h}(x)$ determine the x coordinate of the points of intersection of $\hat{f}(x)$ and $\hat{g}(x)$

$$x_1=-9,\quad x_2=-5,\quad x_3=-2\qquad\text{and}\qquad x_4=-1.$$

The intersection points are obtained by substituting the values x_i into either $\hat{f}(x,y)=0$



(i) Plotting $\hat{f}(x,y)$ and $\hat{g}(x,y)$ over the interval $[-10,0]$



(ii) Plotting $\hat{f}(x,y)$ and $\hat{g}(x,y)$ over the interval $[-3,0]$

Figure 1.1: Plotting the curves $\hat{f}(x,y)=0$ and $\hat{g}(x,y)=0$ and their intersection points in Example 1.3.1

or $\hat{g}(x,y)=0$ and solving for y_i, and are given by

$$(x_1,y_1)=(-9,\,8736)\,,\ (x_2,y_2)=(-5,\,864)\,,\ (x_3,y_3)=(-2,\,0)\ \text{ and }\ (x_4,y_4)=(-1,\,0)\,.$$

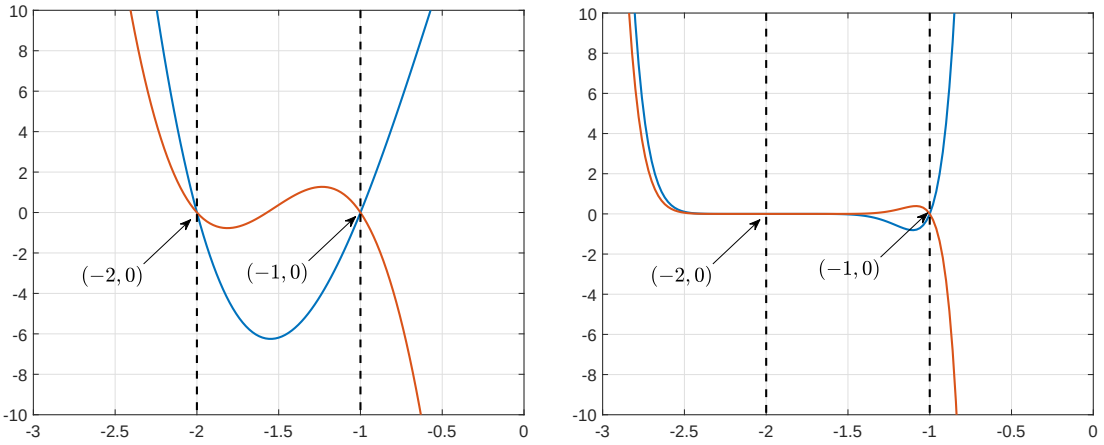
Figure 1.1i shows the four points of intersection in the interval $[-10,0]$, while Figure 1.1ii shows only the two points of intersection in the interval $[-3,0]$.

□

Example 1.3.2. Consider a modified version of Example 1.3.1, where the polynomials $\hat{f}_n(x,y)$ and $\hat{g}_n(x,y)$ are given by

$$\begin{aligned}\hat{f}_n(x,y) &= (x-4)(x-3)(x+1)(x+2)^n-y=0\\ \hat{g}_n(x,y) &= -3(x+1)(x+2)^n(7x+11)-y=0.\end{aligned}$$

Either $\hat{f}_n(x,y)=0$ or $\hat{g}_n(x,y)=0$ can be written explicitly, so the implicit equation



(i) $n = 1$ **(ii)** $n = 9$

Figure 1.2: The intersections of two curves $\hat{f}(x, y) = 0$ and $\hat{g}(x, y) = 0$ in the interval $[-3, 0]$ for various values of n in Example 1.3.2

$y = \hat{f}_n(x)$ can be substituted into $\hat{g}_n(x, y)$ such that $\hat{h}_n(x) = \hat{g}_n(x, \hat{f}_n(x)) = 0$ is given by

$$\begin{aligned}\hat{h}_n(x) &= (x+1)(x+2)^n\left((x-3)(x-4) - (-21x-33)\right)\\ &= (x+1)(x+2)^n(x+5)(x+9).\end{aligned}$$

The roots of $\hat{h}_n(x)$ are -1, -2, -5 and -9. The curves $\hat{f}_2(x, y) = 0$ and $\hat{g}_2(x, y) = 0$ are plotted in Figure 1.2i and the curves $\hat{f}_9(x, y) = 0$ and $\hat{g}_9(x, y) = 0$ are plotted in Figure 1.2ii. It can be seen that the intersection between the two curves $\hat{f}_9(x, y) = 0$ and $\hat{g}_9(x, y) = 0$ is significantly smoother than the intersection between $\hat{f}_2(x, y) = 0$ and $\hat{g}_2(x, y) = 0$, and this smooth intersection gives rise to a polynomial root finding problem with a root of high multiplicity.

□

The smoothness of the intersection of two curves is determined by the multiplicity of the roots of $\hat{h}(x) = 0$, where higher multiplicity in the roots is equivalent to smoother intersections. Roots of high multiplicity are likely to be incorrectly computed when using standard root finding methods so it is necessary to first compute the square-free factorisation, from which simple roots can be computed. The simple roots obtained from the square-free factorisation are the roots of $\hat{h}(x) = 0$, and this algorithm is discussed in more detail in Section 1.5.

A surface intersection problem similarly reduces to the computation of the factors of $\hat{h}(x, y) = 0$, and factors with high multiplicity are associated with smooth intersections.

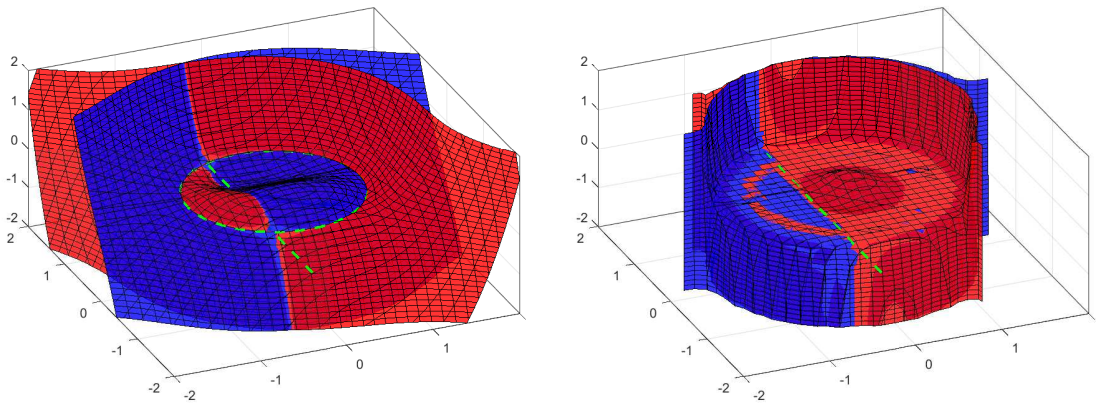
Example 1.3.3. Consider the two implicitly defined surfaces given by

$$\begin{aligned}\hat{f}(x, y, z) &= (x^2 + y^2 - 1)^n \times (x + y + 0.2) - z = 0\\ \hat{g}(x, y, z) &= (x^2 + y^2 - 1)^n \times (y - 0.2) - z = 0.\end{aligned}$$

The intersections between the two surfaces are the factors of

$$\begin{aligned}\hat{h}(x,y,z) &= (x^2+y^2-1)^n \times ((x+y+0.2)-(y-0.2)) \\ &= (x^2+y^2-1)^n(x-0.4)\end{aligned}$$

so there is a circle of intersection given by $(x^2+y^2-1)=0$ and a line of intersection given by $x+0.4=0$. Larger values of n give smoother intersections around the circle $x^2+y^2-1=0$ as shown in Figure 1.3.



(i) $n=1$ **(ii)** $n=10$

Figure 1.3: The intersections of two surfaces $\hat{f}(x,y,z)=0$ (■) and $\hat{g}(x,y,z)=0$ (■) in Example 1.3.3

□

Typically in real world applications the intersection of two surfaces P (Consisting of bicubic patches P_i) and Q (Consisting of bicubic patches Q_j) reduces to the computation of any intersections of the P_i and Q_j patches. Despite these patches being of modest degree, such intersections involve zero finding of bivariate polynomials of significantly higher degree. The focus of this thesis is to develop the matrix based GCD finding methods and polynomial deconvolution necessary for such intersection problems.

Newton's Method for the Computation of Intersections Between Two Implicitly Defined Curves

The intersections of two implicitly defined curves $\hat{f}(x,y)=0$ and $\hat{g}(x,y)=0$ are computed using Newton's method. Given an initial approximation, (x_0,y_0), the iterative procedure generates successive approximations (x_{i+1},y_{i+1}) given by

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} \delta x_i \\ \delta y_i \end{bmatrix},$$

where δx_i and δy_i are given by

$$\begin{bmatrix} \frac{\partial \hat{f}(x_i,y_i)}{\partial x} & \frac{\partial \hat{f}(x_i,y_i)}{\partial y} \\ \frac{\partial \hat{g}(x_i,y_i)}{\partial x} & \frac{\partial \hat{g}(x_i,y_i)}{\partial y} \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \end{bmatrix}^{(i)} = - \begin{bmatrix} \hat{f}(x_i,y_i) \\ \hat{g}(x_i,y_i) \end{bmatrix}^{(i)}.$$

The iterative process is terminated when a solution (x_i, y_i) satisfies the conditions that $\left|\hat{f}(x_i, y_i)\right| \leq \epsilon$ and $|\hat{g}(x_i, y_i)| \leq \epsilon$ for some threshold value ϵ or when a maximum number of iterations has been reached.

An initial approximation of the point of intersection is required, and intersection points can only be found one at a time. As with the root finding implementation of Newton's method, the method may be divergent, and termination can only be achieved when the intersection is computed to within some threshold ϵ of its exact value, and this threshold must be predetermined. For these reasons, the implementation of the intersection method based on Newton's method is reserved for refining already approximated intersection points obtained by other means.

The method is extended to the computation of the points of intersection of two implicitly defined surfaces $\hat{f}(x, y, z) = 0$ and $\hat{g}(x, y, z) = 0$. The intersection point (x_i, y_i, z_i) is given by

$$\begin{bmatrix} \frac{\partial \hat{f}(x_i,y_i,z_i)}{\partial x} & \frac{\partial \hat{f}(x_i,y_i,z_i)}{\partial y} & \frac{\partial \hat{f}(x_i,y_i,z_i)}{\partial z} \\ \frac{\partial \hat{g}(x_i,y_i,z_i)}{\partial x} & \frac{\partial \hat{g}(x_i,y_i,z_i)}{\partial y} & \frac{\partial \hat{g}(x_i,y_i,z_i)}{\partial z} \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \\ \delta z \end{bmatrix} = - \begin{bmatrix} \hat{f}(x_i, y_i, z_i) \\ \hat{g}(x_i, y_i, z_i) \end{bmatrix}^{(i)}.$$

This initial point on the intersection curve can then be used in a tracing based method to compute the curve of intersection.

Implicit and Parametric Curve and Surface Intersections

The points of intersection between the implicit curve $\hat{f}(x, y) = 0$ and the parametric curve defined by equations $x = \hat{g}_1(t)$, $y = \hat{g}_2(t)$ are given by the following process:

1. The variables x and y in $\hat{f}(x, y) = 0$ are replaced by the corresponding parametric equations $x = \hat{g}_1(t)$ and $y = \hat{g}_2(t)$ such that a new equation $\hat{h}(t) = \hat{f}(\hat{g}_1(t), \hat{g}_2(t)) = 0$ is given. The roots $\{t_i\}$ of the univariate polynomial $\hat{h}(t) = 0$ are computed and these are substituted back into the parametric equations $x = \hat{g}_1(t)$ and $y = \hat{g}_2(t)$ to compute x_i and y_i pairs, which are the set of intersection points.

2. The intersections of the implicitly defined surface given by $\hat{f}(x, y, z) = 0$ and the parametric surface defined by $x = \hat{g}_1(s, t)$, $y = \hat{g}_2(s, t)$ and $z = \hat{g}_3(s, t)$ can be computed by the factorisation of the polynomial $\hat{f}(\hat{g}_1(s, t), \hat{g}_2(s, t), \hat{g}_3(s, t)) = \hat{h}(s, t) = 0$.

Intersections of Two Parametric Curves or Surfaces

The intersection of two parametrically defined surfaces $S_1(s, t)$ and $S_2(s, t)$ requires that one of the two surfaces is first implicitised. The implicitisation of the surface $S_1(s, t)$ of degree (m_1, m_2) has degree $d = 2m_1m_2$ and has $\binom{d+3}{3}$ coefficients. For example, a bicubic Bézier patch has an implicit form of total degree 18, and has 1330 coefficients. Having implicitised one of the polynomials, the problem reduces to one of the set of problems listed earlier in the section.

Implicitisation is the process in which, given a parametric curve or surface, an implicit form is found. This is a simple process for curves and surfaces of low degree. However, as the degree of the parametric surface increases, the number of coefficients of the implicit

representation increases. In the literature much work has been completed on the various methods of finding local and approximate implicit forms [1, 13, 24, 35, 46, 57, 58, 60].

Example 1.3.4 shows the implicitisation of a parametrically defined curve.

Example 1.3.4. Consider the parametrically defined curve C_1 which is defined by the parametric equations

$$\hat{f}_1(t) = 0.1 + 0.8t - 0.1t^2$$
$$\hat{f}_2(t) = 1 - 3.4t + 3.5t^2.$$

The standard form of implicitisation by the Sylvester matrix based method [59, Section 4.1] requires the construction of the $2n \times 2n$ Sylvester matrix containing coefficients of

$$\hat{f}_1(t) = 0.1 + 0.8t - 0.1t^2 - x$$
$$\hat{f}_2(t) = 1 - 3.4t + 3.5t^2 - y$$

which is given by

$$S\left(\hat{f}_1(t), \hat{f}_2(t)\right) = \left[\begin{array}{cc|cc} -0.1 & 0 & 3.5 & 0 \\ 0.8 & -0.1 & -3.4 & 3.5 \\ (0.1-x) & 0.8 & (1-y) & -3.4 \\ 0 & (0.1-x) & 0 & (1-y) \end{array}\right].$$

The implicit expression is given by the determinant of the above matrix

$$\hat{h}(x, y) = \frac{49}{4}x^2 + \frac{7}{10}xy + \frac{1}{100}y^2 - \frac{5757}{500}x - \frac{1029}{500}y + \frac{30069}{10000}.$$

□

Bézier Subdivision

This method can be utilized in computing the intersection of two Bézier curves. The convex hull property can be used to determine whether two curves intersect [43]. An absence of intersections of convex hulls can be used to exclude intersections of curves, but an intersection of convex hulls does not guarantee curve intersections. The region of intersection is reduced by subdividing and checking for intersections between the new set of curves. Given that the convex hulls of two Bézier curves C_1 and C_2 intersect, the intersection can be found by subdividing C_1 into $C_{1,Left}$ and $C_{1,Right}$, and C_2 into $C_{2,Left}$ and $C_{2,Right}$.

The convex hulls in each pairing (i) $C_{1,Left}$ and $C_{2,Left}$, (ii) $C_{1,Left}$ and $C_{2,Right}$, (iii) $C_{1,Right}$ and $C_{2,Left}$ and (iv) $C_{1,Right}$ and $C_{2,Right}$ are then checked for possible intersections. Those which do not intersect are rejected while convex hulls which do intersect may contain an intersection point. The curves are repeatedly subdivided and checked for intersections until the subdivided curves can be approximated by straight lines. The intersection of these lines gives an approximation of the exact intersection point.

An improvement to this algorithm is developed by Sederberg [61], which introduces the

concept of fat arcs, and this method converges more quickly than the standard clipping algorithm. One limitation of this algorithm is that a threshold is required to determine when a curve is sufficiently flat to be approximated by a straight line.

In Section 1.3 several curve and surface intersection finding methods have been considered. It has been shown that the algebraic methods generally reduce to the computation of the roots or factors of univariate or bivariate polynomials, and methods for solving these problems are discussed in Section 1.4 and Section 1.5 respectively.

1.4 Polynomial Real Root Finding

This section considers the computation of the real roots of a univariate polynomial $\hat{f}(x)$. Given $\hat{f}(x)$ of degree m, $x_0 \in \mathbb{R}$ is a root of $\hat{f}(x)$ if $\hat{f}(x_0) = 0$. Or, equivalently $(x - x_0)$ is a factor of $\hat{f}(x)$. The multiplicity m_0 of the factor $(x - x_0)$ is equivalently the multiplicity of the root and a polynomial of degree m has at most m distinct real roots.

Roots of high multiplicity are of particular interest in the application of this work since it is these polynomials which define smooth intersections in curve and surface intersection problems. A multiple root of a polynomial $\hat{f}(x)$ is, however, sensitive to small perturbations in the coefficients of $\hat{f}(x)$, which can cause the root to break up into simple roots. It is therefore necessary to consider methods which preserve the multiplicity structure of the roots. Given that this structure is maintained, the roots are well conditioned.

The computation of the intersection of two curves was shown to reduce to root finding problem. The intersection of two cubic Bézier curves requires that one curve is in parametric form while the other is in implicit form. The implicit form of a parametric curve defined in terms of parametric equations of degree three, is similarly of degree three. The intersection problem therefore reduces to finding the roots of a polynomial of degree nine. The root finding and GCD finding problems in this thesis often involve polynomials of degree 20 or more, and this is to highlight the robustness of the algorithms used.

Numerous methods have been considered for the computation of the roots of a polynomial. Laguerre's method [30,37] is an algorithm which always converges to a complex root given any starting point. Other methods, such as Newton's method and its variants [45], require an initial approximation which is sufficiently close to a root, but the method may still be divergent. Interval bisection based methods are slow to converge and may fail to identify roots of even multiplicity. Other methods make use of properties of polynomials in Bernstein form. For instance, the convex hull is used in clipping algorithms [3, 45, 47]. This section considers some of the classical algorithms and their suitability to the root finding problem at hand.

Section 1.4.1 discusses some root isolation techniques for square-free polynomials, as well as the determination of an interval containing all roots based on the polynomial coefficients. Section 1.4.2 discusses the traditional polynomial root finding algorithms, which are particularly useful when polynomials are of low degree and contain simple roots, and these methods work best when roots have already been isolated. Section 1.4.3 discusses the difficulties in computing polynomial roots by a conventional method using the MATLAB `roots()` function.

1.4.1 Bounds on the Number and Size of Real Roots of a Polynomial and Root Isolation Techniques

The fundamental theorem of algebra states that a univariate polynomial with complex coefficients must have at least one complex root, and a polynomial $\hat{f}(x)$ of degree m has m complex roots.

Descartes' Rule of Signs

For polynomials in the power basis, Descartes' rule of signs is used to determine an upper bound for the number of positive roots. The number of positive roots, denoted n, is given by

$$\mathrm{n} = v\left(\hat{f}(x)\right) - 2k$$

for some value of $k \in \mathbb{Z}_+$, and $v\left(\hat{f}(x)\right) \in \mathbb{Z}_+$ is the number of changes of sign in the ordered coefficients of $\hat{f}(x)$.

Example 1.4.1. Consider the polynomial $\hat{f}(x)$ given by

$$\hat{f}(x) = 5x^5 - 2x^4 + 2x^3 - 3x^2 + 2x - 5,$$

which has five changes of signs in its ordered coefficients. The number of positive real roots of $\hat{f}(x)$ is given by

$$n = 5 - 2k \qquad \text{for some} \quad k \geq 0.$$

□

The upper limit of the number of negative roots of $\hat{f}(x)$ is given by the number of changes of sign in the coefficients of $\hat{f}(-x)$

$$n = v\left(\hat{f}(-x)\right) - 2k \qquad \text{for some} \quad k \geq 0.$$

Fourier's Theorem

The Fourier sequence, denoted F_{seq}, is the set of $(m+1)$ polynomials

$$F_{seq}(\hat{f}(x)) = \left\{\hat{f}(x), \hat{f}'(x), \hat{f}^{(2)}(x), \ldots, \hat{f}^{(m)}(x)\right\}.$$

Let $v(F_{seq}(\hat{f}(a)))$ denote the number of sign variations of the Fourier sequence evaluated at $x = a$, and $v(F_{seq}(\hat{f}(b)))$ denote the number of sign variations of the Fourier sequence evaluated at $x = b$, then the number of real roots in the interval (a, b), where $a < b$, is given by

$$n = v\left(F_{seq}\left(\hat{f}(a)\right)\right) - v\left(F_{seq}\left(\hat{f}(b)\right)\right) - 2k,$$

where $k \in \mathbb{Z}_+$.

Sturm's Theorem

Sturm's theorem builds on Fourier's theorem, but the sequence $F_{seq}(\hat{f}(x))$ is replaced by the sequence $S_{seq}(\hat{f}(x))$ which consists of remainders obtained by polynomial long division. A Sturm chain can be used to compute the number of roots in a given interval. By this method, roots can be isolated in ever decreasing intervals. The Sturm chain of a polynomial $\hat{f}(x, y)$ is given by

$$\begin{aligned}
\hat{f}_0(x) &= \hat{f}(x) \\
\hat{f}_1(x) &= \hat{f}_0'(x) \\
\hat{f}_2(x) &= -\text{rem}\left(\hat{f}_0(x), \hat{f}_1(x)\right) \\
&\vdots \\
\hat{f}_{i+1}(x) &= -\text{rem}\left(\hat{f}_{i-1}(x), \hat{f}_i(x)\right) \\
&\vdots \\
0.
\end{aligned}$$

Let $v(\hat{f}(x))$ be the number of changes of sign in the sequence $\hat{f}_0(x)$, $\hat{f}_1(x)$, ..., $\hat{f}_m(x)$, then the number of roots in the interval (α, β) is given by

$$n = v\left(\hat{f}(\alpha)\right) - v\left(\hat{f}(\beta)\right).$$

For a polynomial $\hat{f}(x)$ of degree m, all of its roots are bounded by $[-M, M]$, where

$$M = 1 + \frac{\max\{\hat{a}_0, \hat{a}_1, \ldots, \hat{a}_{m-1}\}}{\hat{a}_m},$$

where $\hat{a}_i$ are the coefficients of $\hat{f}(x)$.

The roots of a polynomial can therefore be isolated using this method by subdividing the interval $[-M, M]$ until each subinterval contains at most one root. The computation of the Sturm chain is, however, computationally inefficient and methods for root isolation based on Descartes' rule of signs are more effective.

Roots of Polynomials in Bernstein Form by Transformation

The polynomial in Bernstein form defined over the unit interval is given by

$$\hat{f}(t) = \sum_{i=0}^{m} \hat{a}_i B_i^n(t) \qquad \text{for} \quad 0 \le t < 1$$

by the substitution $t = \frac{x}{1+x}$ this can be written as

$$\begin{aligned}
\hat{f}\left(\frac{x}{1+x}\right) &= \sum_{i=0}^{m} \hat{a}_i \left(1 - \frac{x}{1+x}\right)^{m-i} \left(\frac{x}{1+x}\right)^i \\
&= \frac{1}{(1+x)^m} \sum_{i=0}^{m} \hat{a}_i x^i \qquad \text{for} \quad 0 < x < \infty
\end{aligned}$$

This transformation is used in later sections to compare alternate methods with the method developed in this thesis, and it is clear to see that the number of roots of $\hat{f}(t)$ in the interval $[0, 1)$ is given by the number of changes of sign in the coefficients $\hat{a}_i$ minus a nonnegative even integer as described in the section on Descartes' rule of sign. For roots t_i of $\hat{f}(t)$ approximate to 1, the roots in x, x_i tend to infinity, and small perturbations in t_i result in large changes of x_i.

Bounds on the Number of Roots of Polynomials in Bernstein Form by variation diminishing property (VDP)

The upper bound of the number of roots of a polynomial $\hat{f}(x)$ in Bernstein form is given by the number of intersections of its control polygon and the x-axis. This is due to the VDP which states that a polynomial in Bernstein form is at least as smooth as its control polygon.

1.4.2 Polynomial Root Finding Algorithms

Polynomial root finding is a classical problem with a long history and this section describes some classical root finding techniques. The first two methods described are the interval bisection method and the method of false position. Both make use of bounding intervals which are known to contain a polynomial root. The bounding intervals are iteratively shrunk until the root can be approximated by the intervals midpoint.

Newton's method and the secant method do not utilise bounding boxes, and are said to be 'open' root finding methods. This often results in faster convergence, but can also lead to divergence away from the root, depending on the shape of the polynomial function. Another method, Brent's method, makes use of the bisection method, secant method and inverse quadratic interpolation and offers the same reliability as the bisection method, but with a faster rate of convergence.

The last two methods, Bézier subdivision and convex hull marching, are used specifically for finding zeros of Bézier curves because they make use of properties of Bézier curves such as the VDP and the convex hull property.

Interval Bisection

The interval bisection root finding method, also known as interval halving or binary search, is perhaps the most trivial of all root finding methods. Suppose a continuous function $\hat{f}(x)$ is given over a closed interval $[a, b]$, where $\hat{f}(a)\hat{f}(b) < 0$, then $\hat{f}(x)$ has at least one root in the interval. This result is a specific case of the intermediate value theorem and is called Bolzano's theorem.

The algorithm works by generating a sequence of intervals of decreasing size, $\{[a_k, b_k]\}$, in which a root of $\hat{f}(x)$ is known to be contained and proceeds as follows:

1. Set $k = 0$, $a_k = a$ and $b_k = b$. The midpoint of the interval is given by $c_k = \frac{a_k + b_k}{2}$.
2. If $\hat{f}(c_k) = 0$, then the root is found and the algorithm terminates.
3. If $b_k - a_k \leq \theta$, then the interval is sufficiently small and the polynomial root is approximated as the midpoint c_k.

4. If $\hat{f}(a_k)\hat{f}(c_k) < 0$, then the root lies in the first half of the interval $[a_k, b_k]$. The algorithm is called again for the interval $[a_{k+1}, b_{k+1}] = [a_k, c_k]$.

5. Otherwise, $\hat{f}(c_k)\hat{f}(b_k) < 0$ and the root lies in the second half of the interval, so the algorithm is called again for the interval $[a_{k+1}, b_{k+1}] = [c_k, b_k]$.

A threshold value θ is required to determine when the size of the interval $[a_k, b_k]$ is sufficiently small for a root to be approximated by the interval midpoint.

The bisection algorithm is slow when compared with other methods, but other methods do not always guarantee convergence. Given that a root is known to be contained within an interval, it is either at the interval midpoint, or contained in one of the two interval halves. Therefore, the algorithm will always converge. However, if the root is of even multiplicity it may be missed completely. For example, let α be a root of the polynomial $\hat{f}(x)$ with even multiplicity, then $\hat{f}(\alpha - \epsilon)$ has the same sign as $f(\alpha + \epsilon)$. A lack of change of sign within an interval means that the algorithm fails to spot that a root is contained, and the root is therefore likely missed by the bisection algorithm.

Before the bisection algorithm can be applied, an interval $[a, b]$ must be selected where $\hat{f}(a)\hat{f}(b) < 0$, and this interval must isolate the root. Such intervals can be obtained by root isolation techniques which were discussed in Section 1.4.1.

Regula Falsi / False Position

If the function $\hat{f}(x)$ is differentiable, a more appropriate root finding method is the method of false position. As with the interval bisection method, the algorithm for false position produces a sequence of intervals $[a_k, b_k]$ of decreasing size, which contain the root α of the polynomial $\hat{f}(x)$. The algorithm proceeds as follows:

1. Two initial values of a_0 and b_0 are chosen such that $\hat{f}(a_0)\hat{f}(b_0) < 0$.

2. The value c_k is given by the intersection of the straight line between the points $(a_k, \hat{f}(a_k))$ and $(b_k, \hat{f}(b_k))$.

3. If $\hat{f}(c_k)$ is sufficiently close to zero, then the algorithm terminates and the root α is equal to c_k.

4. If $\hat{f}(a_k)\hat{f}(c_k) < 0$, then the root lies in the interval $[a_{k+1}, b_{k+1}] = [a_k, c_k]$.

5. If $\hat{f}(c_k)\hat{f}(b_k) < 0$, then the root lies in the interval $[a_{k+1}, b_{k+1}] = [c_k, b_k]$.

As with the bisection method, the method of false position keeps the root bounded and always converges, but typically does so at a faster rate than the bisection method.

Newton's Method

Perhaps the most famous of the classical root finding algorithms is Newton's method. Given an initial root approximation x_0 of $\hat{f}(x)$, a new approximation x_1 is given by the intersection of the tangent of $\hat{f}(x)$ at $(x_0, \hat{f}(x_0))$ and the x-axis. The $(k+1)$th approximation x_{k+1} is given by the intersection of the tangent of $\hat{f}(x)$ at $(x_k, \hat{f}(x_k))$ and the x-axis,

so is given by

$$x_{k+1} = x_k - \frac{\hat{f}(x_k)}{\hat{f}'(x_k)}.$$

The algorithm terminates when the evaluation of $\hat{f}(x_k)$ is sufficiently close to zero, and for this a threshold θ is required.

Newton's method is not guaranteed to converge unless certain conditions are satisfied. If any approximation x_k is a stationary point, the derivative $\hat{f}'(x_k)$ is equal to 0, and x_{k+1} is undefined. Also, some starting points may result in a sequence of approximations which are cyclic. For instance, in a 2-cycle, the approximations $x_0 = x_2 = x_4 = \cdots = x_k$ and $x_1 = x_3 = \cdots = x_{k+1}$ and the approximations clearly fail to converge on the root. However, choosing an alternative starting approximation can overcome this particular problem.

When the algorithm does converges it does so at a quadratic or linear rate dependent on whether the root is a multiple root.

The Secant Method

The secant method is similar to the method of false position, in that a straight line is used to approximate a curve between two points x_0 and x_1, where the interval is known to contain the root. A new point is calculated at the intersection of the line and the x-axis. However, the secant method is open and unlike the method of false position, the root is not bounded. The secant method is a finite difference approximation of Newton's method. It converges more quickly than the false position method, but in some cases can fail to converge. A straight line is constructed between two initial approximations, x_0 and x_1. The point of intersection between the straight line between x_{k-2} and x_{k-1} and the x-axis is denoted x_k, and a new line between x_k and x_{k-1} is considered.

Brent's Method

Brent's method [11], like the bisection method requires an initial bounding interval which is known to contain the root which is to be approximated. This is a hybrid method which makes use of the bisection, secant and inverse quadratic interpolation root finding methods, to offer a reliable root finding method which typically converges more quickly than the bisection method alone.

Bézier Subdivision (Schneider's Algorithm)

This algorithm, described by Schneider in [34, Chapter 8.2], computes the roots of $\hat{f}(x)$, a polynomial in Bernstein form, by repeatedly subdividing the curve. By the variation diminishing property, a line which intersects the control polygon of $\hat{f}(x)$ n times intersects the curve $\hat{f}(x)$ at most n times. In this algorithm, one of three possible scenarios arise:

1. If no intersections occur between the control polygon of $\hat{f}(x)$ and the x-axis, then $\hat{f}(x)$ has no real roots in the interval.

2. If there is more than one intersection between the control polygon of $\hat{f}(x)$ and the x-axis, then the curve $\hat{f}(x)$ is subdivided at its midpoint and the algorithm is recursively called for the two halves of the subdivided curve.

3. If there is only one intersection between the control polygon and the x-axis, and if the control polygon is deemed "flat enough", then the curve is approximated by a line segment which passes through the bounding box of the control polygon. The root is then given by the intersection of the approximating line segment and the x-axis.

This method recursively subdivides the curve until all roots are isolated. However, the algorithm relies on a threshold to determine whether the control polygon is "flat enough".

Convex Hull Marching

The method of convex hull marching relies on the convex hull property of a polynomial in Bernstein form, described in Section 2.2.1. All roots are approached from the left, and cannot be skipped over. This method removes one root at a time, but the removal of inexact roots can cause an accumulation of errors due to division.

Given a polynomial $\hat{f}(x)$ of degree m in Bernstein form, the first intersection of the convex hull of $\hat{f}(x)$ with the x-axis, denoted x_0, is evaluated to determine whether it is sufficiently close to a root α. If $x_0 \neq \alpha$, the curve is subdivided at the point $(x_0, \hat{f}(x_0))$ giving two new curves, $\hat{f}_{left}(x)$ in the interval $[0, x_0]$ and $\hat{f}_{right}(x)$ in the interval $[x_0, 1]$. The left partition is discarded since it does not contain any roots. A new polynomial $\hat{f}_1(x)$ of degree m is given, which has a new convex hull. The process is iterated until the intersection of the convex hull and the x-axis, given by x_k, is sufficiently close to a root. The root is removed from $\hat{f}(x)$ and the process begins again for the polynomial $\hat{f}^*(x)$.

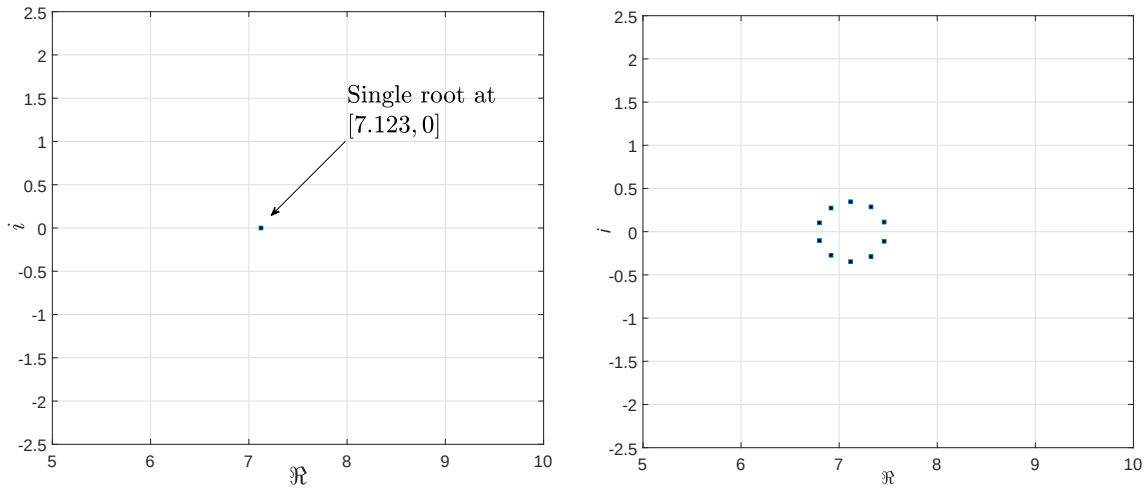
About Classical Root Finding Methods

The classical root finding methods such as the bisection method and Newton's method are typically only useful when determining a single root which has a good initial approximation or bounding interval. These algorithms can be used in two ways:

1. The set of roots $\{r_i\}$ of $\hat{f}_0(x)$ can be computed successively. The ith root r_i is computed given $\hat{f}_i(x)$, which is deflated by dividing by the factor $(x - r_i)$ and the next root is computed from $\hat{f}_{i+1}(x)$. This repeated division can be unstable, particularly when the computed roots are rough approximations of the exact roots.

   However, this can be overcome using methods described in [52], where it is shown that repeated deflation can be stable given sufficiently accurate root approximations and each root is removed in an order dependent on the absolute value of the complete set of roots [54, Section 9.5].

2. Alternatively, bounding intervals of each root r_i of $\hat{f}_i(x)$ are predetermined, and the roots can be computed simultaneously as long as there is good separation between the roots.



(i) Computed roots of $\hat{f}_1(x)$ **(ii)** Computed roots of $\hat{f}_{10}(x)$

Figure 1.4: The approximations of the roots of $\{\hat{f}_i(x) \mid i = 1, 10\}$ computed using MATLAB `roots()` in Example 1.4.2

Neither of the approaches above can effectively compute polynomial roots of high multiplicity, and it is therefore advantageous to consider methods which first compute a factorisation of $\hat{f}(x)$ resulting in a set of polynomials $\{w_i(x)\}$, whose easily obtained simple roots are the roots of $\hat{f}_i(x)$ with multiplicity i.

1.4.3 Roots Computed by MATLAB `roots()`

There are many difficulties in computing the real roots of a polynomial. Firstly, roots of high multiplicity are not reliably computed by standard root finding methods as will be shown in Example 1.4.2. The MATLAB `roots()` function computes the roots of a polynomial $f(x)$ which is a floating-point representation of $\hat{f}(x)$. The computed roots are typically of multiplicity one.

Example 1.4.2. This example considers the set of polynomials $\{\ \hat{f}_i(x) \mid i = 1, \ldots, m\ \}$ given by

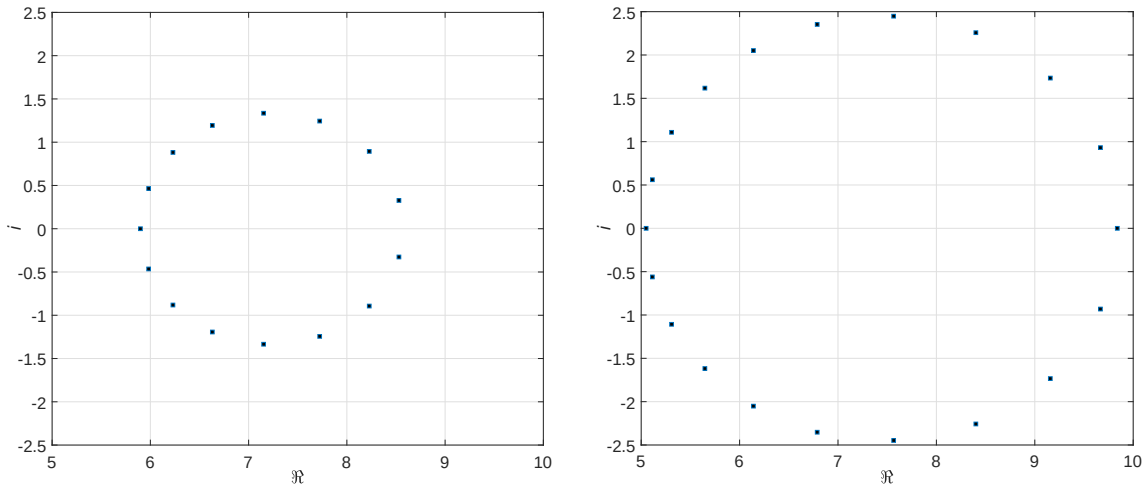
$$\hat{f}_i(x) = (x - \hat{r})^i,$$

where $\hat{r} = 7.123456789$.

The roots of the set of polynomials $\{\hat{f}_i(x)\}$ are approximated by MATLAB `roots()` and are plotted in Figure 1.4 and Figure 1.5 for $i = 1,\ 10,\ 15$ and 20. The set of computed roots of $\hat{f}_i(x)$ are denoted by $\{\, r_{i,j} \mid j = 1, \ldots, i\,\}$. So, $\{r_{5,j}\}$ is the set of five computed roots of $\hat{f}_5(x)$, while $\{r_{10,j}\}$ is the set of ten computed roots of $\hat{f}_{10}(x)$.

The forward error is a measure of the distance between the roots of $\hat{f}_i(x)$ and the computed roots $\{\, r_{i,j} \mid j = 1, \ldots, i\,\}$. Let λ_i denote the average Euclidean distance between the exact root $\hat{r}$ and the set of computed roots of $\{\, \hat{f}_i(x) \mid j = 1, \ldots, i\,\}$ such that λ_i is given by

$$\lambda_i = \frac{\sum_{j=1}^{i} |\hat{r} - r_{i,j}|}{i}.$$



(i) Computed roots of $\hat{f}_{15}(x)$ **(ii)** Computed roots of $\hat{f}_{20}(x)$

Figure 1.5: The approximations of the roots of $\{\hat{f}_i(x) \mid i = 15, 20\}$ computed using MATLAB `roots()` in Example 1.4.2

From Figure 1.4 it can be seen that the radius of the computed roots $\{r_{i,j}\}$ increases (almost linearly) as the root multiplicity i increases, and the forward error λ_i is plotted in Figure 1.6i.

The backward error will be defined as the distance between the input polynomial $\hat{f}_i(x)$ and the polynomial $f_i(x)$, whose roots are given by the set $\{r_{i,j} \mid j = 1, \ldots, i\}$, that is,

$$f_i(x) = (x - r_1)(x - r_2) \ldots (x - r_i).$$

The relative error between the vectors of the coefficients of the two polynomials $\hat{f}_i(x)$ and $f_i(x)$ is given by

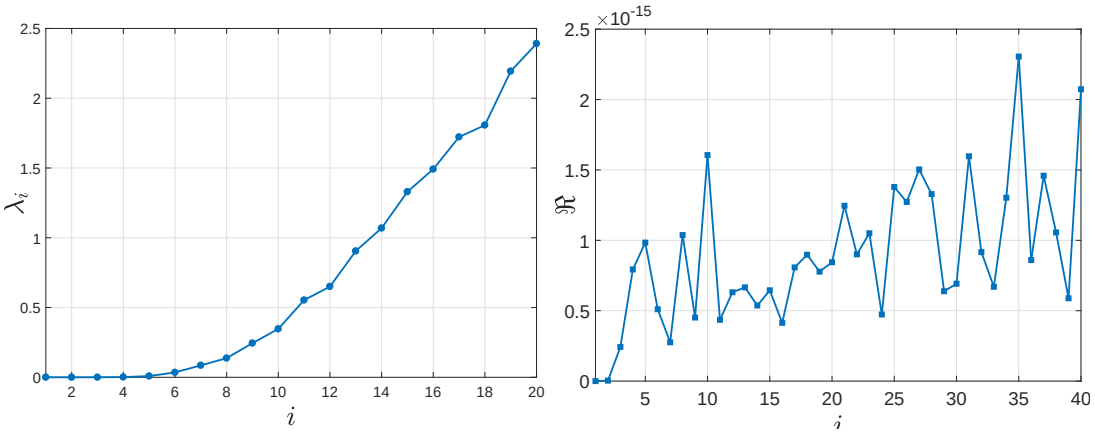
$$\mu_i = \frac{\left\| \hat{\mathbf{f}} - \mathbf{f}_i \right\|_2}{\left\| \hat{\mathbf{f}} \right\|_2},$$

where $\hat{\mathbf{f}}$ and $\hat{\mathbf{f}}_i$ are vectors of the coefficients of the polynomials $\hat{f}(x)$ and $\hat{f}_i(x)$. In Figure 1.6ii it is shown that the backward error μ_i is consistently small.

As previously stated, the MATLAB `roots()` function computes the exact roots of a floating point representation, $f(x)$, of the exact polynomial $\hat{f}(x)$.

□

This example has shown how roots of high multiplicity tend to be incorrectly computed as a cluster of simple roots. As root multiplicity increases, so too does the radius of the cluster of computed roots. Secondly, when two roots of high multiplicity are close, the clusters of computed roots overlap such that they are no longer separable. While the MATLAB `roots()` function accurately determines the roots of the floating point representation of $\hat{f}(x)$, methods which maintain the multiplicity structure of the roots of the exact polynomial must instead be considered.



(i) Forward error λ_i for $i = 1, \ldots, 20$ in Example 1.4.2

(ii) Backward error μ_i for $i = 1, \ldots, 20$ in Example 1.4.2

Figure 1.6: Forward error $\{\lambda_i\}$ and backward error $\{\mu_i\}$ of the computed roots $\{r_i\}$ in Example 1.4.2

1.5 Polynomial Factorisation Algorithms

It has been shown that a curve-curve intersection problem can be reduced to the determination of the square-free factorisation of a univariate polynomial, that is, a factorisation such that $\hat{f}(x)$ is expressed as a set of square-free polynomials $\{\hat{w}_i(x)\}$, each of multiplicity i. The simple roots of the square-free polynomial $\hat{w}_i(x)$ are computed and these are the roots of $\hat{f}(x)$ with multiplicity i. This method gives better results than standard root finding methods since the root multiplicity structure is preserved.

Several papers use a similar square-free factorisation method. Musser describes a set of polynomial factorisation algorithms in [49], while [66, 73, 78] all use variations of the same square-free factorisation algorithm which is described in Section 1.5.1. Yun refers to it as being "due to Tobey and Horowitz" [76], however an earlier version of the algorithm is found in [63] and it is believed to have originated from Gauss. In this thesis it is described as "the algorithm due to Gauss" or "Gauss' Algorithm".

The square-free polynomial and square-free factorisation are now defined and the definition from [33] is used with modified notation which is consistent with the remainder of this thesis.

Definition 1. *Let $\hat{f}(x) \in \mathbb{R}[x]$ be a primitive polynomial over a unique factorisation domain $\mathbb{R}$. Then $\hat{f}(x)$ is square-free if it has no repeated factors, that is, if there exists no $\hat{b}(x)$ with $deg(\hat{b}(x)) \geq 1$ such that*

$$\hat{b}(x)^2 | \hat{f}(x).$$

The square-free factorisation of $\hat{f}(x)$ is given by

$$\hat{f}(x) = \prod_{i=1}^{r} \hat{w}_i^i(x),$$

where each $\hat{w}_i(x)$ *is a square-free polynomial and*

$$GCD\,(\hat{w}_i(x), \hat{w}_j(x)) = 1 \text{ for } i \neq j,$$

that is, each pair of polynomials in the set $\{\hat{w}_i(x)\}$ *are coprime.*

It is convenient to define the polynomial $\hat{f}(x)$ as the product of the factors $\{\hat{w}_i(x)\}$, where each $\hat{w}_i$ is the product of factors of degree i in $\hat{f}(x)$. If $\hat{f}(x)$ does not contain a factor of multiplicity i, then $\hat{w}_i(x)$ is set equal to a constant

$$\hat{f}(x) = \sum_{i=0}^{m} \hat{a}_i \binom{m}{i} (1-x)^{m-i} x^i = \hat{w}_1(x)\hat{w}_2^2(x)\ldots\hat{w}_r^r(x). \tag{1.1}$$

This factorisation is important for the understanding of the methods of Gauss and Musser, which are now described.

1.5.1 Gauss' Algorithm

Gauss' algorithm [63] relies on the principle that if $\hat{f}(x)$ is a polynomial in a unique factorisation domain and $\hat{f}(x)$ is square-free, then

$$\text{GCD}\left(\hat{f}(x), \hat{f}'(x)\right) = 1,$$

otherwise

$$\text{GCD}\left(\hat{f}(x), \hat{f}'(x)\right) = \hat{f}^*(x),$$

where $\hat{f}^*(x)$ is given by the polynomial $\hat{f}(x)$ with all multiplicities of its factors reduced by one. This property is used to compute the square-free decomposition of the polynomial $\hat{f}(x)$. The outputs of the algorithm, denoted $\{\hat{w}_i(x)\}$, are square-free polynomials where each $\hat{w}_i$ is the product of the factors of $\hat{f}_0(x)$ with multiplicity i.

Algorithm 1: Square-free factorization due to Gauss

Input: $\hat{f}_0(x) = \hat{w}_1(x)\hat{w}_2^2(x)\ldots\hat{w}_r^r(x)$

$\hat{f}_1(x) \leftarrow \text{GCD}\left(\hat{f}_0(x), \hat{f}_0'(x)\right) = \hat{w}_2(x)\hat{w}_3^2(x)\ldots\hat{w}_r^{r-1}(x)$

$\hat{h}_1(x) \leftarrow \frac{\hat{f}_0(x)}{\hat{f}_1(x)} = \hat{w}_1(x)\hat{w}_2(x)\ldots\hat{w}_r(x)$

$i \leftarrow 1$;

while $h_i(x) \neq 1$ **do**

  $\hat{f}_{i+1}(x) \leftarrow \text{GCD}\left(\hat{f}_i(x), \hat{f}_i'(x)\right) = \hat{w}_{i+2}(x)\hat{w}_{i+3}^2(x)\ldots\hat{w}_r^{r-i-1}(x)$

  $\hat{h}_{i+1}(x) \leftarrow \frac{\hat{f}_i(x)}{\hat{f}_{i+1}(x)} = \hat{w}_i(x)\hat{w}_{i+1}(x)\ldots\hat{w}_r(x)$

  $\hat{w}_i(x) \leftarrow \frac{\hat{h}_i(x)}{\hat{h}_{i+1}(x)}$

  $i \leftarrow i+1$

end

set $\hat{w}_i(x) = \hat{h}_i(x)$

return $\hat{w}_1(x), \hat{w}_2(x), \ldots, \hat{w}_{i-1}(x)$

The computation of the set of polynomials $\{\hat{f}_i(x)\}$ is independent of the computation

of the sets $\{\hat{h}_i(x)\}$ or $\{\hat{w}_i(x)\}$ so these can be determined first, followed by the set $\{\hat{h}_i\}$ and finally the set $\{\hat{w}_i(x)\}$, as shown in Example 1.5.1.

Example 1.5.1. Consider the polynomial $\hat{f}(x)$, whose factorised form is given by

$$\hat{f}(x) = (x - 0.2)^7 (x - 0.3)^{12}.$$

The set of polynomials $\{\, \hat{f}_i(x) \mid i = 0, \ldots, 12 \,\}$ is given by

$$\begin{aligned} \hat{f}_i(x) &= GCD\left(\hat{f}_{i-1}(x), \hat{f}'_{i-1}(x)\right) \\ &= \begin{cases} (x - 0.3)^{12}(x - 0.2)^7 & i = 0, \\ (x - 0.3)^{12-i}(x - 0.2)^{7-i} & i = 1, \ldots, 6, \\ (x - 0.3)^{12-i} & i = 7, \ldots, 11, \\ 1 & i = 12. \end{cases} \end{aligned}$$

The set of polynomials $\{\, \hat{h}_i(x) \mid i = 1, \ldots, 12 \,\}$ is given by

$$\hat{h}_i(x) = \begin{cases} (x - 0.3)(x - 0.2) & i = 1, \ldots, 7, \\ (x - 0.3) & i = 8, \ldots, 12. \end{cases}$$

Finally, the set of polynomials $\{\, \hat{w}_i(x) \mid i = 1, \ldots, 12 \,\}$ is given by

$$w_i(x) = \begin{cases} 1 & i \in [1, 6] \cup [8, 11], \\ (x - 0.2) & i = 7, \\ (x - 0.3) & i = 12. \end{cases}$$

Therefore, the factors of $\hat{f}(x)$ are $(x - 0.2)$ and $(x - 0.3)$ with multiplicity 7 and 12 respectively. □

1.5.2 Musser's Polynomial Factorisation Algorithm

The algorithm due to Musser [49] given in Algorithm 2 is a more efficient version of the algorithm due to Gauss. Both methods share the same intermediate values, however, all

but the first polynomial differentiations are removed in Musser's method.

Algorithm 2: Musser's square-free factorisation algorithm

Input: $\hat{f}_0(x) = \hat{w}_1(x)\hat{w}_2^2(x)\dots\hat{w}_r^r(x)$

$\hat{f}_1(x) \leftarrow \text{GCD}\left(\hat{f}_0(x), \hat{f}_0'(x)\right) = \hat{w}_2(x)\hat{w}_3^2(x)\dots\hat{w}_r^{r-1}(x)$

$\hat{h}_1(x) \leftarrow \frac{\hat{f}_0(x)}{\hat{f}_1(x)} = \hat{w}_1(x)\hat{w}_2(x)\dots\hat{w}_r(x)$

$i \leftarrow 1$

while $h_i(x) \neq 1$ **do**

  $\hat{h}_{i+1}(x) \leftarrow \text{GCD}\left(\hat{f}_i(x), \hat{h}_i(x)\right) = \hat{w}_{i+1}(x)\hat{w}_{i+2}(x)\dots\hat{w}_r(x)$

  $\hat{f}_{i+1}(x) \leftarrow \frac{\hat{f}_i(x)}{\hat{h}_{i+1}(x)} = \hat{w}_{i+2}(x)\hat{w}_{i+3}^2(x)\dots\hat{w}_r^{r-i-1}(x)$

  $\hat{w}_i(x) \leftarrow \frac{\hat{h}_i(x)}{\hat{h}_{i+1}(x)} = \hat{w}_i(x)$

  $i \leftarrow i + 1$

end

return : $\hat{w}_1(x), \hat{w}_2(x), \dots, \hat{w}_{i-1}(x)$

Given a general polynomial $\hat{f}(x)$ with k factors each of multiplicity m_k, the ith GCD computation is performed on a polynomial $\hat{f}_i(x)$ and a square-free polynomial $\hat{h}_i(x)$.

If $\hat{f}(x)$ does not have a root of multiplicity i then $\hat{h}_{i+1}(x)$, computed in the ith GCD computation, is equal to $\hat{h}_i(x)$. In which case

$$\deg\left(\hat{h}_{i+1}(x)\right) = \deg\left(\hat{h}_i(x)\right).$$

This type of GCD problem (where the GCD is equal to one of the input polynomials) is particularly problematic for the UGCD method developed in this work. The method computes the degree of the GCD of two polynomials $\hat{f}(x)$ and $\hat{g}(x)$ by determining the index of the last rank deficient subresultant matrix in the subresultant matrix sequence. There are two cases for which this method fails:

1. The first is when the two polynomials are coprime, in which case all subresultant matrices are nonsingular (full rank) and therefore a change in their numerical rank does not exist.

2. The second is when the GCD of $\hat{f}(x)$ and $\hat{g}(x)$ is equal to $\hat{g}(x)$ (assuming $\deg(g(x)) < \deg(f(x))$), in which case all subresultant matrices are singular or rank deficient.

In Musser's algorithm it is the second exception which occurs frequently.

Example 1.5.2. Consider the exact polynomial $\hat{f}(x)$, whose factorised form is given by

$$\hat{f}(x) = (x - 0.2)^3(x - 0.7)^5.$$

The algorithm produces the following output

$$\begin{array}{rcccl}
\hat{f}_0(x) & & & = & (x-0.2)^2(x-0.7)^3 \\
\hat{f}_1(x) & = & \mathrm{GCD}\left(\hat{f}_0(x), \hat{f}_0'(x)\right) & = & (x-0.2)(x-0.7)^2 \\
\hat{h}_1(x) & = & \frac{\hat{f}_0(x)}{\hat{f}_1(x)} & = & (x-0.2)(x-0.7) \\
\hat{h}_2(x) & = & \mathrm{GCD}\left(\hat{f}_1(x), \hat{h}_1(x)\right) & = & (x-0.2)(x-0.7) \\
\hat{f}_2(x) & = & \frac{\hat{f}_1(x)}{\hat{h}_2(x)} & = & (x-0.7) \\
\hat{w}_1(x) & = & \frac{\hat{h}_1(x)}{\hat{h}_2(x)} & = & 1 \\
\hat{h}_3(x) & = & \mathrm{GCD}\left(\hat{f}_2(x), \hat{h}_2(x)\right) & = & (x-0.7) \\
\hat{f}_3(x) & = & \frac{\hat{f}_2(x)}{\hat{h}_3(x)} & = & 1 \\
\hat{w}_2(x) & = & \frac{\hat{h}_2(x)}{\hat{h}_3(x)} & = & (x-0.2) \\
\hat{h}_4(x) & = & \mathrm{GCD}\left(\hat{f}_3(x), \hat{h}_3(x)\right) & = & 1 \\
\hat{f}_4(x) & = & \frac{\hat{f}_3(x)}{\hat{h}_4(x)} & = & 1 \\
\hat{w}_3(x) & = & \frac{\hat{h}_3(x)}{\hat{h}_4(x)} & = & (x-0.7).
\end{array}$$

Therefore, the factors of $\hat{f}_0(x)$ are $(x-0.2)$ and $(x-0.7)$ with multiplicities two and three respectively. □

Note on Square Free Factorisation and Root Finding

This work uses a square free factorisation method to approximate the roots of a polynomial where the roots are of high multiplicity. Conversely, the computation of the roots of the polynomials $\hat{f}^{(n-1)}(x), \hat{f}^{(n-2)}(x), \ldots, \hat{f}^{(1)}(x)$ can be used to compute the square-free factorisation of $\hat{f}(x)$. The roots of the polynomial $\hat{f}^{(n-2)}(x)$ are found between the roots of $\hat{f}^{(n-1)}(x)$ and Fourier sign rule can be used to determine their locations. However, this thesis assumes that the problem of root finding is hard and a method of square-free factorisation must be employed to maintain the multiplicity structure of the polynomials roots.

The complexity of the square-free factorisation problem is dependent on the multiplicity structure of the polynomial roots. Smooth intersections are associated with polynomials whose roots are of high multiplicity. The algorithm requires a sequence of GCD computations for two polynomials.

As stated earlier, the intersection of two parametric curves whose parametric equations are of degree three typically reduces to the computation of the roots of a polynomial of degree nine. In which case the polynomial $\hat{f}_0(x)$ and its derivative $\hat{f}_0'(x)$ are of degree 9 and 8 respectively.

Polynomials in subsequent GCD computations are of a lower degree and the algorithm terminates when $\hat{f}_i(x)$ and its derivative $\hat{f}_i'(x)$ are coprime, that is, when $\hat{f}_i(x)$ is square-free. Suppose r denotes the highest multiplicity of any of the roots of $\hat{f}(x)$, then r GCD computations are required in Musser's square-free factorisation algorithm.

1.6 The Condition of a Multiple Root

The previous section considered the computation of the square-free factorisation of a polynomial. Given the square-free factorisation, the polynomial's multiple roots are more easily obtained when using classical root finding methods. This section now considers the structured and unstructured condition numbers of a polynomial's roots. Further discussion of the condition number of a polynomials roots can be found in [28].

The *unstructured condition number* of a root α, is defined by the addition of random perturbations to the coefficients of $\hat{f}(x)$, in which case it can be assumed that the r roots at α break up into simple roots.

The *structured condition number* of α is defined by the addition of structured perturbations to the coefficients of $\hat{f}(x)$, such that the perturbed form of $\hat{f}(x)$ has a root $(\alpha+\delta\alpha)$ of multiplicity r, that is, the value of the root changes, but the multiplicity structure is maintained.

Theorem 1. *Let the monic polynomial $\hat{f}(x)$ have coefficients $\hat{a}_i, i = 0, \ldots, m$, with respect to the basis $\phi_i(x)$ such that $\hat{f}(x)$ is given by*

$$\hat{f}(x) = \sum_{i=0}^{m} \hat{a}_i \phi_i(x).$$

Now let the coefficients $\hat{a}_i$ be perturbed to $a_i = \hat{a}_i + \delta\hat{a}_i$, where

$$|\delta\hat{a}_i| \leq \epsilon \, |\hat{a}_i| \,, \qquad i = 0, \ldots, m,$$

and where ϵ is the upper bound of the relative error.

Let the real root α of $\hat{f}(x)$ have multiplicity r, and let one of these r roots be perturbed to $(\alpha + \delta\alpha)$, where $\delta\alpha^{(\mathrm{i})}$ is the perturbation of the root due to the perturbations in the coefficients $\hat{a}_i$ as described above. The unstructured componentwise condition number of α is defined in [65] and is given by

$$\kappa(\alpha) = \max_{|\delta\hat{a}_i| \leq \epsilon |\hat{a}_i|} \frac{|\delta\alpha|}{|\alpha|} \frac{1}{\epsilon} = \frac{1}{\epsilon^{1-\frac{1}{r}}} \frac{1}{|\alpha|} \left(\frac{r!}{\left| \hat{f}^{(r)}(\alpha) \right|} \sum_{i=0}^{m} |\hat{a}_i \phi_i(\alpha)| \right)^{\frac{1}{r}}. \tag{1.2}$$

The structured condition number requires that $\hat{f}(x)$ be written in terms of its p distinct roots, $\tilde{\alpha} = \{\alpha_i \mid i = 1, \ldots, p\}$, where each α_i is a distinct root, and is of multiplicity m_i. The structured condition number is defined where the multiplicity structure of the roots is maintained.

$$\hat{f}(x, \tilde{\alpha}) = \prod_{i=1}^{p} (\theta(x, \alpha_i))^{m_i}$$

where, for example,

$$\theta(x, \alpha_k) = (x - \alpha_k) \,, \qquad \frac{\partial\theta(x, \alpha_k)}{\partial\alpha_k} = -1 \tag{1.3}$$

[(i)] Note that the notation $\delta\alpha$ describes a perturbation of α, and not a product $\delta \times \alpha$

for the power basis, and

$$\theta(x,\alpha_k) = (1-\alpha_k)\,y - \alpha_k\,(1-x)\,, \qquad \frac{\partial\theta(x,\alpha_k)}{\partial\alpha_k} = -1 \tag{1.4}$$

for the Bernstein basis.

The perturbed form of $\hat{f}(x,\tilde{\alpha})$ is therefore given by

$$\hat{f}(x,\tilde{\alpha}+\delta\tilde{\alpha}) = \prod_{i=1}^{p}\left(\theta\left(x,\alpha_i+\delta\alpha_i\right)\right)^{m_i} \approx \hat{f}(x,\tilde{\alpha}) + \sum_{i=1}^{p}\frac{\partial\hat{f}(x,\tilde{\alpha})}{\partial\alpha_i}\delta\alpha_i, \tag{1.5}$$

to first order.

Theorem 2. *The structured condition number of a root α_k, $\rho(\alpha_k)$, is defined as*

$$\rho(\alpha_k) = \frac{\|\theta(x,\alpha_k)\|}{m_k\,|\alpha_k|}, \tag{1.6}$$

where

$$\rho(\alpha_k) = \frac{\Delta\alpha_k}{\Delta\hat{f}(x,\tilde{\alpha})}, \quad \Delta\alpha_k = \frac{|\delta\alpha_k|}{|\alpha_k|}, \quad \Delta\hat{f}(x,\tilde{\alpha}) = \frac{\left\|\delta\hat{f}(x,\tilde{\alpha})\right\|}{\left\|\hat{f}(x,\tilde{\alpha})\right\|}.$$

and $\delta\hat{f}(x,\tilde{\alpha})$ is calculated from (1.5).

Proof. It follows from (1.5) that, to first order,

$$\begin{aligned}
\hat{f}(x,\tilde{\alpha}+\delta\tilde{\alpha}) &= \prod_{i=1}^{p}\left(\theta(x,\alpha_i) + \frac{\partial\theta(x,\alpha_i)}{\partial\alpha_i}\right)^{m_i} \\
&= \prod_{i=1}^{p}\left((\theta(x,\alpha_i))^{m_i} + m_i(\theta(x,\alpha_i))^{m_i-1}\frac{\partial\theta(x,\alpha_i)}{\partial\alpha_i}\delta\alpha_i\right) \\
&= \prod_{i=1}^{p}(\theta(x,\alpha_i))^{m_i} + \sum_{i=1}^{p}\left(\prod_{j=1,j\neq i}^{p}(\theta(x,\alpha_j)^{m_j}\right) m_i(\theta(x,\alpha_i)^{m_i-1}\frac{\partial\theta(x,\alpha_i)}{\partial\alpha_i}\delta\alpha_i
\end{aligned}$$

hence

$$\sum_{i=1}^{p}\frac{\partial\hat{f}(x,\tilde{\alpha})}{\partial\alpha_i}\delta\alpha_i = \prod_{j=1}^{p}\left(\theta(x,\alpha_j)\right)^{m_j}\sum_{i=1}^{p}\frac{m_i}{\theta(x,\alpha_i)}\frac{\partial\theta(x,\alpha_i)}{\partial\alpha_i}\delta\alpha_i.$$

There are p condition numbers, one for each of the p distinct roots in the set $\{\alpha_i \mid i = 1,\ldots,p\}$. The condition number of the root α_k is obtained by specifying

$$\delta\alpha_i = \begin{cases} 0 & i = 1,\ldots,k-1,k+1,\ldots,p \\ \delta\alpha_k, & i = k \end{cases}$$

and hence

$$\frac{\partial\hat{f}(x,\tilde{\alpha})}{\partial\alpha_k} = \prod_{j=1}^{p}\left(\theta(x,\alpha_j)\right)^{m_j}\frac{m_k}{\theta(x,\alpha_k)}\frac{\partial\theta(x,\alpha_k)}{\delta\alpha_k} = m_k\frac{\hat{f}(x,\tilde{\alpha})}{\theta(x,\alpha_k)}\frac{\partial\theta(x,\alpha_k)}{\partial\alpha_k}$$

It therefore follows from (1.3) and (1.4) that for the power and Bernstein bases,

$$\delta\hat{f}(x,\tilde{\alpha}_k) = -\frac{m_k}{\theta(x,\alpha_k)}\hat{f}(x,\tilde{\alpha})\delta\alpha_k$$

and thus

$$\left\|\theta(x,\alpha_k)\delta\hat{f}(x,\tilde{\alpha}_k)\right\| = m_k\,|\delta\alpha_k|\,\left\|\hat{f}(x,\tilde{\alpha}_k)\right\| \tag{1.7}$$

from which, the result (1.6) follows □

Note how the unstructured condition number $\kappa(\alpha)$ is a function of the upper bound of the relative error in the coefficients ϵ, while the structured condition number, $\rho(\alpha_k)$, of a root α_k is independent of ϵ.

Example 1.6.1. Consider the polynomial $\hat{f}(x)$ with one root α of multiplicity m, where $0 < \alpha \leq 1$, given by

$$\hat{f}(x) = \Big(-\alpha(1-x) + (1-\alpha)x\Big)^m = \sum_{i=0}^{m} \hat{a}_i \binom{m}{i} (1-x)^{m-i}x^i.$$

If m is sufficiently large, the unstructured condition number (1.2) is given by

$$\kappa(\alpha) \approx \frac{1}{\epsilon\,|\alpha|}$$

and a decrease in ϵ causes an increase in $\kappa(\alpha)$.

The structured condition number is easily obtained from (1.6) and

$$\rho(\alpha) < \frac{\|(-\alpha(1-x)+(1-\alpha)x)^m\|}{m\,|\alpha|\,\left\|(-\alpha(1-x)+(1-\alpha)x)^{m-1}\right\|} \leq \frac{\left(\alpha^2+(1-\alpha)^2\right)^{\frac{1}{2}}}{m\,|\alpha|}$$

assuming the two norm is used, and thus

$$\frac{1}{\sqrt{2}m\alpha} \leq \rho(\alpha) \leq \frac{1}{m\alpha} \quad \text{for} \quad 0 < \alpha \leq 1.$$

A decrease in ϵ, that is, a decrease in the upper bound of the relative error in the coefficients of b_i, causes an increase in the unstructured condition number $\kappa(\alpha)$. The bounds on the structured condition number $\rho(\alpha)$ are independent of ϵ, and as the multiplicity m increases the bounds decrease. The two condition numbers $\kappa(\alpha)$ and $\rho(\alpha)$ differ greatly and the multiple root α is stable if the perturbation of $\hat{f}(x)$ preserves its multiplicity. □

1.7 The Pejorative Manifold of a Polynomial

Let $\hat{f}(x)$ have p distinct roots α_i, $i = 1, \ldots, p$, each of multiplicity m_i, such that $\hat{f}(x)$ can be written as

$$\hat{f}(x) = \sum_{i=0}^{m} \bar{a}_i \binom{m}{i} (1-x)^{m-i} x^i = \prod_{i=1}^{p} \left(\alpha_i(1-x) - (1-\alpha_i)x\right)^{m_i},$$

where

$$\hat{a}_0 = \prod_{i=1}^{p} \alpha_i^{m_i}, \qquad \hat{a}_m = \prod_{i=1}^{p} \left(-(1-\alpha_i)\right)^{m_i} \quad \text{and} \quad \sum_{i=1}^{p} m_i = m.$$

The pejorative manifold of a polynomial is defined by the multiplicities of its roots, and it is convenient to consider the monic form $\hat{g}(x)$ of $\hat{f}(x)$. The polynomial $\hat{g}(x)$ is obtained by normalising the coefficients of $\hat{f}(x)$ by $\hat{a}_0$ and is given by

$$\hat{g}(x) = \sum_{i=0}^{m} \hat{b}_i \binom{m}{i} (1-x)^{m-i} x^i = \prod_{i=1}^{p} \left((1-x) - \lambda_i x\right)^{m_i}, \tag{1.8}$$

where

$$\hat{b}_i = \frac{\hat{a}_i}{\hat{a}_0} \qquad \text{and} \qquad \lambda_i = \frac{1-\alpha_i}{\alpha_i} \quad \text{where} \quad \alpha_i \neq 0.$$

Equation (1.8) shows that there exists a set of functions $\{\, h_i(\lambda_1, \ldots, \lambda_p) \mid i = 1, \ldots, m_i \,\}$ that define the transformation between the coefficients $\{\, b_i \mid i = 1, \ldots, m\}$ and the set of parameters $\{\, \lambda_i \mid i = 1, \ldots, p \,\}$ given by

$$\begin{aligned} b_1 &= h_1(\lambda_1, \ldots, \lambda_p) \\ b_2 &= h_2(\lambda_1, \ldots, \lambda_p) \\ &\vdots \\ b_m &= h_m(\lambda_1, \ldots, \lambda_p). \end{aligned}$$

Definition 2. *Let $\mu = \{m_1, m_2, \ldots, m_p\}$ be the set of multiplicities of the polynomial $\hat{g}(x)$. The pejorative manifold $\mathcal{M}_\mu \subset \mathbb{R}^m$ is the set of real coefficients $\{\hat{b}_1, \ldots \hat{b}_m\}$ such that $\hat{g}(x)$ has p distinct roots whose multiplicities are equal to μ.*

Example 1.7.1. The polynomial $f(x)$ of degree $m = 3$ with roots α_1, α_2 and α_3 is given by

$$\begin{aligned} = \alpha_1\alpha_2\alpha_3 \binom{3}{0}(1-x)^3 - \frac{(\alpha_1\alpha_2\beta_3 + \alpha_2\alpha_3\beta_1 + \alpha_3\alpha_1\beta_2)}{\binom{3}{1}} \binom{3}{1}(1-x)^2 x \\ + \frac{(\alpha_1\beta_2\beta_3 + \alpha_2\beta_3\beta_1 + \alpha_3\beta_1\beta_2)}{\binom{3}{2}} \binom{3}{2}(1-x)x^2 - \beta_1\beta_2\beta_3 \binom{3}{3} x^3, \end{aligned}$$

where $\beta_i = 1 - \alpha_i$. There are two distinct pejorative manifolds for a cubic polynomial:

(i) A curve $\mathcal{C}$ defines the pejorative manifold of a cubic polynomial with one cubic root, that is, $\alpha_1 = \alpha_2 = \alpha_3$.

(ii) A surface $\mathcal{S}$ defines the pejorative manifold of a cubic polynomial with one simple root α_1 and one double root α_2.

First suppose that the polynomial $f(x)$ has a real triple root $\alpha = \alpha_1 = \alpha_2 = \alpha_3$

$$f(x) = (\alpha(1-x) - (1-\alpha)x)^3 = \sum_{i=0}^{3} (-1)^i \alpha^{3-i}(1-\alpha)^i \binom{3}{i} (1-x)^{3-i}x^i,$$

whose monic form is given by

$$g(x) = \binom{3}{0}(1-x)^3 - \lambda\binom{3}{1}(1-x)^2x + \lambda^2\binom{3}{2}(1-x)x^2 - \lambda^3\binom{3}{3}x^3,$$

where $\lambda = \frac{1-\alpha}{\alpha}$, $\alpha \neq 0$ and the curve $\mathcal{C}$ is therefore given by

$$\mathcal{C} : (X(\lambda), Y(\lambda), Z(\lambda)) = \left(-\lambda,\ \lambda^2,\ -\lambda^3\right).$$

Consider now the situation in which $\hat{f}(x)$ has a double root α_1 and a simple root α_2

$$\begin{aligned}\hat{f}(x) = \alpha_1^2\alpha_2\binom{3}{0}(1-x)^3 &- \frac{\left(\alpha_1^2(1-\alpha_2) + 2\alpha_1\alpha_2(1-\alpha_1)\right)}{\binom{3}{1}}\binom{3}{1}(1-x)^2x \\ &+ \frac{\left(2\alpha_1(1-\alpha_1)(1-\alpha_2) + \alpha_2(1-\alpha_1)^2\right)}{\binom{3}{2}}\binom{3}{2}(1-x)x^2 \\ &- (1-\alpha_1)^2(1-\alpha_2)\binom{3}{3}x^3.\end{aligned}$$

The monic form of $\hat{f}(x)$ can be considered by defining λ and μ as

$$\lambda = \frac{1-\alpha_1}{\alpha_1} \qquad \text{and} \qquad \mu = \frac{1-\alpha_2}{\alpha_2}, \qquad \alpha_1, \alpha_2 \neq 0, \tag{1.9}$$

and thus the pejorative manifold $\mathcal{S}$ of a real cubic Bernstein basis polynomial that has one simple root and one double root is given by

$$\mathcal{S} : \begin{pmatrix} X(\lambda,\mu) & Y(\lambda,\mu) & Z(\lambda,\mu) \end{pmatrix} = \left(\frac{-(2\lambda+\mu)}{3},\ \frac{\lambda(\lambda+2\mu)}{3},\ -\lambda^2\mu\right).$$

□

The Pejorative Manifold and the Square-Free Factorisation

Square-free factorisation methods such as the methods due to Gauss and Musser work to first preserve the multiplicity structure of the polynomial $\hat{f}(x)$, equivalent to determining the manifold on which the $f(x)$ lies. The set of GCD computations and polynomial deconvolutions (In this context meaning polynomial division, described in more detail in Section 2.2.1) work to find where on the manifold the polynomial lies.

If the given polynomial $\hat{f}(x)$ is exact, then it is represented by a point on a manifold $\mathcal{M}$, and GCD computations and polynomial deconvolutions are performed exactly. However, if the coefficients of $f(x)$ are inexact, it can be assumed that the roots are all simple and

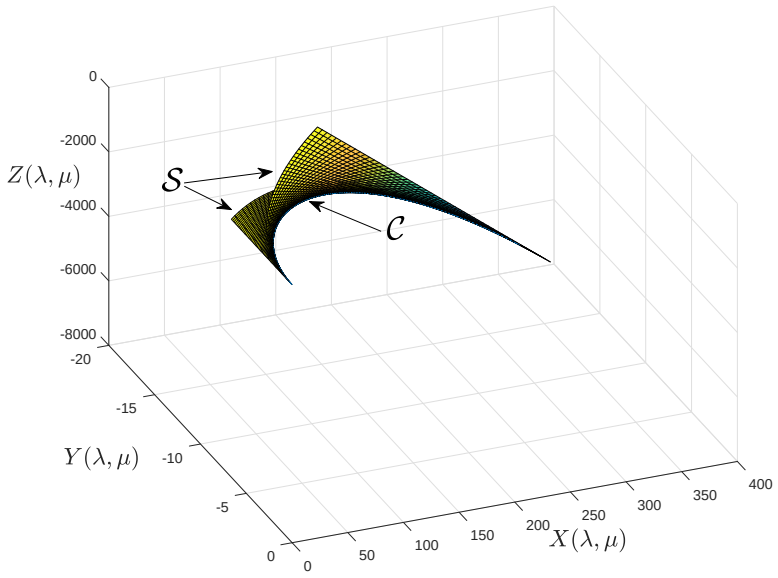


Figure 1.7: The curve $\mathcal{C}$ and the surface $\mathcal{S}$ as defined in Example 1.7.1

Stage	Numerical Operation	Geometric Operation
(1) Computation of the multiplicity of roots of $f(x)$	GCD computations and polynomial deconvolutions	Identification of the pejorative manifold
(2) Computation of root values	Solutions of polynomial equations $w_i(x) = 0$, all of whose roots are simple	Identification of the point on $\mathcal{M}$ that defines the polynomial

therefore do not lie on the manifold $\mathcal{M}$. In this case, the GCD computations project onto a manifold $\mathcal{M}^*$.

This section has described the motivation for determining a polynomial's square-free factorisation. It has already been stated that a fundamental component of the square-free factorisation algorithm is the polynomial GCD computation, and methods for this computation are considered in the next section.

1.8 Methods for the Computation of the GCD and AGCD

It was shown in Section 1.5 that the algorithms for square-free factorisation require a sequence of GCD computations. The computation of the GCD of two polynomials has applications in many areas. For instance, one method involved in image deblurring considers the pixel values in the rows and columns of an image as polynomials. An unknown blurring function is computed by taking the GCD of the polynomial representation of two rows of pixels [2, 53, 67]. GCD computations are also used for the decoding and error correction of Reed-Solomon codes which are used to encode data on compact discs [31].

Algorithms for computing the intersection of two algebraic curves require that the curves f and g be coprime before their resultant is computed, so the common factors must first be removed. Similarly, computations on rational functions $f(x) = \frac{a(x)}{b(x)}$ may require

that the two polynomials $a(x)$ and $b(x)$ be coprime, with common factors removed, which requires a GCD computation. For instance, this is required in the degree reduction of rational Bézier curves.

The GCD problem is ill-conditioned, in that small perturbations in the input polynomials give rise to large changes in the output GCD. Typically, if two exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ have an exact GCD $\hat{d}(x)$, then it is highly likely that their perturbed forms $f(x) = \hat{f}(x) + \delta\hat{f}(x)$ and $g(x) = \hat{g}(x) + \delta\hat{g}(x)$ are coprime.

The GCD is defined for exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ only, and it is therefore necessary to define an AGCD of two inexact polynomials. Methods for the computation of exact GCDs which fail to compute the AGCD for inexact polynomials do not suit the requirements of the inexact problem stated in this thesis. It is therefore necessary to instead consider methods which can compute the AGCD of two polynomials.

Univariate Polynomial GCD Computation by Euclid's Algorithm

The Euclidean algorithm for integer GCD computation can be extended to the computation of the GCD of two univariate polynomials [41]. Euclidean division is replaced by polynomial long division and the remainder sequence in the Euclidean algorithm is replaced by the polynomial remainder sequence (PRS).

Example 1.8.1. Consider the exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ given by

$$\begin{aligned}\hat{f}(x) &= (x-1)(x-2)(x-3)(x-4)(x-9)\\ &= x^5 - 19x^4 + 125x^3 - 365x^2 + 474x - 216\\ \hat{g}(x) &= (x-1)(x-2)(x-3)(x-5)(x-12)\\ &= x^5 - 23x^4 + 173x^3 - 553x^2 + 762x - 360.\end{aligned}$$

The first polynomial long division in the Euclidean algorithm is given by

$$\hat{f}(x) = p_0(x)\hat{g}(x) + r_0(x),$$

where

$$\begin{aligned}\hat{f}(x) &= (1 \times \hat{g}(x)) + x^5 - 23x^4 + 173x^3 - 553x^2 + 762x - 360\\ \hat{p}_0(x) &= 1\\ \hat{r}_0(x) &= x^5 - 23x^4 + 173x^3 - 553x^2 + 762x - 360.\end{aligned}$$

The second and third polynomial long divisions are completed in the same way as the first and are omitted from this text. The remainder $\hat{r}_2(x)$ is zero, and thus the algorithm terminates and the GCD is given by $\hat{r}_1(x)$, that is, $\hat{d}(x)$ is given by

$$\hat{d}(x) = -6x^3 + 36x^2 - 66x + 36.$$

□

The extended Euclidean algorithm [14, Section 4.2] also returns the cofactor polynomials $\hat{u}(x)$ and $\hat{v}(x)$ such that $\hat{f}(x)\hat{u}(x)+\hat{g}(x)\hat{v}(x) = \hat{d}(x)$. Despite the decreasing degree of

the polynomials in the PRS, the coefficients of these polynomials are often large. Instead, it is advantageous to consider the psuedo-PRS, which is generated by replacing polynomial division with psuedo-division.

Repeated polynomial division in Euclid's algorithm is equivalent to solving a linear system whose coefficient matrix is ill-conditioned, and thus small perturbations in the coefficients are magnified by the ill-conditioned problem. In [50] Noda and Sasaki note the limitations of the Euclidean algorithm, namely that close roots and floating-point numbers reduce the accuracy of coefficients in the PRS. Because of this, they developed a modified algorithm in which regularization is applied [50], with improved results.

1.8.1 Polynomial GCD Computation Using the Sylvester Matrix

Many GCD finding methods make use of the singular value decomposition (SVD) of the Sylvester matrix or its sequence of subresultant matrices [15,21], while others consider the QR decomposition [16,77] and methods which use the structure of the Sylvester matrix in the computation of the AGCD [5,6,44,79]. The Sylvester matrix and subresultant matrices are briefly defined for two polynomials in the power basis, and this is later extended to two polynomials in Bernstein form.

The Sylvester matrix, $S(\hat{f}(x), \hat{g}(x)) \in \mathbb{R}^{m+n+2\times m+n}$, of two polynomials $\hat{f}(x)$ and $\hat{g}(x)$ of degrees m and n respectively, contains entries which are coefficients of the two polynomials. The partitioned structure consists of two Toeplitz matrices $T_n(\hat{f}(x)) \in \mathbb{R}^{(m+n)\times n}$ and $T_m(\hat{g}(x)) \in \mathbb{R}^{(m+n)\times m}$, and is given by

$$
\begin{aligned}
S\left(\hat{f}(x), \hat{g}(x)\right) &= \left[\begin{array}{c|c} T_n\left(\hat{f}(x)\right) & T_m\left(\hat{g}(x)\right) \end{array}\right] \\
&= \left[\begin{array}{ccc|ccc}
\hat{a}_0 & & & \hat{b}_0 & & \\
\vdots & \ddots & & \vdots & \ddots & \\
\hat{a}_m & & \hat{a}_0 & \hat{b}_n & & \hat{b}_0 \\
& \ddots & \vdots & & \ddots & \vdots \\
& & \hat{a}_m & & & \hat{b}_n
\end{array}\right]
\end{aligned}
$$

The determinant of the Sylvester matrix is zero if the polynomials $\hat{f}(x)$ and $\hat{g}(x)$ have a common root.

The sequence of subresultant matrices $\{S_k(\hat{f}(x), \hat{g}(x)) \mid k = 1, \ldots, \min(m,n)\}$ are obtained by the removal of a subset of rows and columns from the Sylvester matrix, $S(\hat{f}(x), \hat{g}(x)) = S_1(\hat{f}(x), \hat{g}(x))$. Each subresultant matrix $S_k(\hat{f}(x), \hat{g}(x)) \in \mathbb{R}^{(m+n-k+1)\times(m+n-2k+2)}$ is given by removing the last $(k-1)$ rows and the $(k-1)$ rightmost columns from each of the two partitions $T_n(\hat{f}(x))$ and $T_m(\hat{g}(x))$ of $S_1(\hat{f}(x), \hat{g}(x))$. The kth subresultant matrix is therefore given by

$$
S_k\left(\hat{f}(x), \hat{g}(x)\right) = \left[\begin{array}{c|c} T_{n-k}\left(\hat{f}(x)\right) & T_{m-k}\left(\hat{g}(x)\right) \end{array}\right] \quad \text{for} \quad k = 1, \ldots, \min(m,n),
$$

where $T_{n-k}(\hat{f}(x)) \in \mathbb{R}^{(m+n-k+1)\times(n-k+1)}$ and $T_{m-k}(\hat{g}(x)) \in \mathbb{R}^{(m+n-k+1)\times(m-k+1)}$ are Toeplitz matrices. Two polynomials have a common divisor of degree k if the kth subresultant matrix has a zero determinant, therefore the sequence of subresultant matrices can be used to determine the degree of the GCD.

Singular Value Decomposition of the Sylvester Matrix

Many methods have been devised which make use of the SVD of the Sylvester matrix $S(\hat{f}(x), \hat{g}(x))$. The SVD is useful in the determination of the numerical rank of a matrix. In this thesis a matrix is described as being numerically rank deficient if it is sufficiently close to a matrix which is exactly rank deficient, and this is typically indicated by a large separation between two distinct subsets of singular values obtained by the SVD. This is a purely heuristic definition.

The discussion found in [36] is sufficient to describe the appeal of the SVD, where it is stated that "rounding errors and fuzzy data make rank determination a non-trivial exercise". Definitions of the SVD are frequently found throughout the literature. Here it is defined in the context of the Sylvester matrix.

Let $\hat{f}(x)$ and $\hat{g}(x)$ be polynomials of degree m and n with a common divisor of degree t, then the Sylvester matrix $S(\hat{f}(x), \hat{g}(x)) \in \mathbb{R}^{(m+n)\times(m+n)}$ has the SVD $S = U\Sigma V^T$, where columns of U given by $\mathbf{u}_1, \ldots, \mathbf{u}_{m+n}$ are the left singular vectors, columns of V given by $\mathbf{v}_1, \ldots, \mathbf{v}_{m+n}$ are the right singular vectors, and the diagonal matrix Σ contains the ordered singular values of $S(\hat{f}(x), \hat{g}(x))$

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{m+n-t-1} \geq \sigma_{m+n-t} = \sigma_{m+n-t+1} = \cdots = \sigma_{m+n} = 0.$$

However, in a floating-point environment, matrices can rarely be defined in terms of their rank since it is highly unlikely that they have singular values which are exactly equal to zero. Instead, it is necessary to consider the numerical rank.

Again, the numerical rank is defined in terms of the Sylvester matrix. The Sylvester matrix is said to have numerical rank $(m + n - t)$ if

$$\sigma_1 \geq \cdots \geq \sigma_{m+n-t-1} \geq \mu \left\| S(\hat{f}(x), \hat{g}(x)) \right\|_2 \geq \sigma_{m+n-t} \geq \sigma_{m+n-t+1} \geq \cdots \geq \sigma_{m+n},$$

where μ is the machine precision. However, this assumes the matrix $S(\hat{f}(x), \hat{g}(x))$ is exactly defined. Suppose the Sylvester matrix $S(f(x), g(x))$ of inexact polynomials is instead considered, then the ϵ-rank is considered. However, this is dependent on knowledge of the size of the errors in the entries of $S(f(x), g(x))$.

The Sylvester matrix of exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ in row echelon form reveals the GCD to within a constant factor [42]. This theorem is used by Corless *et al.* [15], where the SVD of a Sylvester matrix is used to compute the degree of the AGCD of two polynomials. Although it is numerically stable, the SVD is computationally inefficient. Their method computes the GCD of $\hat{f}(x)$ and $\hat{g}(x)$ of degree m and n by determining the singular values of the Sylvester matrix $S(\hat{f}(x), \hat{g}(x))$, and the numerical rank is used to determine the degree of the AGCD.

This differs from the work in this thesis, wherein the singular values of the Sylvester matrix and a set of subresultant matrices $\{S_k(\hat{f}(x), \hat{g}(x))\}$ are considered in the determination of the degree of the GCD. The process of computing a sequence of matrices is more costly, but often provides better results than the method described above. Several examples will show that there is often no separation between the numerically zero and non-zero singular values of $S_1(\hat{f}(x), \hat{g}(x))$, particularly in the presence of noise.

Elias and Zitko [20] similarly determine the degree of the AGCD by computing the set of minimum singular values $\{\dot{\sigma}_i \mid i = 1, \ldots, \min(m,n)\}$ and the corresponding singular vectors $\{\mathbf{v}_i \mid i = 1, \ldots, \min(m,n)\}$ for every subresultant matrix in the set $\{ S_k(\hat{f}(x), \hat{g}(x)) \mid k = 1, \ldots, \min(m,n) \}$. The degree t of the AGCD is given by the index of the largest entry in the set of $\{ \dot{\sigma}_i \mid i = 1, \ldots, min(m,n) \}$ such that $\sigma_i \leq \theta$ for some predefined threshold value θ. The cofactor polynomials $u(x) = \frac{f(x)}{d(x)}$ and $v(x) = \frac{g(x)}{d(x)}$ are then extracted from the vector $\mathbf{v}_t$.

This method is dependent on the determination of some threshold value θ, but the work described in this thesis removes the necessity for threshold determination. Instead, this thesis considers the point of maximal change in a set of values $\{ \rho_k \mid k = 1, \ldots, \min(m,n) \}$ to be indicative of the GCD degree, where ρ_k is a measure of rank of the matrix and may be obtained by SVD or QR decomposition.

Rank Revealing QR Decomposition

The rank of a matrix can also be determined by a rank revealing QR decomposition. Each subresultant matrix $S_{k+1}(\hat{f}(x), \hat{g}(x))$ in the sequence of subresultant matrices (where $\hat{f}(x)$ and $\hat{g}(x)$ are in the power basis) can be constructed by row and column removals from the previous matrix $S_k(\hat{f}(x), \hat{g}(x))$.

The matrix Q is orthogonal so the rank of the subresultant matrix S_k is equal to the rank of matrix R. Since R is upper triangular, its diagonal entries can be used in place of the set of singular values to determine the numerical rank of S_k. Good numerical results were shown in [9].

The QR decomposition, unlike the SVD, can be updated when rows and columns are removed, with a quadratic rather than cubic cost. However, this thesis deals with polynomials in the Bernstein form where the construction of the subresultant matrix sequence requires row and column multiplication as well as row and column removal. The QR update function can therefore no longer be applied.

A method of determining the coefficients of a GCD by QR decomposition of a matrix denoted S_4^* (a modified form of the Sylvester matrix) is given in [77]. A similar method is used by Corless et al. in their `QRGCD` method implemented in the `SNAP` package of Maple [16].

1.8.2 The Bézoutian Matrix

The Bézoutian matrix is used in the computation of the GCD of two polynomials. If $\hat{f}(x)$ and $\hat{g}(x)$ of degrees m and n respectively have a common root at y then

$$\frac{\hat{f}(x)\hat{g}(y) - \hat{f}(y)\hat{g}(x)}{(x-y)} = 0.$$

The Bézoutian matrix of $\hat{f}$ and $\hat{g}$ defined in the power basis is given by

$$B(\hat{f},\hat{g}) = \begin{bmatrix} c_{0,0} & c_{0,1} & \dots & c_{0,n-1} \\ c_{1,0} & c_{1,1} & \dots & c_{1,n} \\ \vdots & \vdots & & \vdots \\ c_{n,0} & c_{n,1} & \dots & c_{n,n} \end{bmatrix}, \tag{1.10}$$

where $\{c_{i,j}\}$ are the $(m \times m)$ coefficients of the bivariate polynomial $\hat{h}(x,y)$ of degree $(m-1, m-1)$

$$\hat{h}(x,y) = \sum_{i_2=0}^{m-1}\sum_{i_1=0}^{m-1} c_{i_1,i_2} x^{i_1} y^{i_2} = \frac{\hat{f}(x)\hat{g}(y) - \hat{f}(y)\hat{g}(x)}{x-y}.$$

The determinant of the Bézoutian matrix is the resultant of the two polynomials $\hat{f}$ and $\hat{g}$, which is zero if and only if they have a common divisor. The degree of the GCD of two polynomials $\hat{f}(x)$ and $\hat{g}(x)$ is therefore given by

$$t = \max(m,n) - \operatorname{rank}\left(B(\hat{f},\hat{g})\right),$$

that is, t is given by the rank loss of $B(\hat{f},\hat{g})$.

The coefficients of the common divisor are given by the last non-zero row of entries of the Bézoutian matrix in row echelon form. The extension of the Bézoutian matrix for the computation of the GCD of two polynomials in Bernstein form is considered in [7,8] and a Bézoutian matrix preprocessing operation is described in [74, 75]. The method described in [74] is used as a comparison for the methods developed in this thesis.

1.8.3 The GCD of Two Polynomials in Bernstein Form

The computation of the GCD of two polynomials is a well established problem, with several different approaches. However, much less work exists in relation to the computation of the GCD of two univariate polynomials in Bernstein form.

In [7,8] the Bernstein-Bézoutian matrix is used to compute the GCD of two polynomials in Bernstein form, with applications in computing the intersections of rational curves. The Sylvester matrix is also adapted for use in computing the GCD of two polynomials in Bernstein form [9, 10, 74].

Given an arbitrary polynomial, the roots will lie in the complex plane. In the Bernstein basis only the roots in the unit interval are of interest, but a polynomial that arises from an intersection problem will, in general, have roots in the complex plane. Outside thisinterval, the Bernstein basis has no advantages with respect to the power basis. Examples in this thesis will consist of GCD and root finding problems where the specified polynomials roots both inside and outside the unit interval.

1.8.4 Bivariate Polynomial GCDs

A principal ideal domain is defined as an integral domain where every proper ideal can be generated by a single element. While the univariate polynomials form a Euclidean

domain, the bivariate polynomials do not, since the ideal (x, y) is not principal. However the bivariate polynomials do form a unique factorisation domain, and the GCD is still defined.

The computation of the GCD of multivariate polynomials has received much less attention than the univariate problem. Brown's method for computing the GCD of two bivariate polynomials with integer coefficients in [12] reduces the multivariate problem to a simpler domain. The multivariate GCD algorithm requires a set of modular homomorphisms and evaluation homomorphisms. An evaluation homomorphism reduces a polynomial in $\mathbb{Z}[x_1, \ldots, x_k]$ to a problem in $\mathbb{Z}[x_1]$ by evaluating the polynomial at a set of points for all other variables $[x_2, \ldots, x_k]$. The Chinese remainder theorem is used to determine the GCD in the original problem domain. Extensions of this are the sparse GCD method due to Zippel [81, 82] and the EZGCD algorithm developed by Moses and Yun [48].

This thesis is mostly concerned with structured matrix based methods. The singular value decomposition of the Sylvester matrix is used in the computation of the degree of the GCD of two bivariate polynomials [15, 79, 80]. However, an extension of the Sylvester matrix for use in the computation of the GCD of two bivariate polynomials in Bernstein form does not seem to appear in the literature. Therefore, this form of Sylvester matrix will be defined in the later chapters of this thesis.

1.9 Conclusion

This chapter has discussed a set of curve and surface intersection problems and a variety of methods for their solution. Bézier curve-curve and surface-surface intersections were shown to reduce to (i) the computation of the roots of a univariate polynomial or (ii) the factorisation of a bivariate polynomial where polynomials are defined in Bernstein form.

The methods of Gauss and Musser were considered for the polynomial square-free factorisation problem, and given the square-free factors of a polynomial, the multiple roots are more easily obtained. These methods have the benefit of preserving the multiplicity structure of the polynomial roots, and this structure is given by a sequence of GCD computations. Two main components of Gauss' and Musser's algorithms are the set of polynomial GCD computations and the structured set of polynomial deconvolutions. Structured matrix based methods are used to solve both problems in the following chapters.

The two-polynomial GCD finding problem has been considered, and a significant portion of this chapter was focused on methods using the Sylvester matrix. The UGCD method defined in the following chapters will make use of the same structure for the computation of the GCD of polynomials in Bernstein form.

In the next chapter the Bézier curve and surface patches are defined and some properties are explored. The relationship between Bernstein polynomial multiplication and convolution matrices is described such that the computation of polynomial GCDs by matrix methods is more easily defined in Chapter 3.

Chapter 2

Curves, Surfaces and Polynomial Representations

The Bézier curve and Bézier surface patches have been briefly introduced as means of representing curves and surfaces in CAGD. This section defines the Bézier curve and Bézier surface representations as well as univariate and bivariate polynomials in Bernstein form. Some basic arithmetic will be discussed, focusing on univariate and bivariate polynomial multiplication by matrix based methods. This gives insight into the development of the Sylvester matrix and subresultant matrices described in the Chapter 3.

Section 2.1 In this section the Bézier curves, triangular Bézier surface patches and rectangular Bézier surface patches are defined and some properties of these curves and surfaces are given. Of more interest to this work are the univariate and bivariate Bernstein polynomials which define these curves and surfaces.

Section 2.2 The univariate polynomial in Bernstein form and the two bivariate extensions required for the representation of the two forms of the Bézier patch are defined. The representation of polynomials in matrix and vector form is considered and a method is described in which a polynomial multiplication, $h = fg$, can be written as a matrix vector product, where the matrix and vector contain the coefficients of f and g respectively. Multiplication in Bernstein form is somewhat more involved than the equivalent operation for polynomials in the power basis, particularly when considering the multiplication of bivariate polynomials.

2.1 Bézier Curves and Surfaces

2.1.1 The Bézier Curve

The planar Bézier curve $\mathbf{B}(t)$ of degree m is a parametrically defined curve which is uniquely determined by a set of $(m+1)$ coordinate pairs, $\{P_i = (x_i, y_i) \mid i = 0, \ldots, m\}$. These control points define the shape of the curve which starts and ends at P_0 and P_m respectively. The intermediate control points act as a pull on the direction of the curve, and these points may or may not be weighted to strengthen or relax this pull. In CAGD software the control points of a curve or surface can easily be manipulated by a designer to intuitively alter the shape.

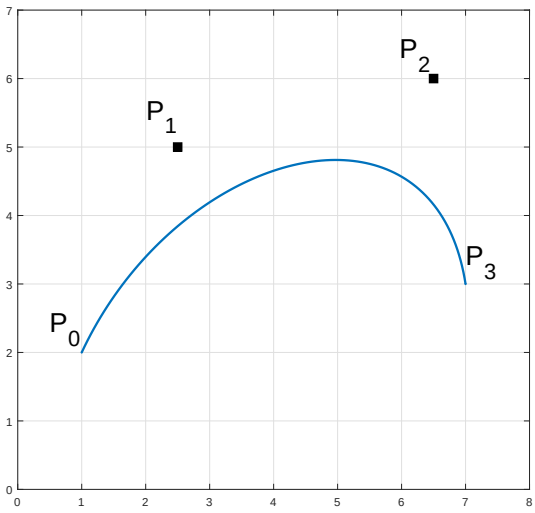


Figure 2.1: The cubic Bézier curve with control points $P_0 = (1, 2)$, $P_1 = (2.5, 5)$, $P_2 = (6.5, 6)$ and $P_3 = (7, 3)$

A planar Bézier curve of degree m is defined by

$$\mathbf{B}(t) = \sum_{i=0}^{m} P_i B_i^m(t), \tag{2.1}$$

where the set of $(m+1)$ basis elements $\{ B_i^m(t) \mid i = 0, \ldots, m\}$ are given by

$$B_i^m(t) = \binom{m}{i} (1-t)^{m-i}\, t^i \quad \text{where} \quad \sum_{i=0}^{m} B_i^m(t) = 1. \tag{2.2}$$

The curve is defined parametrically and the defining functions $x(t)$ and $y(t)$ are given as

$$x(t) = \sum_{i=0}^{m} x_i B_i^m(t) \qquad \text{and} \quad y(t) = \sum_{i=0}^{m} y_i B_i^m(t).$$

The non-planar Bézier curve of degree m is defined in a similar way. Its control points P_i are coordinates in three-dimensional space, and it is defined by three parametric functions $x(t)$, $y(t)$ and $z(t)$.

Bézier curves used in CAGD are typically limited to degree three curves, since a cubic curve is flexible enough to allow the construction of most simple shapes. More intricately shaped curves can be constructed by stitching cubic Bézier curves together in the form of splines. A generalization of the Bézier curve is the rational Bézier curve, which can represent a wider variety of curves, including conics, through the introduction of weights w_i. The rational Bézier curve is given by

$$\mathbf{B}(t) = \frac{\sum_{i=0}^{m} w_i P_i B_i^m(t)}{\sum_{i=0}^{m} w_i B_i^m(t)}.$$

It was mentioned in Section 1.2 that a parametric curve as described in (2.1) is rendered by evaluation at a set of values t_i. When rendering a Bézier curve it can be more computationally efficient to utilise the de Casteljau algorithm (Appendix A.3) for generating a set of points which have the advantage of being evenly distributed along the curve.

Properties of Bézier Curves

The properties of Bézier curves are well documented [23, Section 4.2] and the most important of these properties are now outlined:

End Point Interpolation : A Bézier curve with $(m+1)$ control points passes through the points P_0 and P_m.

Convex Hull Property : The entire Bézier curve is contained within its convex hull. Methods for computing the convex hull can be found in [51, Chapter 3], and the convex hulls of two curves can be used to compute their intersection.

Degree Elevation : A Bézier curve of degree m can be degree elevated to a curve of degree $(m+r)$ for $r \in \mathbb{Z}^+$.

2.1.2 The Rectangular Tensor-Product Bézier Surface Patch

The rectangular Bézier surface patch whose degree with respect to s and t is given by m_1 and m_2 is given by

$$\mathbf{B}(s,t) = \sum_{i_2=0}^{m_2} \sum_{i_1=0}^{m_1} P_{i_1,i_2} B_{i_1}^{m_1}(s) B_{i_2}^{m_2}(t),$$

where

$$B_{i_1}^{m_1}(s) = \binom{m_1}{i_1}(1-s)^{m_1-i_1}s^{i_1} \quad \text{and} \quad B_{i_2}^{m_2}(t) = \binom{m_2}{i_2}(1-t)^{m_2-i_2}t^{i_2}$$

are the univariate Bernstein basis functions described in (2.2).

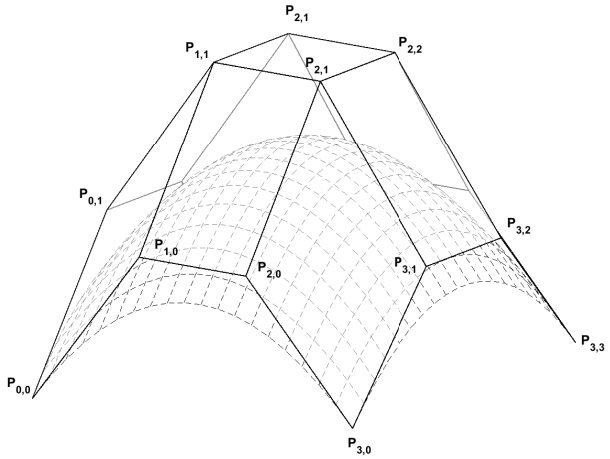


Figure 2.2: A bicubic Bézier surface patch

The surface is defined by the set of $(m_1+1)\times(m_2+1)$ control points which form a control mesh for the surface. The points P_{i_1,i_2} are the $(m_1+1)\times(m_2+1)$ three-dimensional

control points given by

$$\mathbf{P}_{i_1,i_2} = \left[\; x_{i_1,i_2}, \quad y_{i_1,i_2}, \quad z_{i_1,i_2} \; \right]^T \quad \text{for} \quad i_1 = 0, \ldots m_1; \; i_2 = 0, \ldots, m_2.$$

In CAGD the rectangular Bézier surface patch is typically limited to a bicubic patch, where $m_1 = 3$ and $m_2 = 3$ (as shown in Figure 2.2) and the control points of a bicubic Bézier patch are given by a two-dimensional array of vectors. Similar to Bézier curves, bicubic Bézier patches are often stitched together to form more complex shapes and objects.

2.1.3 The Triangular Bézier Surface Patch

An overview of triangular Bézier patches is now given, with its defining basis polynomials being considered in the next section. A more detailed study of triangular Bézier patches is given by Farin in [22] and more recently in [23].

The triangular Bézier patch of total degree m is given by the set of $\binom{m+2}{2} = \frac{(m+1)(m+2)}{2}$ control points P_{i_1,i_2} which form a triangular control net. A point on the surface is given by the function

$$S(s,t) = \sum_{i_1+i_2=0}^{m} P_{i_1,i_2} B^m_{i_1,i_2}(s,t),$$

where $B^m_{i_1,i_2}(s,t)$ are the basis elements given by

$$B^m_{i_1,i_2}(s,t) = \binom{m}{i_1, i_2} (1-s-t)^{m-i_1-i_2} s^{i_1} t^{i_2} \qquad \text{for} \quad 0 \leq i_1 + i_2 \leq m$$

and the trinomial coefficients

$$\binom{m}{i_1, i_2} = \frac{m}{i_1! i_2! (m-i_1-i_2)!} \tag{2.3}$$

generate Pascal's triangle.

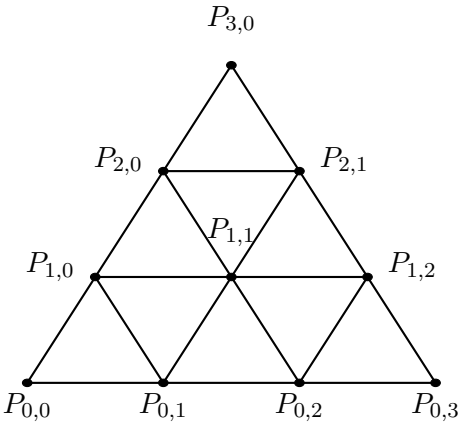


Figure 2.3: Control points of a cubic triangular Bézier patch

The three edges of the control net, E_1, E_2 and E_3, are defined by a subset of control points, where E_1 is defined by the set of points $\{ P_{0,i_2} \mid i_2 = 0, \ldots, m \}$, E_2 is defined by the points $\{ P_{i_1,0} \mid i_1 = 0, \ldots, m \}$ and E_3 is defined by the points $\{ P_{i_1,i_2} \mid i_1 + i_2 = m \}$.

All three of these edges are Bézier curves of degree m.

2.2 The Bernstein Polynomial Representation

Section 2.1 introduced the Bézier curve, the rectangular Bézier surface patch and the triangular Bézier surface patch. These curves and surfaces are parametric representations and their defining parametric equations are polynomials in Bernstein form. Basic arithmetic of univariate polynomials in Bernstein form can be found in [29] and extensions to bivariate polynomials are found in [4].

This section introduces the Bernstein polynomial representation for univariate and bivariate polynomials. The bivariate polynomial in Bernstein form has two distinct forms, which are described in Sections 2.2.2 and 2.2.3 respectively.

2.2.1 The Univariate Polynomial in Bernstein Form

In Section 2.1.1 the Bézier curve was introduced, and was defined by a set of control points whose coordinates are the coefficients of two univariate polynomials in Bernstein form. A univariate polynomial in Bernstein form $\hat{f}(x)$ of degree m is given by

$$\hat{f}(x) = \sum_{i=0}^{m} \hat{a}_i B_i^m(x) = \sum_{i=0}^{m} \hat{a}_i \binom{m}{i} (1-x)^{m-i} x^i. \tag{2.4}$$

The polynomial $\hat{f}(x)$ is said to be in scaled Bernstein form when the binomial coefficient $\binom{m}{i}$ is included with the polynomial coefficient $\hat{a}_i$, that is, $\hat{f}(x)$ in scaled Bernstein form is given by

$$\hat{f}(x) = \sum_{i=0}^{m} \left[\hat{a}_i \binom{m}{i} \right] \left((1-x)^{m-i} x^i \right). \tag{2.5}$$

Properties of the Bernstein Basis Functions

Properties of the Bernstein basis functions are described by Farouki in [27], and as with the properties of the Bézier curve, a subset of key properties associated with this thesis are now described:

Non-Negative : The Bernstein basis functions $B_i^m(t)$ are non-negative over the interval $[0,1]$, that is, $B_i^m(t) \geq 0$ for $i = 0, \ldots, m$, $t \in [0,1]$.

Partition of Unity : For any value t in the interval $[0,1]$, the sum of the set of Bernstein basis functions $B_i^m(t)$ for $i = 0, \ldots, m$ is equal to one, that is,

$$\sum_{i=0}^{m} B_i^m(t) = 1 \quad \text{for} \quad t \in [0,1].$$

Symmetry : The Bernstein basis functions are symmetric, that is,

$$B_i^m(t) = B_{m-i}^m(1-t) \qquad t \in [0,1].$$

Degree Elevation : In Chapter 7 the degree elevation of univariate and bivariate polynomials will be considered in the computation of the degree of a bivariate GCD. A simple method of degree elevation is described in Appendix A.1.

Vector Representation and Polynomial Multiplication

The coefficients of a polynomial $\hat{f}(x)$ of degree m can be arranged as a coefficient vector $\hat{\mathbf{f}} \in \mathbb{R}^{m+1}$. In this thesis the coefficients of a polynomial are ordered in ascending power, i.e. the coefficient $\hat{a}_i$ corresponding to basis element $B_i^m(x)$ precedes $\hat{a}_{i+1}$ in the coefficient vector. The vector of coefficients of the polynomial $\hat{f}(x)$ as described in (2.4) is given by

$$\hat{\mathbf{f}} = \begin{bmatrix} \hat{a}_0, & \hat{a}_1, & \ldots, & \hat{a}_m \end{bmatrix}^T \tag{2.6}$$

and the vector of coefficients of the polynomial $\hat{f}(x)$ in scaled Bernstein form in (2.5) is given by

$$\hat{\mathbf{f}}_{bi} = \begin{bmatrix} \hat{a}_0\binom{m}{0}, & \hat{a}_1\binom{m}{1}, & \ldots, & \hat{a}_m\binom{m}{m} \end{bmatrix}^T.$$

Algorithms for arithmetic operations on polynomials in the Bernstein basis are found in [29]. The work in this thesis requires the multiplication of polynomials in Bernstein form, and the method required is quite different from that required for the multiplication of polynomials in the power basis.

The product of two polynomials $\hat{f}(x)$ and $\hat{g}(x)$ of degrees m and n respectively with basis elements $\{ B_i^m(x) \mid i = 0, \ldots, m \}$ and $\{ B_i^n(x) \mid i = 0, \ldots, n \}$ is a polynomial $\hat{h}(x)$ of degree $(m+n)$ with basis elements $\{ B_i^{m+n}(x) \mid i = 0, \ldots, m+n \}$. The polynomial $\hat{f}(x)$ is defined in (2.4) and $\hat{g}(x)$ is given by

$$\hat{g}(x) = \sum_{i=0}^{n} \hat{b}_i \binom{n}{i} (1-x)^{n-i} x^i.$$

The product $\hat{h}(x)$ in terms of the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$ is given by

$$\hat{h}(x) = \sum_{j=0}^{n} \sum_{i=0}^{m} \hat{a}_i \hat{b}_j \binom{m}{i} \binom{n}{j} (1-x)^{m+n-i-j} x^{i+j}, \tag{2.7}$$

which is 'almost' in Bernstein form, and multiplying each term in the summation in (2.7) by $\frac{\binom{m+n}{i+j}}{\binom{m+n}{i+j}}$ yields a polynomial in Bernstein form given by

$$\begin{aligned} \hat{h}(x) &= \sum_{i=0}^{m} \sum_{j=0}^{n} \hat{a}_i \hat{b}_j \frac{\binom{m+n}{i+j}}{\binom{m+n}{i+j}} \binom{m}{i} \binom{n}{j} (1-x)^{m+n-i-j} x^{i+j} \\ &= \sum_{i=0}^{m} \sum_{j=0}^{n} \hat{a}_i \hat{b}_j \frac{\binom{m}{i}\binom{n}{j}}{\binom{m+n}{i+j}} B_{i+j}^{m+n}(x). \end{aligned}$$

The coefficients of $\hat{h}(x)$ are computed by the matrix-vector multiplication

$$\begin{aligned}\hat{\mathbf{h}} &= C_n\left(\hat{f}(x)\right)\hat{\mathbf{g}} \\ \hat{\mathbf{h}} &= \left(D_{m+n}^{-1}T_n\left(\hat{f}(x)\right)Q_n\right)\hat{\mathbf{g}},\end{aligned} \tag{2.8}$$

where $C_n(\hat{f}(x))$ is the nth order convolution matrix of the polynomial $\hat{f}(x)$ in Bernstein form given by

$$C_n\left(\hat{f}(x)\right) = \left(D_{m+n}^{-1}T_n\left(\hat{f}(x)\right)Q_n\right). \tag{2.9}$$

This convolution matrix is the product of three matrices D_{m+n}^{-1}, $T_n(\hat{f}(x))$ and Q_n, where the diagonal matrix $D_{m+n}^{-1} \in \mathbb{R}^{(m+n+1)\times(m+n+1)}$ is given by

$$D_{m+n}^{-1} = \operatorname{diag}\left[\ \tfrac{1}{\binom{m+n}{0}},\quad \tfrac{1}{\binom{m+n}{1}},\quad \ldots,\quad \tfrac{1}{\binom{m+n}{m+n}}\ \right], \tag{2.10}$$

the Toeplitz matrix $T_n(\hat{f}(x)) \in \mathbb{R}^{(m+n+1)\times(n+1)}$ consists of the coefficients of the polynomial $\hat{f}(x)$ in scaled Bernstein form as defined in (2.5) and is given by

$$T_n\left(\hat{f}(x)\right) = \begin{bmatrix} \hat{a}_0\binom{m}{0} & & & \\ \hat{a}_1\binom{m}{1} & \hat{a}_0\binom{m}{0} & & \\ \vdots & \hat{a}_1\binom{m}{1} & \ddots & \\ \vdots & \vdots & \ddots & \hat{a}_0\binom{m}{0} \\ \hat{a}_m\binom{m}{m} & \vdots & & \hat{a}_1\binom{m}{1} \\ & \hat{a}_m\binom{m}{m} & & \vdots \\ & & \ddots & \vdots \\ & & & \hat{a}_m\binom{m}{m} \end{bmatrix}, \tag{2.11}$$

and the diagonal matrix $Q_n \in \mathbb{R}^{(n+1)\times(n+1)}$ is given by

$$Q_n = \operatorname{diag}\left[\ \tbinom{n}{0},\quad \tbinom{n}{1},\quad \ldots,\quad \tbinom{n}{n}\ \right]. \tag{2.12}$$

The vector $\hat{\mathbf{g}} \in \mathbb{R}^{n+1}$ in (2.8) is a column vector containing the coefficients of the polynomial $\hat{g}(x)$ and is given by

$$\hat{\mathbf{g}} = \left[\ \hat{b}_0,\quad \hat{b}_1,\quad \ldots,\quad \hat{b}_n\ \right]^T \in \mathbb{R}^{n+1}.$$

Note that the product of the diagonal matrix Q_n and the vector $\hat{\mathbf{g}}$ is equal to a vector of coefficients in scaled Bernstein form as described in Section 2.2.1, but in this thesis the binomial coefficients are deliberately separated into the matrix Q_n.

The multiplication of two polynomials is commutative, that is, $\hat{f}(x)\hat{g}(x) = \hat{h}(x)$ and $\hat{g}(x)\hat{f}(x) = \hat{h}(x)$, so the matrix-vector product can be written as either

$$D_{m+n}^{-1}T_n\left(\hat{f}(x)\right)Q_n\hat{\mathbf{g}} = \hat{\mathbf{h}} \quad \text{or} \quad D_{m+n}^{-1}T_m\Big(\hat{g}(x)\Big)Q_m\hat{\mathbf{f}} = \hat{\mathbf{h}},$$

where $T_m\left(\hat{g}(x)\right) \in \mathbb{R}^{(m+n+1)\times(m+1)}$ is a Toeplitz matrix of the coefficients of $\hat{g}(x)$ in scaled Bernstein form, similar to (2.11), and $Q_m \in \mathbb{R}^{(m+1)\times(m+1)}$ is similar to (2.12).

Example 2.2.1. Consider the polynomial $\hat{f}(x)$ of degree $m = 2$ which is given in Bernstein form as

$$\hat{f}(x) = 7B_0^2(x) + 9.5B_1^2(x) + 15B_2^2(x),$$

and the polynomial $\hat{g}(x)$ of degree $n = 1$ which is given in Bernstein from as

$$\hat{g}(x) = 1B_0^1(x) + 3B_1^1(x).$$

The product $\hat{h}(x)$ of degree $m + n = 3$ is given by the matrix-vector product

$$\begin{bmatrix} \frac{1}{\binom{3}{0}} & & & \\ & \frac{1}{\binom{3}{1}} & & \\ & & \frac{1}{\binom{3}{2}} & \\ & & & \frac{1}{\binom{3}{3}} \end{bmatrix} \begin{bmatrix} 7\binom{2}{0} & 0 \\ 9.5\binom{2}{1} & 7\binom{2}{0} \\ 15\binom{2}{2} & 9.5\binom{2}{1} \\ 0 & 15\binom{2}{2} \end{bmatrix} \begin{bmatrix} \binom{1}{0} & \\ & \binom{1}{1} \end{bmatrix} \begin{bmatrix} 1 \\ 3 \end{bmatrix} = \begin{bmatrix} 7 \\ 40/3 \\ 24 \\ 45 \end{bmatrix}.$$

The resulting vector contains the coefficients of the polynomial $\hat{h}(x)$ which is given by

$$\hat{h}(x) = 7B_0^3(x) + \frac{40}{3}B_1^3(x) + 24B_2^3(x) + 45B_3^3(x).$$

The multiplication of polynomials in Bernstein form is required in Section 3.1, where it will be shown that the Sylvester subresultant matrix $S_k(\hat{f}(x), \hat{g}(x))$ of two polynomials $\hat{f}(x)$ and $\hat{g}(x)$ is a matrix consisting of two convolution matrices. The Sylvester matrix and the sequence of subresultant matrices will be utilized in the computation of the degree and coefficients of the GCD of two polynomials.

Division

Polynomial division follows from its inverse operation multiplication, and is briefly described here. In the problems described in this thesis, it is assumed that the polynomials found in the division problems divide exactly with zero remainder, or, are inexact forms of exact polynomials which divide with zero remainder. Given two polynomials $\hat{f}(x)$ and $\hat{g}(x)$ of degrees m and n respectively, the equation $\frac{\hat{f}(x)}{\hat{g}(x)} = \hat{h}(x)$ can be written as $\hat{g}(x)\hat{h}(x) = \hat{f}(x)$, where $\hat{h}(x)$ is unknown. This can be written as the linear system

$$\begin{aligned} C_{m-n}\left(\hat{g}(x)\right)\hat{\mathbf{h}} &= \hat{\mathbf{f}} \\ \left(D_m^{-1}T_{m-n}\left(\hat{g}(x)\right)Q_{m-n}\right)\hat{\mathbf{h}} &= \hat{\mathbf{f}} \end{aligned}$$

which is solved for $\hat{\mathbf{h}}$ by simple least squares. As with univariate polynomial multiplication, the convolution matrix $C_{m-n}\left(\hat{g}(x)\right)$ is the product of three matrices and is given by $C_{m-n}\left(\hat{g}(x)\right) = D_m^{-1}T_{m-n}\left(\hat{g}(x)\right)Q_{m-n}$, where the component matrices D_m^{-1}, $T_{m-n}\left(\hat{g}(x)\right)$ and Q_{m-n} are of the same structure as the matrices defined in (2.10), (2.11) and (2.12) respectively.

2.2.2 The Bivariate Bernstein Polynomial over a Rectangular Domain

In Section 2.1.2 the rectangular Bézier patch was introduced, and the defining parametric equations for this patch are given by bivariate Bernstein polynomials over a rectangular domain. The bivariate polynomial $\hat{f}(x,y)$ in Bernstein form of degree (m_1, m_2) is given by

$$\begin{aligned}\hat{f}(x,y) &= \sum_{i_1=0}^{m_1}\sum_{i_2=0}^{m_2} \hat{a}_{i_1,i_2} B_{i_1}^{m_1}(x) B_{i_2}^{m_2}(y) \\ &= \sum_{i_1=0}^{m_1}\sum_{i_2=0}^{m_2} \hat{a}_{i_1,i_2} \binom{m_1}{i_1}\binom{m_2}{i_2} (1-x)^{m_1-i_1} x^{i_1} (1-y)^{m_2-i_2} y^{i_2}. \end{aligned} \tag{2.13}$$

The bracketed pair (m_1, m_2) describes the degree structure of a bivariate polynomial, where $\deg_x(\hat{f}(x,y)) = m_1$ and $\deg_y(\hat{f}(x,y)) = m_2$. This notation is used throughout the remainder of this thesis and can be extended to describe the degree structure of any n variate polynomial with the vector $\left[\begin{array}{cccc} m_1, & m_2, & \ldots, & m_n \end{array}\right]$ containing the degree of the polynomial with respect to each of the n variables.

As with the univariate polynomial, the bivariate polynomial can be expressed in scaled Bernstein form which is given by

$$\hat{f}(x,y) = \sum_{i_1=0}^{m_1}\sum_{i_2=0}^{m_2} \left[\hat{a}_{i_1,i_2}\binom{m_1}{i_1}\binom{m_2}{i_2}\right] (1-x)^{m_1-i_1} x^{i_1} (1-y)^{m_2-i_2} x^{i_2},$$

where the set $\{\, \hat{a}_{i_1,i_2}\binom{m_1}{i_1}\binom{m_2}{i_2} \mid i_1 = 0,\ldots,m_1;\, i_2 = 0,\ldots,m_2 \,\}$ contains the $(m_1+1)\times(m_2+1)$ coefficients of the polynomial in scaled Bernstein form. The polynomial $\hat{f}(x,y)$ can also be written as the product of a coefficient matrix and two vectors of basis elements and is given by

$$\hat{f}(x,y) = \left[\begin{array}{ccc} B_0^{m_1}(x), & \ldots, & B_{m_1}^{m_1}(x) \end{array}\right] \left[\begin{array}{ccc} \hat{a}_{0,0} & \ldots & \hat{a}_{0,m_2} \\ \vdots & \ddots & \vdots \\ \hat{a}_{m_1,0} & \ldots & \hat{a}_{m_1,m_2} \end{array}\right] \left[\begin{array}{c} B_0^{m_2}(y) \\ \vdots \\ B_{m_2}^{m_2}(y) \end{array}\right]. \tag{2.14}$$

Vector Representation and Multiplication by a Matrix Method

As with univariate polynomials, it is necessary to define a vector representation of the bivariate polynomial in Bernstein form. The polynomial $\hat{f}(x,y)$ can be considered as a polynomial in y whose coefficients are polynomials in x, that is, the (m_2+1) coefficients are given by the set $\{\, \hat{f}_j(x) \mid j = 0,\ldots,m_2 \,\}$ and $\hat{f}(x,y)$ is given by

$$\hat{f}(x,y) = \left(\hat{f}_0(x)\times B_0^{m_2}(y)\right) + \left(\hat{f}_1(x)\times B_1^{m_2}(y)\right) + \cdots + \left(\hat{f}_{m_2}(x)\times B_{m_2}^{m_2}(y)\right), \tag{2.15}$$

where each of the univariate polynomials in the set $\{\, \hat{f}_j(x) \mid j = 0,\ldots,m_2 \,\}$ is given by

$$\hat{f}_j(x) = \sum_{i=0}^{m_1} \hat{a}_{i,j} B_i^{m_1}(x), \quad j = 0,\ldots,m_2$$

and the coefficients of each polynomial $\hat{f}_j(x)$ form the $(j+1)$th column of the coefficient matrix in (2.14).

The coefficients of the bivariate Bernstein polynomial $\hat{f}(x,y)$ are arranged to form the vector $\hat{\mathbf{f}} \in \mathbb{R}^{(m_1+1)(m_2+1)\times 1}$ as follows:

$$\hat{\mathbf{f}} = \left[\begin{array}{cccc} \hat{\mathbf{f}}_0, & \hat{\mathbf{f}}_1, & \ldots, & \hat{\mathbf{f}}_{m_2} \end{array} \right]^T,$$

where each of the vectors $\hat{\mathbf{f}}_j \in \mathbb{R}^{m_1+1}$ for $j = 0, \ldots, m_2$ contain the coefficients of the polynomial $\hat{f}_j(x)$ and are given by

$$\hat{\mathbf{f}}_j = \left[\begin{array}{cccc} \hat{a}_{0,j}, & \hat{a}_{1,j}, & \ldots, & \hat{a}_{m_1,j} \end{array} \right]^T.$$

An alternative ordering of the coefficients is given by considering $\hat{f}(x,y)$ as a polynomial in the primary variable x whose (m_1+1) coefficients are polynomials in the secondary variable y, but for the remainder of this work it can be assumed that the first ordering is used.

Having defined a system for ordering the coefficients of the bivariate polynomial in Bernstein form, the multiplication of two bivariate polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ is now considered. Let the bivariate polynomial $\hat{f}(x,y)$ with degree structure (m_1, m_2) be as defined in (2.13) and let $\hat{g}(x,y)$ with degree structure (n_1, n_2) be given by

$$\hat{g}(x,y) = \sum_{i_2=0}^{n_2} \sum_{i_1=0}^{n_1} \hat{b}_{i,j} B_{i_1}^{n_1}(x) B_{i_2}^{n_2}(y).$$

The polynomial $\hat{g}(x,y)$ can be thought of as a polynomial in y whose (n_2+1) coefficients are each polynomials in x, each of degree n_1, such that $\hat{g}(x,y)$ is given by

$$\hat{g}(x,y) = \hat{g}_0(x) B_0^{n_2}(y) + \hat{g}_1(x) B_1^{n_2}(y) + \cdots + \hat{g}_{n_2}(x) B_{n_2}^{n_2}(y),$$

where

$$\hat{g}_j(x) = \sum_{i_1=0}^{n_1} \hat{b}_{i_1,j} B_{i_1}^{n_1}(x) \quad \text{for} \quad j = 0, \ldots, n_2.$$

The product of $\hat{f}(x,y)$ and $\hat{g}(x,y)$ is given by $\hat{h}(x,y)$

$$\hat{h}(x,y) = \sum_{i_2=0}^{m_2+n_2} \sum_{i_1=0}^{m_1+n_1} \hat{c}_{i,j} B_{i_1}^{m_1+n_1}(x) B_{i_2}^{m_2+n_2}(y).$$

The polynomial $\hat{h}(x,y)$ is considered as a polynomial in y of degree (m_2+n_2), where the set of coefficients $\{\, \hat{h}_i(x) \mid i = 0, \ldots, m_2+n_2 \,\}$ are polynomials in x, each of degree (m_1+n_1)

$$\hat{h}(x,y) = \hat{h}_0(x) B_0^{m_2+n_2}(y) + \hat{h}_1(x) B_1^{m_2+n_2}(y) + \cdots + \hat{h}_{m_2+n_2}(x) B_{m_2+n_2}^{m_2+n_2}(y).$$

Alternatively, expressed in terms of the set of polynomials $\{\, \hat{f}_i(x) \mid i = 0, \ldots, m_2 \,\}$ and

$\{\, \hat{g}_j(x) \mid j = 0, \ldots, n_2 \,\}$, $\hat{h}(x,y)$ is given by

$$\begin{aligned}\hat{h}(x,y) = \hat{f}_0(x)\hat{g}_0(x)B_0^{m_2}(y)B_0^{n_2}(y) &+ \left(\hat{f}_0(x)\hat{g}_1(x)B_0^{m_2}(y)B_1^{n_2}(y) + \hat{f}_1(x)\hat{g}_0(x)B_1^{m_2}(y)B_0^{n_2}(y)\right) \\ &+ \left(\hat{f}_2(x)\hat{g}_0(x)B_2^{m_2}(y)B_0^{n_2}(y) + \hat{f}_1(x)\hat{g}_1(x)B_1^{m_2}(y)B_1^{n_2}(y) + \hat{f}_0(x)\hat{g}_2(x)B_0^{m_2}(y)B_2^{n_2}(y)\right) \\ &+ \cdots + \left(\hat{f}_{m_2}(x)\hat{g}_{n_2}(x)B_{m_2}^{m_2}(y)B_{n_2}^{n_2}(y)\right) \qquad (2.16)\end{aligned}$$

and a general product from the summation (2.16) above is given by

$$\begin{aligned}\hat{f}_s(x)\hat{g}_t(x)B_s^{m_2}(y)B_t^{n_2}(y) &= \left(B_s^{m_2}(y)B_t^{n_2}(y)\right)\hat{f}_s(x)\hat{g}_t(x) \\ &= \frac{\binom{m_2}{s}\binom{n_2}{t}}{\binom{m_2+n_2}{s+t}} B_{s+t}^{m_2+n_2}(y)\hat{f}_s(x)\hat{g}_t(x).\end{aligned}$$

The expression (2.16) can be rewritten as

$$\begin{aligned}\hat{h}(x,y) = B_0^{m_2+n_2}(y) &\left(\hat{f}_0(x)\hat{g}_0(x)\frac{\binom{m_2}{0}\binom{n_2}{0}}{\binom{m_2+n_2}{0}}\right) \\ + B_1^{m_2+n_2}(y) &\left(\hat{f}_0(x)\hat{g}_1(x)\frac{\binom{m_2}{0}\binom{n_2}{1}}{\binom{m_2+n_2}{1}} + \hat{f}_1(x)\hat{g}_0(x)\frac{\binom{m_2}{1}\binom{n_2}{0}}{\binom{m_2+n_2}{1}}\right) \\ + B_2^{m_2+n_2}(y) &\left(\hat{f}_2(x)\hat{g}_0(x)\frac{\binom{m_2}{2}\binom{n_2}{0}}{\binom{m_2+n_2}{2}} + \hat{f}_1(x)\hat{g}_1(x)\frac{\binom{m_2}{1}\binom{n_2}{1}}{\binom{m_2+n_2}{2}} + \hat{f}_0(x)\hat{g}_2(x)\frac{\binom{m_2}{0}\binom{n_2}{2}}{\binom{m_2+n_2}{2}}\right) \\ + \cdots + B_{m_2+n_2}^{m_2+n_2}(y) &\left(\hat{f}_{m_2}(x)\hat{g}_{n_2}(x)\frac{\binom{m_2}{m_2}\binom{n_2}{n_2}}{\binom{m_2+n_2}{m_2+n_2}}\right), \qquad (2.17)\end{aligned}$$

which can be written as a matrix-vector product

$$\begin{aligned}\hat{\mathbf{h}} &= C_{n_1,n_2}\left(\hat{f}(x,y)\right)\hat{\mathbf{g}} \\ \hat{\mathbf{h}} &= \left(D^{-1}_{m_1+n_1,m_2+n_2} T_{n_1,n_2}\left(\hat{f}(x,y)\right) Q_{n_1,n_2}\right)\hat{\mathbf{g}}, \qquad (2.18)\end{aligned}$$

where the matrix $C_{n_1,n_2}(\hat{f}(x,y))$ has the dimensions $(m_1+n_1+1)(m_2+n_2+1) \times (n_1+1)(n_2+1)$ and is the convolution matrix for two bivariate polynomials in Bernstein form defined in the rectangular domain. The partitioned structure of $C_{n_1,n_2}(\hat{f}(x,y))$ is given by

$$\begin{bmatrix}
\frac{C_{n_1}\left(\hat{f}_0(x)\right)\binom{m_2}{0}\binom{n_2}{0}}{\binom{m_2+n_2}{0}} & & & \\
\frac{C_{n_1}\left(\hat{f}_1(x)\right)\binom{m_2}{1}\binom{n_2}{0}}{\binom{m_2+n_2}{1}} & \frac{C_{n_1}\left(\hat{f}_0(x)\right)\binom{m_2}{0}\binom{n_2}{1}}{\binom{m_2+n_2}{1}} & & \\
\vdots & \frac{C_{n_1}\left(\hat{f}_1(x)\right)\binom{m_2}{1}\binom{n_2}{1}}{\binom{m_2+n_2}{3}} & \ddots & \\
\vdots & \vdots & \ddots & \frac{C_{n_1}\left(\hat{f}_0(x)\right)\binom{m_2}{0}\binom{n_2}{n_2}}{\binom{m_2+n_2}{n_2}} \\
\frac{C_{n_1}\left(\hat{f}_{m_2}(x)\right)\binom{m_2}{m_2}\binom{n_2}{0}}{\binom{m_2+n_2}{m_2}} & \vdots & & \frac{C_{n_1}\left(\hat{f}_1(x)\right)\binom{m_2}{1}\binom{n_2}{n_2}}{\binom{m_2+n_2}{n_2+1}} \\
 & \frac{C_{n_1}\left(\hat{f}_{m_2}(x)\right)\binom{m_2}{m_2}\binom{n_2}{1}}{\binom{m_2+n_2}{m_2+1}} & & \vdots \\
 & & \ddots & \vdots \\
 & & & \frac{C_{n_1}\left(\hat{f}_{m_2}(x)\right)\binom{m_2}{m_2}\binom{n_2}{n_2}}{\binom{m_2+n_2}{m_2+n_2}}
\end{bmatrix}. \qquad (2.19)$$

In (2.15) the bivariate polynomial $\hat{f}(x,y)$ was defined as a polynomial in y whose coeffi-

cients were polynomials in x which were denoted $\hat{f}_i(x)$. The matrix (2.19) has a partitioned structure and each matrix of the form $C_{n_1}(\hat{f}_j(x))$ is the convolution matrix of a univariate polynomial $\hat{f}_j(x)$ expressed in Bernstein form, multiplied by a polynomial of degree n_1. These matrices have dimensions $(m_1+n_1+1)\times(n_1+1)$ and have the same structure as the univariate convolution matrix which is defined in (2.9).

The matrix (2.19) can also be thought of as the product of three matrices, $D^{-1}_{m_1+n_1,m_2+n_2}$, $T_{n_1,n_2}(\hat{f}(x,y))$ and Q_{n_1,n_2}, which are now described. The block diagonal matrix $D^{-1}_{m_1+n_1,m_2+n_2}$ of order $(m_1+n_1+1)(m_2+n_2+1)$ is given by

$$D^{-1}_{m_1+n_1,m_2+n_2} = \operatorname{diag}\left[\ \tfrac{1}{\binom{m_2+n_2}{0}}D^{-1}_{m_1+n_1},\quad \tfrac{1}{\binom{m_2+n_2}{1}}D^{-1}_{m_1+n_1},\quad \ldots,\quad \tfrac{1}{\binom{m_2+n_2}{m_2+n_2}}D^{-1}_{m_1+n_1}\ \right],$$

where the diagonal matrix $D^{-1}_{m_1+n_1}$ is given by (2.10). The matrix $T_{n_1,n_2}(\hat{f}(x,y))$ is the Toeplitz matrix $T_{n_1,n_2}(\hat{f}(x,y))$ of a bivariate polynomial $\hat{f}(x,y)$ and is given by

$$\begin{bmatrix}
T_{n_1}\left(\hat{f}_0(x)\right)\binom{m_2}{0} & & & & \\
T_{n_1}\left(\hat{f}_1(x)\right)\binom{m_2}{1} & T_{n_1}\left(\hat{f}_0(x)\right)\binom{m_2}{0} & & & \\
\vdots & T_{n_1}\left(\hat{f}_1(x)\right)\binom{m_2}{1} & \ddots & & \\
\vdots & \vdots & \ddots & T_{n_1}\left(\hat{f}_0(x)\right)\binom{m_2}{0} & \\
T_{n_1}\left(\hat{f}_{m_2}(x)\right)\binom{m_2}{m_2} & \vdots & & T_{n_1}\left(\hat{f}_1(x)\right)\binom{m_2}{1} & \\
 & T_{n_1}\left(\hat{f}_{m_2}(x)\right)\binom{m_2}{m_2} & & \vdots & \\
 & & \ddots & \vdots & \\
 & & & T_{n_1}\left(\hat{f}_{m_2}(x)\right)\binom{m_2}{m_2} &
\end{bmatrix}, \tag{2.20}$$

where each matrix $T_{n_1}(\hat{f}_i(x))$ is a Toeplitz matrix of the same structure as (2.11).
The block diagonal matrix $Q_{n_1,n_2}\in\mathbb{R}^{(n_1+1)\times(n_2+1)}$ of order $(n_1+1)(n_2+1)$ is given by

$$Q_{n_1,n_2} = \operatorname{diag}\left[\ Q_{n_1}\tbinom{n_2}{0},\quad Q_{n_1}\tbinom{n_2}{1},\quad \ldots,\quad Q_{n_1}\tbinom{n_2}{n_2}\ \right],$$

where each matrix Q_{n_1} is of the same structure as (2.12).

The vectors $\hat{\mathbf{g}}$ and $\hat{\mathbf{h}}$ in (2.18) are vectors of the coefficients of the bivariate polynomials $\hat{g}(x,y)$ and $\hat{h}(x,y)$ and these vectors follow the format described in Section 2.2.2. The vector $\hat{\mathbf{g}}$ is given by

$$\hat{\mathbf{g}} = \left[\ \hat{\mathbf{g}}_0,\quad \hat{\mathbf{g}}_1,\quad \ldots,\quad \hat{\mathbf{g}}_{n_2}\ \right]^T \in \mathbb{R}^{(n_2+1)(n_1+1)\times 1},$$

where each vector $\hat{\mathbf{g}}_i$ contains the coefficients of the polynomial $\hat{g}_i(x)$ of degree n_1 and is given by

$$\hat{\mathbf{g}}_i = \left[\ \hat{b}_{0,j},\quad \hat{b}_{1,j}\quad \ldots,\quad \hat{b}_{n_1,j}\ \right]^T \in \mathbb{R}^{n_1+1}.$$

The vector $\hat{\mathbf{h}}$ is given by

$$\hat{\mathbf{h}} = \left[\ \hat{\mathbf{h}}_0,\quad \hat{\mathbf{h}}_1,\quad \ldots,\quad \hat{\mathbf{h}}_{m_2+n_2}\ \right]^T \in \mathbb{R}^{(m_2+1)(m_1+1)\times 1},$$

where each vector $\hat{\mathbf{h}}_i$ contains the coefficients of the polynomial $\hat{h}_i(x)$ of degree $(m_1 + n_1)$

$$\hat{\mathbf{h}}_i = \begin{bmatrix} \hat{c}_{0,i}, & \hat{c}_{1,i}, & \ldots, & \hat{c}_{m_1+n_1,i} \end{bmatrix}^T \in \mathbb{R}^{m_1+n_1+1}.$$

Example 2.2.2. Consider the exact polynomial $\hat{f}(x, y)$ with degree structure $(2, 2)$ which is given by

$$\begin{aligned} \hat{f}(x,y) &= 7B_0^2(x)B_0^2(y) + 4B_1^2(x)B_0^2(y) + 1B_0^2(x)B_1^2(y) + 6B_2^2(x)B_0^2(y) \\ &\quad + 5B_1^2(x)B_1^2(y) + 2B_0^2(x)B_2^2(y) + 3B_2^2(x)B_1^2(y) + 8B_1^2(x)B_2^2(y) \\ &\quad + 5B_1^2(x)B_1^2(y) \\ &= \begin{bmatrix} B_0^2(x) & B_1^2(x) & B_2^2(x) \end{bmatrix} \begin{bmatrix} 7 & 1 & 2 \\ 4 & 5 & 8 \\ 6 & 3 & 9 \end{bmatrix} \begin{bmatrix} B_0^2(y) \\ B_1^2(y) \\ B_2^2(y) \end{bmatrix} \end{aligned}$$

and the polynomial $\hat{g}(x, y)$ with relative degree structure $(n_1, n_2) = (1, 1)$ which is given as

$$\begin{aligned} \hat{g}(x,y) &= 2B_0^1(x)B_0^1(y) + 3B_1^1(x)B_0^1(y) + 4B_0^1(x)B_1^1(y) + 7B_1^1(x)B_1^1(y) \\ &= \begin{bmatrix} B_0^1(x) & B_1^1(x) \end{bmatrix} \begin{bmatrix} 2 & 4 \\ 3 & 7 \end{bmatrix} \begin{bmatrix} B_0^1(y) \\ B_1^1(y) \end{bmatrix}. \end{aligned}$$

The polynomial $\hat{f}(x, y)$ can be thought of as a combination of the three polynomials $\hat{f}_0(x)$, $\hat{f}_1(x)$ and $\hat{f}_2(x)$ and is given by

$$\hat{f}(x,y) = B_0^2(y)\hat{f}_0(x) + B_1^2(y)\hat{f}_1(x) + B_2^2(y)\hat{f}_2(x),$$

where

$$\begin{aligned} \hat{f}_0(x) &= 7B_0^2(x) + 4B_1^2(x) + 6B_2^2(x), \\ \hat{f}_1(x) &= 1B_0^2(x) + 5B_1^2(x) + 3B_2^2(x), \\ \hat{f}_2(x) &= 2B_0^2(x) + 8B_1^2(x) + 9B_2^2(x). \end{aligned}$$

The polynomial $\hat{g}(x, y)$ is the combination of polynomials $\hat{g}_0(x)$ and $\hat{g}_1(x)$

$$\hat{g}(x,y) = \hat{g}_0(x)B_0^1(y) + \hat{g}_1(x)B_1^1(y),$$

where

$$\begin{aligned} \hat{g}_0(x) &= 2B_0^1(x) + 3B_1^1(x), \\ \hat{g}_1(x) &= 4B_0^1(x) + 7B_1^1(x). \end{aligned}$$

The product $\hat{h}(x, y)$ is a polynomial of degree $(3, 3)$ and is the sum of four polynomials $\hat{h}_0(x), \ldots, \hat{h}_3(x)$. The coefficients of $\hat{h}(x, y)$ are computed by the matrix-vector multipli-

cation

$$C_{n_1,n_2}\left(\hat{f}(x,y)\right)\hat{\mathbf{g}} = \hat{\mathbf{h}}$$

$$\begin{bmatrix} \frac{C_1(\hat{f}_0(x))\binom{2}{0}\binom{1}{0}}{\binom{3}{0}} & \\ \frac{C_1(\hat{f}_1(x))\binom{2}{1}\binom{1}{0}}{\binom{3}{1}} & \frac{C_1(\hat{f}_0(x))\binom{2}{0}\binom{1}{1}}{\binom{3}{1}} \\ \frac{C_1(\hat{f}_2(x))\binom{2}{2}\binom{1}{0}}{\binom{3}{2}} & \frac{C_1(\hat{f}_1(x))\binom{2}{1}\binom{1}{1}}{\binom{3}{2}} \\ & \frac{C_1(\hat{f}_2(x))\binom{2}{2}\binom{2}{2}}{\binom{3}{3}} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{g}}_0 \\ \hat{\mathbf{g}}_1 \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{h}}_0 \\ \hat{\mathbf{h}}_1 \\ \hat{\mathbf{h}}_2 \\ \hat{\mathbf{h}}_3 \end{bmatrix}. \tag{2.21}$$

The convolution matrix $C_{1,1}(\hat{f}(x,y))$ is given by

$$\left[\begin{array}{cc|cc} 7\frac{\binom{2}{0}\binom{1}{0}\binom{2}{0}\binom{1}{0}}{\binom{3}{0}\binom{3}{0}} & & & \\ 4\frac{\binom{2}{1}\binom{1}{0}\binom{2}{0}\binom{1}{0}}{\binom{3}{1}\binom{3}{0}} & 7\frac{\binom{2}{0}\binom{1}{1}\binom{2}{0}\binom{1}{0}}{\binom{3}{1}\binom{3}{0}} & & \\ 6\frac{\binom{2}{2}\binom{1}{0}\binom{2}{0}\binom{1}{0}}{\binom{3}{2}\binom{3}{0}} & 4\frac{\binom{2}{1}\binom{1}{1}\binom{2}{0}\binom{1}{0}}{\binom{3}{2}\binom{3}{0}} & & \\ & 6\frac{\binom{2}{2}\binom{1}{1}\binom{2}{0}\binom{1}{0}}{\binom{3}{3}\binom{3}{0}} & & \\ \hline 1\frac{\binom{2}{0}\binom{1}{0}\binom{2}{1}\binom{1}{0}}{\binom{3}{0}\binom{3}{1}} & & 7\frac{\binom{2}{0}\binom{1}{0}\binom{2}{0}\binom{1}{1}}{\binom{3}{0}\binom{3}{1}} & \\ 5\frac{\binom{2}{1}\binom{1}{0}\binom{2}{1}\binom{1}{0}}{\binom{3}{1}\binom{3}{1}} & 1\frac{\binom{2}{0}\binom{1}{1}\binom{2}{1}\binom{1}{0}}{\binom{3}{1}\binom{3}{1}} & 4\frac{\binom{2}{1}\binom{1}{0}\binom{2}{0}\binom{1}{1}}{\binom{3}{1}\binom{3}{1}} & 7\frac{\binom{2}{0}\binom{1}{1}\binom{2}{0}\binom{1}{1}}{\binom{3}{1}\binom{3}{1}} \\ 3\frac{\binom{2}{2}\binom{1}{0}\binom{2}{1}\binom{1}{0}}{\binom{3}{2}\binom{3}{1}} & 5\frac{\binom{2}{1}\binom{1}{1}\binom{2}{1}\binom{1}{0}}{\binom{3}{2}\binom{3}{1}} & 6\frac{\binom{2}{2}\binom{1}{0}\binom{2}{0}\binom{1}{1}}{\binom{3}{2}\binom{3}{1}} & 4\frac{\binom{2}{1}\binom{1}{1}\binom{2}{0}\binom{1}{1}}{\binom{3}{2}\binom{3}{1}} \\ & 3\frac{\binom{2}{2}\binom{1}{1}\binom{2}{1}\binom{1}{0}}{\binom{3}{3}\binom{3}{1}} & & 6\frac{\binom{2}{2}\binom{1}{1}\binom{2}{0}\binom{1}{1}}{\binom{3}{3}\binom{3}{1}} \\ \hline 2\frac{\binom{2}{0}\binom{1}{0}\binom{2}{2}\binom{1}{0}}{\binom{3}{0}\binom{3}{2}} & & 1\frac{\binom{2}{0}\binom{1}{0}\binom{2}{1}\binom{1}{0}}{\binom{3}{0}\binom{3}{2}} & \\ 8\frac{\binom{2}{1}\binom{1}{0}\binom{2}{2}\binom{1}{0}}{\binom{3}{1}\binom{3}{2}} & 2\frac{\binom{2}{0}\binom{1}{1}\binom{2}{2}\binom{1}{0}}{\binom{3}{1}\binom{3}{2}} & 5\frac{\binom{2}{1}\binom{1}{0}\binom{2}{1}\binom{1}{1}}{\binom{3}{1}\binom{3}{2}} & 1\frac{\binom{2}{0}\binom{1}{1}\binom{2}{1}\binom{1}{1}}{\binom{3}{1}\binom{3}{2}} \\ 9\frac{\binom{2}{2}\binom{1}{0}\binom{2}{2}\binom{1}{0}}{\binom{3}{2}\binom{3}{2}} & 8\frac{\binom{2}{1}\binom{1}{1}\binom{2}{2}\binom{1}{0}}{\binom{3}{2}\binom{3}{2}} & 3\frac{\binom{2}{2}\binom{1}{0}\binom{2}{1}\binom{1}{1}}{\binom{3}{2}\binom{3}{2}} & 5\frac{\binom{2}{1}\binom{1}{1}\binom{2}{1}\binom{1}{1}}{\binom{3}{2}\binom{3}{2}} \\ & 9\frac{\binom{2}{2}\binom{1}{1}\binom{2}{2}\binom{1}{0}}{\binom{3}{3}\binom{3}{2}} & & 3\frac{\binom{2}{2}\binom{1}{1}\binom{2}{1}\binom{1}{1}}{\binom{3}{3}\binom{3}{2}} \\ \hline & & 2\frac{\binom{2}{0}\binom{1}{0}\binom{2}{2}\binom{1}{1}}{\binom{3}{0}\binom{3}{3}} & \\ & & 8\frac{\binom{2}{1}\binom{1}{0}\binom{2}{2}\binom{1}{1}}{\binom{3}{1}\binom{3}{3}} & 2\frac{\binom{2}{0}\binom{1}{1}\binom{2}{2}\binom{1}{1}}{\binom{3}{1}\binom{3}{3}} \\ & & 9\frac{\binom{2}{2}\binom{1}{0}\binom{2}{2}\binom{1}{1}}{\binom{3}{2}\binom{3}{3}} & 8\frac{\binom{2}{1}\binom{1}{1}\binom{2}{2}\binom{1}{1}}{\binom{3}{2}\binom{3}{3}} \\ & & 0 & 9\frac{\binom{2}{2}\binom{1}{1}\binom{2}{2}\binom{1}{1}}{\binom{3}{3}\binom{3}{3}} \end{array}\right],$$

where the constant terms are the coefficients of the polynomial $\hat{f}(x,y)$. Vectors $\hat{\mathbf{g}}$ and $\hat{\mathbf{h}}$ in (2.21) are given by

$$\hat{\mathbf{g}} = \left[\begin{array}{cc|cc} \hat{\mathbf{g}}_0, & \hat{\mathbf{g}}_1 \end{array}\right]^T = \left[\begin{array}{cc|cc} 2 & 3 & 4 & 7 \end{array}\right]^T$$

$$\hat{\mathbf{h}} = \left[\begin{array}{cccc|cccc|cccc|cccc} 14 & 12\frac{1}{8} & 12 & 18 & 10\frac{2}{3} & 14\frac{1}{9} & 16\frac{8}{9} & 20 & 4 & 14\frac{2}{3} & 25\frac{5}{9} & 23 & 8 & 26 & 49\frac{1}{3} & 63 \end{array}\right]^T.$$

The polynomial $\hat{h}(x,y)$ is therefore given by

$$\begin{bmatrix} B_0^3(x), & B_1^3(x), & B_2^3(x), & B_3^3(x) \end{bmatrix} \begin{bmatrix} 14 & 10\frac{2}{3} & 4 & 8 \\ 12\frac{1}{8} & 14\frac{1}{9} & 14\frac{2}{3} & 26 \\ 12 & 16\frac{8}{9} & 25\frac{5}{9} & 49\frac{1}{3} \\ 18 & 20 & 23 & 63 \end{bmatrix} \begin{bmatrix} B_0^3(y) \\ B_1^3(y) \\ B_2^3(y) \\ B_3^3(y) \end{bmatrix}.$$

□

2.2.3 The Bivariate Bernstein Polynomial over the Triangular Domain

In Section 2.1.3 the triangular Bézier surface patch was introduced and the defining parametric equations are bivariate polynomials over the triangular domain. The bivariate polynomial over the triangular domain $\hat{f}(x,y)$ of total degree m is given by

$$\begin{aligned} \hat{f}(x,y) &= \sum_{i_1+i_2=0}^{m} \hat{a}_{i_1,i_2} B_{i_1,i_2}^m(x,y) \\ &= \sum_{i_1+i_2=0}^{m} \hat{a}_{i_1,i_2} \binom{m}{i_1,i_2} (1-x-y)^{m-i_1-i_2} x^{i_1} y^{i_2}, \end{aligned} \tag{2.22}$$

where the term $\binom{m}{i_1,i_2}$ is defined in (2.3). The bivariate polynomial in Bernstein form has $\binom{m+2}{2}$ coefficients and is quite different from the bivariate polynomial over a rectangular domain defined in (2.13). The polynomial $\hat{f}(x,y)$ can be considered as the sum of the set of polynomials $\{\, \hat{f}_i(x,y) \mid i = 0,\ldots,m \,\}$ and is given by

$$\hat{f}(x,y) = \hat{f}_0(x,y) + \hat{f}_1(x,y) + \cdots + \hat{f}_m(x,y),$$

where each polynomial in the set $\{\, \hat{f}_k(x,y) \mid k = 1,\ldots,m \,\}$ is given by

$$\hat{f}_k(x,y) = \sum_{j=0}^{k} \hat{a}_{k-j,j} \binom{m}{k-j,j} (1-x-y)^{m-k} x^{k-j} y^j. \tag{2.23}$$

Vector Representation and Multiplication

In Section 2.2.2 the vector representation and multiplication of bivariate polynomials over a rectangular domain was considered, and now the representation and multiplication of bivariate polynomials over a triangular domain is described.

The vector representation $\hat{\mathbf{f}} \in \mathbb{R}^{\binom{m+2}{2}}$ of the polynomial $\hat{f}(x,y)$ is a column vector given by

$$\hat{\mathbf{f}} = \begin{bmatrix} \hat{\mathbf{f}}_0, & \hat{\mathbf{f}}_1, & \ldots, & \hat{\mathbf{f}}_m \end{bmatrix}^T,$$

where each vector $\hat{\mathbf{f}}_k \in \mathbb{R}^{k+1}$ from the set of vectors $\{\, \hat{\mathbf{f}}_k \mid k = 0,\ldots,m \,\}$ contains the coefficients of $\hat{f}_k(x,y)$ seen in (2.23) and is given by

$$\hat{\mathbf{f}}_k = \begin{bmatrix} \hat{a}_{k,0}, & \hat{a}_{k-1,1}, & \ldots, & \hat{a}_{0,k} \end{bmatrix}^T. \tag{2.24}$$

Consider the bivariate polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ in Bernstein form over the triangular domain, of total degrees m and n respectively. The polynomial $\hat{f}(x,y)$ is defined in (2.22) and $\hat{g}(x,y)$ is given by

$$\hat{g}(x,y) = \sum_{i+j=0}^{n} \hat{b}_{i,j} B_{i,j}^{n}(x,y).$$

The polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ can be thought of as the sums of the sets of polynomials $\{\, \hat{f}_j(x,y) \mid j = 0,\ldots,m \,\}$ and $\{\, \hat{g}_j(x,y) \mid j = 0,\ldots,n \,\}$ respectively, where $\hat{f}_j(x,y)$ is defined in (2.23) and the polynomials $\hat{g}_j(x,y)$ are given by

$$\hat{g}_j(x,y) = \sum_{i=0}^{j} \hat{b}_{i,j-i} B_{i,j-i}^{n}(x,y)$$

such that

$$\hat{g}(x,y) = \hat{g}_0(x,y) + \hat{g}_1(x,y) + \cdots + \hat{g}_{n-1}(x,y) + \hat{g}_n(x,y).$$

The product $\hat{h}(x,y) = \hat{f}(x,y)\hat{g}(x,y)$ of degree $(m+n)$ is given by

$$\hat{h}(x,y) = \sum_{i+j=0}^{m} \hat{c}_{i,j} \binom{m+n}{i,j} (1-x-y)^{m+n-i-j} x^i y^j$$

and $\hat{h}(x,y)$ can be thought of as the sum of polynomials $\{\, \hat{h}_j(x,y) \mid j = 0,\ldots,m+n \,\}$, where

$$\hat{h}_j(x,y) = \sum_{i=0}^{j} \hat{c}_{i,j-i} \binom{m+n}{i,j-i} (1-x-y)^{m-j} x^i y^{j-i}.$$

Expressed as a product of $\hat{f}(x,y)$ and $\hat{g}(x,y)$, $\hat{h}(x,y)$ is given as

$$\begin{aligned}
\hat{h}(x,y) = {} & \Big(\hat{f}_0(x,y)\hat{g}_0(x,y)\Big) + \Big(\hat{f}_0(x,y)\hat{g}_1(x,y) + \hat{f}_1(x,y)\hat{g}_0(x,y)\Big) \\
& + \Big(\hat{f}_2(x,y)\hat{g}_0(x,y) + \hat{f}_1(x,y)\hat{g}_1(x,y) + \hat{f}_0(x,y)\hat{g}_2(x,y)\Big) + \ldots \\
& \cdots + \Big(\hat{f}_m(x,y)\hat{g}_n(x,y)\Big), \qquad (2.25)
\end{aligned}$$

where one of the products of the form $\hat{f}_s(x,y)\hat{g}_t(x,y)$ in the summation in (2.25) is given by

$$\begin{aligned}
& = \sum_{i=0}^{s}\sum_{j=0}^{t} \hat{a}_{i,s-i}\hat{b}_{j,t-j} \binom{m}{i,s-i} \binom{n}{j,t-j} x^{i+j} y^{s+t-i-j} (1-x-y)^{m+n-s-t} \\
& = \sum_{i=0}^{s}\sum_{j=0}^{t} \hat{a}_{i,s-i}\hat{b}_{j,t-j} \frac{\binom{m}{i,s-i}\binom{n}{j,t-j}}{\binom{m+n}{i+j,s+t-i-j}} \binom{m+n}{i+j,s+t-i-j} x^{i+j} y^{s+t-i-j} (1-x-y)^{m+n-s-t} \\
& = \sum_{i=0}^{s}\sum_{j=0}^{t} \frac{\hat{a}_{i,s-i}\hat{b}_{j,t-j}\binom{m}{i,s-i}\binom{n}{j,t-j}}{\binom{m+n}{i+j,s+t-i-j}} B_{i+j,s+t-i-j}^{s+t}(x,y). \qquad (2.26)
\end{aligned}$$

The expression (2.26) can be written as the matrix-vector product

$$\dot{C}_t\left(\hat{f}_s(x,y)\right)\hat{\mathbf{g}}_t \quad \text{or} \quad \left(\dot{D}^{-1}_{m+n,s+t}\dot{T}_t\left(\hat{f}_s(x,y)\right)\dot{Q}_{n,t}\right)\hat{\mathbf{g}}_t. \tag{2.27}$$

The matrix $\dot{C}_t(\hat{f}_s(x,y))$ is the convolution matrix of a polynomial $\hat{f}_s(x,y)$ multiplied by $\hat{g}_t(x,y)$ and is the product of three matrices $\dot{D}^{-1}_{m+n,s+t}$, $T_t(\hat{f}_s(x,y))$ and $\dot{Q}_{n,t}$. The diagonal matrix $\dot{D}^{-1}_{m+n,s+t} \in \mathbb{R}^{(s+t+1)\times(s+t+1)}$ is given by

$$\dot{D}^{-1}_{m+n,s+t} = \operatorname{diag}\left[\ \tfrac{1}{\binom{m+n}{0,s+t}},\ \ \tfrac{1}{\binom{m+n}{1,s+t-1}},\ \ \ldots,\ \ \tfrac{1}{\binom{m+n}{s+t,0}}\ \right], \tag{2.28}$$

the Toeplitz matrix $\dot{T}_t(\hat{f}_s(x,y)) \in \mathbb{R}^{(s+t+1)\times(t+1)}$ is given by

$$\begin{bmatrix}
\hat{a}_{s,0}\binom{m}{s,0} & & & \\
\hat{a}_{s-1,1}\binom{m}{s-1,1} & \hat{a}_{s,0}\binom{m}{s,0} & & \\
\vdots & \hat{a}_{s-1,1}\binom{m}{s-1,1} & \ddots & \\
\vdots & \vdots & \ddots & \hat{a}_{s,0}\binom{m}{s,0} \\
\hat{a}_{1,s-1}\binom{m}{1,s-1} & \vdots & & \hat{a}_{s-1,1}\binom{m}{s-1,1} \\
\hat{a}_{0,s}\binom{m}{0,s} & \hat{a}_{1,s-1}\binom{m}{1,s-1} & & \vdots \\
 & \hat{a}_{0,s}\binom{m}{0,s} & \ddots & \vdots \\
 & & \ddots & \hat{a}_{1,s-1}\binom{m}{1,s-1} \\
 & & & \hat{a}_{0,s}\binom{m}{0,s}
\end{bmatrix}, \tag{2.29}$$

and the diagonal matrix $\dot{Q}_{n,t} \in \mathbb{R}^{t+1}$ is given by

$$\dot{Q}_{n,t} = \operatorname{diag}\left[\ \tbinom{n}{t,0},\ \ \tbinom{n}{t-1,1},\ \ \ldots,\ \ \tbinom{n}{0,t}\ \right] \in \mathbb{R}^{(t+1)\times(t+1)}. \tag{2.30}$$

The vector $\hat{\mathbf{g}}_t \in \mathbb{R}^{t+1}$ in (2.27) has the same structure as the vector of the coefficients of $\hat{f}(x,y)$ defined in Equation (2.24) and is given by

$$\hat{\mathbf{g}}_t = \left[\ \hat{b}_{0,t},\ \ \hat{b}_{1,t-1},\ \ \ldots,\ \ \hat{b}_{t,0}\ \right]^T. \tag{2.31}$$

The entire expression (2.25) can be written as the matrix-vector product

$$C_n\left(\hat{f}(x,y)\right)\hat{\mathbf{g}} = \hat{\mathbf{h}}, \tag{2.32}$$

where the bivariate convolution matrix $C_n(\hat{f}(x,y))$ is given by

$$C_n\left(\hat{f}(x,y)\right) = \begin{bmatrix} \dot{C}_0\left(\hat{f}_0(x,y)\right) & & & \\ \dot{C}_0\left(\hat{f}_1(x,y)\right) & \dot{C}_1\left(\hat{f}_0(x,y)\right) & & \\ \vdots & \dot{C}_1\left(\hat{f}_1(x,y)\right) & \ddots & \\ \vdots & \vdots & \ddots & \dot{C}_n\left(\hat{f}_0(x,y)\right) \\ \dot{C}_0\left(\hat{f}_m(x,y)\right) & \vdots & & \dot{C}_n\left(\hat{f}_1(x,y)\right) \\ & \dot{C}_1\left(\hat{f}_m(x,y)\right) & & \vdots \\ & & \ddots & \vdots \\ & & & \dot{C}_n\left(\hat{f}_m(x,y)\right) \end{bmatrix} \tag{2.33}$$

and each partition $\dot{C}_t(\hat{f}_s(x,y))$ for $t = 0,\ldots,n;\ s = 0,\ldots,m$, is a convolution matrix with the same structure as (2.27).

The bivariate convolution matrix $C_n(\hat{f}(x,y))$ can also be written as

$$C_n(\hat{f}(x,y)) = D_{m+n}^{-1} T_k\left(\hat{f}(x,y)\right) Q_n, \tag{2.34}$$

where the block diagonal matrix $D_{m+n}^{-1} \in \mathbb{R}^{\binom{m+n+2}{2}\times\binom{m+n+2}{2}}$ is given by

$$D_{m+n}^{-1} = \text{diag}\left[\ \dot{D}_{m+n,0}^{-1},\quad \dot{D}_{m+n,1}^{-1},\quad \ldots,\quad \dot{D}_{m+n,m+n}^{-1}\ \right]$$

and each $\dot{D}_{m+n,i}^{-1}$ for $i = 0,\ldots,m+n$ has the same structure as (2.28). The matrix $T_n(\hat{f}(x,y))$ has a partitioned structure and is given by

$$T_n\left(\hat{f}(x,y)\right) = \begin{bmatrix} \dot{T}_0\left(\hat{f}_0(x,y)\right) & & & \\ \dot{T}_0\left(\hat{f}_1(x,y)\right) & \dot{T}_1\left(\hat{f}_0(x,y)\right) & & \\ \vdots & \dot{T}_1\left(\hat{f}_1(x,y)\right) & \ddots & \\ \vdots & \vdots & \ddots & \dot{T}_n\left(\hat{f}_0(x,y)\right) \\ \dot{T}_0\left(\hat{f}_m(x,y)\right) & \vdots & & \dot{T}_n\left(\hat{f}_1(x,y)\right) \\ & \dot{T}_1\left(\hat{f}_m(x,y)\right) & & \vdots \\ & & \ddots & \vdots \\ & & & \dot{T}_n\left(\hat{f}_m(x,y)\right) \end{bmatrix}, \tag{2.35}$$

where each of the non-zero partitions $\dot{T}_j(\hat{f}_i(x,y)) \in \mathbb{R}^{(i+j+1)\times(j+1)}$ are of the same structure as the Toeplitz matrix in (2.29). The block diagonal matrix $Q_n \in \mathbb{R}^{\binom{n+2}{2}\times\binom{n+2}{2}}$ is given by

$$Q_n = \text{diag}\left[\ \dot{Q}_{n,0},\quad \dot{Q}_{n,1},\quad \ldots,\quad \dot{Q}_{n,n}\ \right], \tag{2.36}$$

where the matrices $\dot{Q}_{n,t}$ for $t = 0,\ldots,n$ are of the same structure as the matrix in (2.30).

The vector $\hat{\mathbf{g}}$ in (2.32) is given by

$$\hat{\mathbf{g}} = \left[\begin{array}{cccc} \hat{\mathbf{g}}_0, & \hat{\mathbf{g}}_1, & \ldots, & \hat{\mathbf{g}}_n \end{array} \right]^T \in \mathbb{R}^{\binom{n+2}{2}},$$

where each $\hat{\mathbf{g}}_i \in \mathbb{R}^{i+1}$ is a vector with the same structure as (2.31) and the vector $\hat{\mathbf{h}}$ containing the coefficients of the polynomial $\hat{h}(x, y)$ is given by

$$\hat{\mathbf{h}} = \left[\begin{array}{cccc} \hat{\mathbf{h}}_0, & \hat{\mathbf{h}}_1, & \ldots, & \hat{\mathbf{h}}_{m+n} \end{array} \right]^T \in \mathbb{R}^{\binom{m+n+2}{2}}. \tag{2.37}$$

Example 2.2.3. The polynomial $\hat{f}(x, y)$ of degree $m = 2$ is given in Bernstein form by

$$\begin{aligned} \hat{f}(x,y) &= \sum_{i+j=0}^{2} \hat{a}_{i,j} \binom{2}{i,j} x^i y^j (1-x-y)^{2-i-j} \\ &= \hat{a}_{0,0} B^2_{0,0}(x,y) + \hat{a}_{1,0} B^2_{1,0}(x,y) + \hat{a}_{0,1} B^2_{0,1}(x,y) \\ &\quad + \hat{a}_{2,0} B^2_{2,0}(x,y) + \hat{a}_{1,1} B^2_{1,1}(x,y) + \hat{a}_{0,2} B^2_{0,2}(x,y). \end{aligned}$$

Alternatively, $\hat{f}(x, y)$ can be considered as the sum of the set of polynomials $\{ \hat{f}_i(x, y) \mid i = 0, 1, 2\}$ and is given by $\hat{f}(x, y) = \hat{f}_0(x, y) + \hat{f}_1(x, y) + \hat{f}_2(x, y)$, where

$$\begin{aligned} \hat{f}_0(x,y) &= \hat{a}_{0,0} \binom{2}{0,0} (1-x-y)^2, \\ \hat{f}_1(x,y) &= \left(\hat{a}_{1,0} \binom{2}{1,0} x + \hat{a}_{0,1} \binom{2}{0,1} y \right) (1-x-y), \\ \hat{f}_2(x,y) &= \hat{a}_{2,0} \binom{2}{2,0} x^2 + \hat{a}_{1,1} \binom{2}{1,1} xy + \hat{a}_{0,2} \binom{2}{0,2} y^2. \end{aligned}$$

The polynomial $\hat{g}(x, y)$ of degree $n = 2$ is given by

$$\hat{g}(x,y) = \sum_{i+j=0}^{2} \hat{b}_{i,j} \binom{2}{i,j} x^i y^j (1-x-y)^{2-i-j}$$

and is the sum of polynomials $\hat{g}_0(x, y)$, $\hat{g}_1(x, y)$ and $\hat{g}_2(x, y)$, where

$$\begin{aligned} \hat{g}_0(x,y) &= \hat{b}_{0,0} \binom{2}{0,0} (1-x-y)^2, \\ \hat{g}_1(x,y) &= \left(\hat{b}_{1,0} \binom{2}{1,0} x + \hat{b}_{0,1} \binom{2}{0,1} y \right) (1-x-y), \\ \hat{g}_2(x,y) &= \hat{b}_{2,0} \binom{2}{1,0} x^2 + \hat{b}_{1,1} \binom{2}{1,1} xy + \hat{b}_{0,2} \binom{2}{0,2} y^2. \end{aligned}$$

The product $\hat{h}(x, y)$ is given in terms of the set of polynomials $\{ \hat{f}_i(x, y) \mid i = 0, 1, 2 \}$ and $\{ \hat{g}_i(x, y) \mid i = 0, 1, 2 \}$

$$\hat{f}(x,y) \times \hat{g}(x,y) = \Big(\hat{f}_0(x,y) + \hat{f}_1(x,y) + \hat{f}_2(x,y) \Big) \Big(\hat{g}_0(x,y) + \hat{g}_1(x,y) + \hat{g}_2(x,y) \Big),$$

so $\hat{h}(x, y)$ in terms of $\hat{f}_0(x, y)$, $\hat{f}_1(x, y)$ and $\hat{f}_2(x, y)$, and $\hat{g}_0(x, y)$, $\hat{g}_1(x, y)$ and $\hat{g}_2(x, y)$ is

given by

$$\begin{aligned}\hat{h}(x,y) = \hat{f}_2(x,y)\hat{g}_2(x,y) &+ \Big(\hat{f}_2(x,y)\hat{g}_1(x,y) + \hat{f}_1(x,y)\hat{g}_2(x,y)\Big) \\ &+ \Big(\hat{f}_2(x,y)\hat{g}_0(x,y) + \hat{f}_1(x,y)\hat{g}_1(x,y) + \hat{f}_0(x,y)\hat{g}_2(x,y)\Big) \\ &+ \Big(\hat{f}_1(x,y)\hat{g}_0(x,y) + \hat{f}_0(x,y)\hat{g}_1(x,y)\Big) + \hat{f}_0(x,y)\hat{g}_0(x,y).\end{aligned}$$

This can be written as the matrix-vector product

$$C_2\left(\hat{f}(x,y)\right)\hat{\mathbf{g}} = \hat{\mathbf{h}},$$

$$\begin{bmatrix} \dot{C}_0\left(\hat{f}_0(x,y)\right) & & \\ \dot{C}_0\left(\hat{f}_1(x,y)\right) & \dot{C}_1\left(\hat{f}_0(x,y)\right) & \\ \dot{C}_0\left(\hat{f}_2(x,y)\right) & \dot{C}_1\left(\hat{f}_1(x,y)\right) & \dot{C}_2\left(\hat{f}_0(x,y)\right) \\ & \dot{C}_1\left(\hat{f}_2(x,y)\right) & \dot{C}_2\left(\hat{f}_1(x,y)\right) \\ & & \dot{C}_2\left(\hat{f}_2(x,y)\right) \end{bmatrix} \begin{bmatrix} \hat{\mathbf{g}}_0 \\ \hat{\mathbf{g}}_1 \\ \hat{\mathbf{g}}_2 \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{h}}_0 \\ \hat{\mathbf{h}}_1 \\ \hat{\mathbf{h}}_2 \\ \hat{\mathbf{h}}_3 \\ \hat{\mathbf{h}}_4 \end{bmatrix}$$

and $\hat{\mathbf{h}}$ contains the ordered coefficients of $\hat{h}(x,y)$.

□

Example 2.2.4. Consider the polynomials $\hat{f}(x,y)$ of degree $m = 2$ which is given by

$$\hat{f}(x,y) = 3B^2_{0,0}(x,y) + 2B^2_{1,0}(x,y) + 2B^2_{0,1}(x,y) + 1B^2_{2,0}(x,y) + 5B^2_{1,1}(x,y) + 2B^2_{0,2}(x,y)$$

and $\hat{g}(x,y)$ of degree $n = 1$ which is given by

$$\hat{g}(x,y) = 1B^1_{0,0}(x,y) + 3B^1_{1,0}(x,y) + 2B^1_{0,1}(x,y).$$

The coefficients of $\hat{f}(x,y)$ and $\hat{g}(x,y)$ are given in matrix form as

$$M\left(\hat{f}\right) = \begin{array}{c} \\ i_1 = 0 \\ i_1 = 1 \\ i_1 = 2 \end{array} \begin{array}{c} \begin{array}{ccc} i_2 = 0 & i_2 = 1 & i_2 = 2 \end{array} \\ \begin{pmatrix} 3 & 2 & 2 \\ 2 & 5 & \\ 1 & & \end{pmatrix} \end{array}, \quad \text{and} \quad M\left(\hat{g}\right) = \begin{array}{c} \\ i_1 = 0 \\ i_1 = 1 \end{array} \begin{array}{c} \begin{array}{cc} i_2 = 0 & i_2 = 1 \end{array} \\ \begin{pmatrix} 1 & 2 \\ 3 & \end{pmatrix} \end{array}.$$

The polynomial $\hat{f}(x,y)$ can be thought of as the sum of polynomials $\hat{f}_0(x,y)$, $\hat{f}_1(x,y)$ and $\hat{f}_2(x,y)$

$$\hat{f}(x,y) = \hat{f}_0(x,y) + \hat{f}_1(x,y) + \hat{f}_2(x,y),$$

where

$$\begin{aligned}\hat{f}_0(x,y) &= 3B^2_{0,0}(x,y),\\ \hat{f}_1(x,y) &= 2B^2_{1,0}(x,y) + 2B^2_{0,1}(x,y),\\ \hat{f}_2(x,y) &= 1B^2_{2,0}(x,y) + 5B^2_{1,1}(x,y) + 2B^2_{0,2}(x,y).\end{aligned}$$

The polynomial $\hat{g}(x,y)$ is given by

$$\hat{g}(x,y) = \hat{g}_0(x,y) + \hat{g}_1(x,y),$$

where

$$\begin{aligned}\hat{g}_1(x,y) &= 1B^1_{0,0}(x,y)\\ \hat{g}_1(x,y) &= 3B^1_{1,0}(x,y) + 2B^1_{0,1}(x,y).\end{aligned}$$

The product $\hat{h}(x,y)$ is given by

$$D_3^{-1} T_1\left(\hat{f}(x,y)\right) Q_1 \hat{\mathbf{g}}, \tag{2.38}$$

where the diagonal matrix $D_3^{-1} \in \mathbb{R}^{10\times 10}$ is given by

$$D_3^{-1} = \operatorname{diag}\left[\begin{array}{c|cc|ccc|cccc} \frac{1}{\binom{3}{0,0}} & \frac{1}{\binom{3}{1,0}} & \frac{1}{\binom{3}{0,1}} & \frac{1}{\binom{3}{2,0}} & \frac{1}{\binom{3}{1,1}} & \frac{1}{\binom{3}{0,2}} & \frac{1}{\binom{3}{3,0}} & \frac{1}{\binom{3}{2,1}} & \frac{1}{\binom{3}{1,2}} & \frac{1}{\binom{3}{0,3}} \end{array}\right],$$

the matrix $T_1(\hat{f}(x,y)) \in \mathbb{R}^{10\times 3}$ is given by

$$T_1\left(\hat{f}(x,y)\right) = \left[\begin{array}{c|cc} 3\binom{2}{0,0} & & \\ \hline 2\binom{2}{1,0} & 3\binom{2}{0,0} & \\ 2\binom{2}{0,1} & & 3\binom{2}{0,0} \\ \hline 1\binom{2}{0,2} & 2\binom{2}{1,0} & \\ 5\binom{2}{1,1} & 2\binom{2}{0,1} & 2\binom{2}{1,0} \\ 2\binom{2}{0,2} & & 2\binom{2}{0,1} \\ \hline & 1\binom{2}{2,0} & \\ & 5\binom{2}{1,1} & 1\binom{2}{2,0} \\ & 2\binom{2}{0,2} & 5\binom{2}{1,1} \\ & & 2\binom{2}{0,2} \end{array}\right]$$

and the diagonal matrix $Q_1 \in \mathbb{R}^{3\times 3}$ is given by

$$Q_1 = \operatorname{diag}\left[\begin{array}{ccc} \binom{1}{0,0}, & \binom{1}{1,0}, & \binom{1}{0,1} \end{array}\right].$$

The matrix-vector product (2.38) is given by

$$\tilde{D}_3^{-1} T_1\left(\hat{f}(x,y)\right) \tilde{Q}_1 \hat{\mathbf{g}} = \left[\begin{array}{c|cc|ccc|cccc} 3 & \frac{20}{3} & \frac{14}{3} & \frac{13}{3} & \frac{25}{6} & \frac{10}{3} & 6 & \frac{34}{3} & \frac{32}{3} & 8 \end{array}\right]^T$$

and the matrix of the coefficients of $\hat{h}(x, y)$ is therefore given by

$$M\left(\hat{h}\right) = \begin{bmatrix} 3 & \frac{14}{3} & \frac{10}{3} & 8 \\ \frac{20}{3} & \frac{25}{6} & \frac{32}{3} & \\ \frac{13}{3} & \frac{34}{3} & & \\ 6 & & & \end{bmatrix}.$$

□

Partial Derivatives

Differentiation of bivariate polynomials in Bernstein form is required for determining the square-free factorisation in the extended version of Gauss' algorithm, where the sequence of polynomials $\{\hat{f}_i(x, y)\}$ is generated and each $\hat{f}_{i+1}(x, y)$ is the GCD of $f_i(x, y)$ and its partial derivatives with respect to x and y.

The partial derivative of $\hat{f}(x, y)$ defined in (2.22) with respect to x is given by

$$\frac{\partial \hat{f}(x, y)}{\partial x} = \sum_{i+j=0}^{m-1} m\left(\hat{a}_{i+1,j} - \hat{a}_{i,j}\right) B_{i,j}^{m-1}(x, y)$$

and the partial derivative with respect to y is given by

$$\frac{\partial \hat{f}(x, y)}{\partial y} = \sum_{i+j=0}^{m-1} m\left(\hat{a}_{i,j+1} - \hat{a}_{i,j}\right) B_{i,j}^{m-1}(x, y).$$

2.3 Conclusion

This section has introduced the Bézier curve, the rectangular Bézier surface patch and the triangular Bézier surface patch. The polynomials that define these curves and surfaces have also been discussed. It has been shown how the coefficients of the product of two polynomials are generated by the multiplication of a convolution matrix and a vector of coefficients. These definitions are used in the next chapter wherein the Sylvester matrix and subresultant matrices of two polynomials in Bernstein form are defined.

Chapter 3

The Univariate Polynomial GCD - The Two Polynomial Problem

The calculation of the points of intersection of Bézier curves reduces to a univariate polynomial root finding problem. By using the square-free factorisation algorithm (Algorithm 1) this reduces to a set of polynomial GCD problems and a set of polynomial deconvolutions. Polynomial GCD computation and deconvolution methods are developed specifically for polynomials in Bernstein form as the conversion between bases is known to be ill-conditioned [17, 26, 29].

In general it is unlikely that any two randomly chosen polynomials have a common divisor. The computation of points of intersection between two curves or surfaces require a sequence of GCD computations where, if the intersections are smooth, the polynomials in each GCD computation are not coprime. The method of AGCD computation developed in this thesis is a general method, and examples using arbitrary polynomials which are not necessarily derived from intersection problems, i.e $\hat{g}(x)$ is not necessarily the derivative of $\hat{f}(x)$, are considered.

This chapter focuses on the development of general-purpose algorithm for computing the GCD of two arbitrary polynomials $\hat{f}(x)$ and $\hat{g}(x)$ in Bernstein form, and this method, referred to as UGCD, is adapted in Chapter 4 for use in the specific GCD problem found in the square-free factorisation algorithm, that is where $\hat{g}(x)$ is the derivative of $\hat{f}(x)$. Examples in this chapter are defined such that $\hat{f}(x)$ and $\hat{g}(x)$ always have a common divisor, but their inexact counterparts $f(x)$ and $g(x)$ are highly likely to be coprime.

The aim will be to determine the coefficients of the AGCD which is close to the GCD of the exact polynomials. Since the problem is converted from a polynomial based problem to a matrix-vector problem, closeness between two polynomials is to be defined in terms of the distance between the vectors of their coefficients as described in Section 3.5. That is, the distance between the exact GCD and AGCD will be defined as $\|\hat{\mathbf{d}} - \mathbf{d}\|/\|\hat{\mathbf{d}}\|$.

Several polynomial GCD finding methods were discussed in Section 1.8 and this chapter focuses on a method which uses the Sylvester matrix and the sequence of subresultant matrices to determine the degree and coefficients of the GCD, specifically where polynomials are defined in Bernstein form.

Section 3.1 It is shown how the computation of the degree of the GCD of two univariate polynomials in Bernstein form reduces to the determination of the index of the

last numerically singular subresultant matrix in the sequence of subresultant matrices. Several variants of the subresultant matrix sequence are defined. It is also shown how the set of subresultant matrices can be constructed by a series of matrix transformations applied to the first subresultant matrix $S_1(\hat{f}(x), \hat{g}(x))$.

Section 3.2 Several methods are considered for the computation of the degree of the GCD using either the SVD or QR decomposition of the sequence of subresultant matrices. Despite being more computationally expensive, methods which use the complete set of subresultant matrices to compute the degree of the GCD will be shown to be more reliable than methods which consider only the numerical rank of the first subresultant matrix.

Section 3.3 Given that several variants of the sequence of subresultant matrices have been defined and methods for the computation of the degree of the GCD have been considered, this section determines the optimal variant of the set of subresultant matrices for use in the computation of the degree of the GCD.

Section 3.4 This section considers three preprocessing operations for each of the subresultant matrices in the subresultant matrix sequence. It will be shown that the degree of the GCD or AGCD is reliably obtained by analysis of the numerical rank of each of the subresultant matrices containing the coefficients of preprocessed polynomials. It will also be shown that approximations of coefficients of the cofactor polynomials and the GCD are more accurate than those obtained from unprocessed polynomials. Preprocessing does have a small but significant cost associated, however methods to mitigate this cost will also be described.

Section 3.5 This section describes two methods for the computation of approximations of the cofactor polynomials and the GCD or AGCD of two polynomials in Bernstein form. A simple least squares based method is used given the tth subresultant matrix $S_t(f, g)$ and an alternative method is presented in which the coefficients of the cofactor polynomials are approximated from a structured low rank approximation of the tth subresultant matrix of preprocessed polynomials $\tilde{f}_t(\omega)$ and $\alpha_t\tilde{g}_t(\omega)$. This method has previously been considered for computing the low rank approximation of the tth subresultant matrix of two polynomials in the power basis [69, 70] but is now extended to polynomials in Bernstein form.

It will be shown that this second method compares favourably with the standard least squares approach and that the approximations of $\hat{u}_t(x)$ and $\hat{v}_t(x)$ and the GCD $\hat{d}_t(x)$ obtained by this method can be some orders of magnitude more accurate.

3.1 The Computation of the Degree of the AGCD of Two Univariate Polynomials in Bernstein Form

The computation of the GCD of two polynomials $\hat{f}(x)$ and $\hat{g}(x)$ is an ill-posed problem, in that the result is highly sensitive to perturbations in the input polynomials. That is, small changes in the input polynomials result in large discontinuous changes in the output. Consequently, the GCD is only defined for exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$, and

it is highly likely that the inexact polynomials $f(x)$ and $g(x)$ (obtained by the addition of minimal noise to $\hat{f}(x)$ and $\hat{g}(x)$) are coprime.

Integer GCD computations are similarly ill-posed. For example, the integers 54 and 24 have a GCD equal to 6, but it is highly likely that any small perturbations would cause the inexact integers $54 + \epsilon$ and $6 + \epsilon$ to be coprime.

A polynomial GCD finding algorithm which requires exact inputs is not suitable for the problems considered in this thesis, where polynomials are subject to errors. Instead, a method which considers that $f(x)$ and $g(x)$ are close to two polynomials with a non-constant common divisor is required. The UGCD method is developed in the following chapter, which returns an AGCD whose coefficients are close to those of the exact GCD, even in the presence of significant levels of noise.

Noise can come from many sources in computational mathematics, and some of these are now considered:

1. The first source of error comes from round-off in floating-point representations. A floating-point representation consists of a coefficient and an exponent. Therefore, numerical accuracy is traded off against range. A subset of rational numbers have an infinite decimal representation, but must be represented by a truncated and finite decimal form. For example, $\frac{1}{3}$ has an infinite decimal representation and a truncated floating-point representation.

2. The two polynomials whose GCD is to be determined may themselves be the output of some earlier computation, which has potentially introduced computational and round-off errors. For instance, the *ith* GCD computation in the square-free factorisation algorithm (Algorithm 1) computes the GCD of $f_i(x)$ and $f_i^{'}(x)$, where $f_i(x)$ is the $(i-1)$th computed GCD. Since each computation is subject to round-off error it is likely that the cumulated error after many iterations is significant.

3. In curve-curve and surface-surface intersection problems, the polynomial whose square-free factorisation must be computed may be derived from an approximated curve or surface. Or, the process of conversion from a parametric to implicit form may have introduced errors. The implicitisation of bicubic patches produces implicit surfaces of high degree, and it is standard practice to perform an approximate implicitisation which minimises the degree of the implicit form. This approximation is generally computed more quickly than an exact implicitisation, and consequently the polynomial to be factorised will be inexact.

Since noise is seemingly inevitable in this type of problem, it is necessary to consider defining the AGCD, $d_t(x)$, of two inexact polynomials $f(x)$ and $g(x)$, whose exact counterparts $\hat{f}(x)$ and $\hat{g}(x)$ have a GCD $\hat{d}_t(x)$. The AGCD, $d_t(x)$, of two polynomials $f(x)$ and $g(x)$ is defined to be correct when it is sufficiently close to the the GCD, $\hat{d}_t(x)$, of two polynomials $\hat{f}(x)$ and $\hat{g}(x)$ obtained by minimal perturbations of $f(x)$ and $g(x)$. A crucial component of this definition is that the degree of the AGCD is equal to the degree of the GCD $\hat{d}_t(x)$. In [6] a more formal definition of a AGCD is given, which takes into account the machine precision and in [56] the quasi-GCD is defined.

The method of computing the degree of the GCD, described in the following section,

is defined in terms of exact polynomials. However, the same methods are successfully applied to inexact polynomials $f(x)$ and $g(x)$.

3.1.1 The Degree of the GCD by the Subresultant Matrix Methods

This section considers the computation of the degree of the GCD of exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ by using the Sylvester matrix and the sequence of subresultant matrices. A variation on this is used in the computation of the AGCD of two inexact polynomials. It will be shown that computing the degree t of the GCD reduces to determining the numerical rank of each matrix in the set of subresultant matrices $\{ S_k(\hat{f}(x), \hat{g}(x)) \mid k = 1, \ldots, \min(m, n) \}$.

Consider two exact polynomials in Bernstein form, $\hat{f}(x)$ and $\hat{g}(x)$, of degrees m and n respectively which are given by

$$\hat{f}(x) = \sum_{i=0}^{m} \hat{a}_i \binom{m}{i} (1-x)^{m-i} x^i \quad \text{and} \quad \hat{g}(x) = \sum_{i=0}^{n} \hat{b}_i \binom{n}{i} (1-x)^{n-i} x^i \tag{3.1}$$

and whose GCD $\hat{d}_t(x)$ of degree t is given by

$$\hat{d}_t = \sum_{i=0}^{t} d_{t,i} B_i^t(x).$$

There exists a set of common divisors of $\hat{f}(x)$ and $\hat{g}(x)$ denoted $\hat{d}_k(x)$ for $k = 1, \ldots, t^{(\mathrm{i})}$. Note that the GCD of $\hat{f}(x)$ and $\hat{g}(x)$ is unique to within a scalar multiplier, however the common divisors of degree $k = 1, \ldots, t-1$ are not unique.

Example 3.1.1. The two polynomials $\hat{f}(x)$ and $\hat{g}(x)$ given by

$$\hat{f}(x) = (x-1)(x-2)(x-3) \qquad \text{and} \qquad \hat{g}(x) = (x-1)(x-2)(x-4)$$

have a set of common divisors of degree one given by $\{\hat{d}_1(x)\} = \{(x-1), (x-2)\}$ and a unique GCD of degree $t = 2$ given by $\hat{d}_2(x) = (x-1)(x-2)$. □

By Bézout's identity, there also exist cofactor polynomials $\{ \hat{u}_k(x) \mid k = 1, \ldots, t \}$ and $\{ \hat{v}_k(x) \mid k = 1, \ldots, t \}$ of degrees $(m-k)$ and $(n-k)$, which are the cofactors of $\hat{d}_k(x)$, and these are given by

$$\begin{aligned} \hat{u}_k(x) &= \sum_{i=0}^{m-k} \hat{u}_{k,i} \binom{m-k}{i} (1-x)^{m-k-i} x^i \\ \hat{v}_k(x) &= \sum_{i=0}^{n-k} \hat{v}_{k,i} \binom{n-k}{i} (1-x)^{n-k-i} x^i \qquad \text{for} \qquad k = 1, \ldots, t. \end{aligned}$$

Therefore, the statements

$$\hat{f}(x) = \hat{u}_k(x)\hat{d}_k(x) \quad \text{and} \quad \hat{g}(x) = \hat{v}_k(x)\hat{d}_k(x) \tag{3.2}$$

hold for $k = 1, \ldots, t$, and

$$\begin{aligned} \hat{v}_k &\equiv 0 \quad \text{for} \quad k = t+1, \ldots, \min(m,n), \\ \hat{u}_k &\equiv 0 \quad \text{for} \quad k = t+1, \ldots, \min(m,n). \end{aligned}$$

It follows from (3.2) that

$$\hat{f}(x)\hat{v}_k(x) - \hat{g}(x)\hat{u}_k(x) = 0, \quad \text{for} \quad k = 0, \ldots, t,$$

which can be written as the matrix-vector product

$$S_k\left(\hat{f}(x), \hat{g}(x)\right) \begin{bmatrix} \hat{\mathbf{v}}_k \\ -\hat{\mathbf{u}}_k \end{bmatrix} = \mathbf{0}. \tag{3.3}$$

This has non-trivial solutions for $k = 1, \ldots, t$ and the solution vector consists of vectors $\hat{\mathbf{v}}_k \in \mathbb{R}^{n-k+1}$ and $\hat{\mathbf{u}}_k \in \mathbb{R}^{m-k+1}$, which contain the coefficients of cofactor polynomials $\hat{u}_k(x)$ and $\hat{v}_k(x)$, given by

$$\hat{\mathbf{v}}_k = \begin{bmatrix} \hat{v}_{k,0}, & \hat{v}_{k,1}, & \ldots, & \hat{v}_{k,n-k} \end{bmatrix}^T \quad \text{and} \quad \hat{\mathbf{u}}_k = \begin{bmatrix} \hat{u}_{k,0}, & \hat{u}_{k,1}, & \ldots, & \hat{u}_{k,m-k} \end{bmatrix}^T.$$

The matrix $S_k(\hat{f}(x), \hat{g}(x)) \in \mathbb{R}^{(m+n-k+1)\times(m+n-2k+2)}$ in (3.3) is the kth subresultant matrix and is given by

$$S_k(\hat{f}(x), \hat{g}(x)) = \left[\; C_{n-k}\Big(\hat{f}(x)\Big) \;\Big|\; C_{m-k}\Big(\hat{g}(x)\Big) \; \right], \tag{3.4}$$

where $C_{n-k}(\hat{f}(x))$ and $C_{m-k}(\hat{g}(x))$ are the $(n-k)$th and $(m-k)$th order convolution matrices (as described in (2.9)) of $\hat{f}(x)$ and $\hat{g}(x)$ respectively.

Since (3.3) has a non-zero solution for $k = 1, \ldots, t$, the subresultant matrices $\{ S_k(\hat{f}(x), \hat{g}(x)) \mid k = 1, \ldots, t \}$ are singular, while the subresultant matrices $\{ S_k(\hat{f}(x), \hat{g}(x)) \mid k = t+1, \ldots, \min(m,n) \}$ are nonsingular. The computation of the degree t of the GCD $\hat{d}(x)$ is therefore reduced to the determination of the largest k such that $S_k(\hat{f}(x), \hat{g}(x))$ is rank deficient

$$\begin{aligned} \text{rank } S_k\left(\hat{f}(x), \hat{g}(x))\right) &< m+n-2k+2 \quad \text{for } k = 1, \ldots, t, \\ \text{rank } S_k\left(\hat{f}(x), \hat{g}(x)\right) &= m+n-2k+2 \quad \text{for } k = t+1, \ldots, \min(m,n). \end{aligned}$$

The rank of the kth subresultant matrix $S_k(\hat{f}(x), \hat{g}(x))$ for the exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ is unlikely to be equal to the rank of $S_k(f(x), g(x))$, whose entries contain coefficients of inexact polynomials $f(x)$ and $g(x)$. It is probable that each subresultant matrix of the set $\{S_k(f(x), g(x)) \mid k = 1, \ldots, t\}$ is of full rank, since inexact polynomials $f(x)$ and $g(x)$ (perturbed versions of exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$) are typically coprime.

Methods for determining the rank of a subresultant matrix are also inexact due to round-off error in floating-point arithmetic. The theoretical rank loss of the tth subresultant matrix $S_t(\hat{f}(x), \hat{g}(x))$ (where $\hat{f}(x)$ and $\hat{g}(x)$ are known exactly) is one. However, it is highly likely that the computed rank loss is zero, that is, the matrix is of full rank.

It is instead necessary to consider the numerical rank of the set of subresultant matrices

$\{ S_k(f(x), g(x)) \mid k = 1, \ldots, \min(m, n) \}$ when computing the degree of the AGCD of two inexact polynomials.

The next section describes the Sylvester matrix and the matrices of the subresultant matrix sequence (as seen in (3.3)) of two polynomials in Bernstein form. The variants obtained by including or excluding diagonal matrices D^{-1}_{m+n-k} and $\hat{Q}_k$ are also considered.

3.1.2 The Sylvester Matrix and the Subresultant Matrix Sequence

Given the definition of the convolution matrix in (2.9), the kth subresultant matrix (3.4) can be considered as the product of three matrices and is given by

$$S_k\left(\hat{f}(x), \hat{g}(x)\right) = D^{-1}_{m+n-k} T_k\left(\hat{f}(x), \hat{g}(x)\right) \hat{Q}_k. \tag{3.5}$$

The diagonal matrix $D^{-1}_{m+n-k} \in \mathbb{R}^{(m+n-k+1)\times(m+n-k+1)}$ in (3.5) is of the same structure as the matrix D^{-1}_{m+n} defined in (2.10) and is given by

$$D^{-1}_{m+n-k} = \operatorname{diag}\left[\begin{array}{cccc} \frac{1}{\binom{m+n-k}{0}}, & \frac{1}{\binom{m+n-k}{1}}, & \cdots, & \frac{1}{\binom{m+n-k}{m+n-k}} \end{array} \right]. \tag{3.6}$$

The matrix $T_k(\hat{f}(x), \hat{g}(x)) \in \mathbb{R}^{(m+n-k+1)\times(m+n-2k+2)}$ consists of two partitions which are both Toeplitz matrices, and is given by

$$T_k\left(\hat{f}(x), \hat{g}(x)\right) = \left[\begin{array}{c|c} T_{n-k}\left(\hat{f}(x)\right) & T_{m-k}\left(\hat{g}(x)\right) \end{array} \right]$$

$$= \left[\begin{array}{ccc|ccc} \hat{a}_0\binom{m}{0} & & & \hat{b}_0\binom{n}{0} & & \\ \hat{a}_1\binom{m}{1} & \ddots & & \hat{b}_1\binom{n}{1} & \ddots & \\ \vdots & \ddots & \hat{a}_0\binom{m}{0} & \vdots & \ddots & \hat{b}_0\binom{n}{0} \\ \hat{a}_{m-1}\binom{m}{m-1} & & \hat{a}_1\binom{m}{1} & \hat{b}_{n-1}\binom{n}{n-1} & & \hat{b}_1\binom{n}{1} \\ \hat{a}_m\binom{m}{m} & \ddots & \vdots & \hat{b}_n\binom{n}{n} & \ddots & \vdots \\ & \ddots & \hat{a}_{m-1}\binom{m}{m-1} & & \ddots & \hat{b}_{n-1}\binom{n}{n-1} \\ & & \hat{a}_m\binom{m}{m} & & & \hat{b}_n\binom{n}{n} \end{array}\right],$$

where $T_{n-k}(\hat{f}(x)) \in \mathbb{R}^{(m+n-k+1)\times(n-k+1)}$ and $T_{m-k}(\hat{g}(x)) \in \mathbb{R}^{(m+n-k+1)\times(m-k+1)}$ have the same structure as $T_n(\hat{f}(x))$ (defined in (2.11)). The block diagonal matrix $\hat{Q}_k$ in (3.5) of order $(m + n - 2k + 2)$ contains binomial coefficients corresponding to the cofactor polynomials $\hat{v}_k(x)$ and $\hat{u}_k(x)$ and is given by

$$\hat{Q}_k = \operatorname{diag}\left[\begin{array}{cc} Q_{n-k}, & Q_{m-k} \end{array} \right], \tag{3.7}$$

where the matrices $Q_{n-k} \in \mathbb{R}^{(n-k+1)\times(n-k+1)}$ and $Q_{m-k} \in \mathbb{R}^{(m-k+1)\times(m-k+1)}$ are diagonal matrices with the same structure as the matrix Q_n defined in (2.12), and are given by

$$Q_{m-k} = \operatorname{diag}\left[\begin{array}{cccc} \binom{m-k}{0}, & \binom{m-k}{1}, & \ldots, & \binom{m-k}{m-k} \end{array} \right]$$
$$Q_{n-k} = \operatorname{diag}\left[\begin{array}{cccc} \binom{n-k}{0}, & \binom{n-k}{1}, & \ldots, & \binom{n-k}{n-k} \end{array} \right].$$

The Sylvester matrix $S(\hat{f}(x), \hat{g}(x))$ is the first matrix in the subresultant matrix sequence and is also denoted $S_1(\hat{f}(x), \hat{g}(x))$. The matrices of the sequence $S_1(\hat{f}(x), \hat{g}(x))$, $S_2(\hat{f}(x), \hat{g}(x))$, $\ldots$, $S_{min(m,n)}(\hat{f}(x), \hat{g}(x))$ are of decreasing size, where each matrix

$S_k(\hat{f}(x), \hat{g}(x))$ is of dimensions $(m+n-k+1) \times (m+n-2k+2)$, and a method for the computation of the matrices in this sequence is now described.

3.1.3 The Construction of the Subresultant Matrix Sequence

The subresultant matrices of polynomials in the power basis contain the same set of non-zero entries and each subresultant matrix $S_k(\hat{f}(x), \hat{g}(x))$ is easily obtained by the removal of a set of $(k-1)$ rows and $(2k-2)$ columns from $S_1(\hat{f}(x), \hat{g}(x))$. However, the relationship between the two subresultant matrices $S_k(\hat{f}(x), \hat{g}(x))$ and $S_j(\hat{f}(x), \hat{g}(x))$ for polynomials $\hat{f}(x)$ and $\hat{g}(x)$ in Bernstein form is non-trivial. Each of the non-zero entries of $S_k(\hat{f}(x), \hat{g}(x))$ consists of three binomial terms which are dependent on k, so entries of each subresultant matrix in the sequence are distinct.

This section begins by developing the transformation to obtain $S_k(\hat{f}(x), \hat{g}(x))$ given $S_{k-1}(\hat{f}(x), \hat{g}(x))$, and this is extended to the general transformation to compute $S_j(\hat{f}(x), \hat{g}(x))$ from $S_k(\hat{f}(x), \hat{g}(x))$ for any j, where $k < j \leq \min(m, n)$.

The matrix $S_{k+1}(\hat{f}(x), \hat{g}(x))$ is given by

$$S_{k+1}\left(\hat{f}(x), \hat{g}(x)\right) = \mathcal{A}_{m+n-k} S_k\left(\hat{f}(x), \hat{g}(x)\right) \hat{\mathcal{B}}_k,$$

where the matrix $\mathcal{A}_{m+n-k} \in \mathbb{R}^{(m+n-k)\times(m+n-k+1)}$ is given by

$$\left[\begin{array}{ccccc|c} \frac{m+n-k}{m+n-k} & & & & & 0 \\ & \frac{m+n-k}{m+n-k-1} & & & & 0 \\ & & \ddots & & & \vdots \\ & & & \frac{m+n-k}{1} & & 0 \end{array}\right]. \tag{3.8}$$

The block diagonal matrix $\hat{\mathcal{B}}_k \in \mathbb{R}^{(m+n-2k+2)\times(m+n-2k)}$ is given by

$$\hat{\mathcal{B}}_k = \operatorname{diag}\left[\begin{array}{cc} \mathcal{B}_{n-k}, & \mathcal{B}_{m-k} \end{array}\right], \tag{3.9}$$

where the matrices $\mathcal{B}_{n-k} \in \mathbb{R}^{(n-k+1)\times(n-k)}$ and $\mathcal{B}_{m-k} \in \mathbb{R}^{(m-k+1)\times(m-k)}$ are given by

$$\mathcal{B}_{n-k} = \left[\begin{array}{cccc} \frac{n-k}{n-k} & & & \\ & \frac{n-k-1}{n-k} & & \\ & & \ddots & \\ & & & \frac{1}{n-k} \\ \hline 0 & 0 & \ldots & 0 \end{array}\right] \quad \text{and} \quad \mathcal{B}_{m-k} = \left[\begin{array}{cccc} \frac{m-k}{m-k} & & & \\ & \frac{m-k-1}{m-k} & & \\ & & \ddots & \\ & & & \frac{1}{m-k} \\ \hline 0 & 0 & \ldots & 0 \end{array}\right]. \tag{3.10}$$

This transformation is extended such that the $(k+j)$th subresultant matrix $S_{k+j}(\hat{f}(x), \hat{g}(x))$ for $1 \leq j \leq (\min(m, n) - k)$ can be obtained by the transformation of the kth subresultant matrix.

The matrix $S_{k+j}(\hat{f}(x), \hat{g}(x))$ is given by

$$S_{k+j}\left(\hat{f}(x), \hat{g}(x)\right) = \mathcal{A}_{m+n-k-j+1}\mathcal{A}_{m+n-k-j}\ldots\mathcal{A}_{m+n-k}\times \\ S_k\left(\hat{f}(x), \hat{g}(x)\right) \times \hat{\mathcal{B}}_k\hat{\mathcal{B}}_{k+1}\ldots\hat{\mathcal{B}}_{k+j}, \tag{3.11}$$

where the matrices $\mathcal{A}_{m+n-k}, \ldots, \mathcal{A}_{m+n-k-j+1}$ and $\hat{\mathcal{B}}_k, \ldots, \hat{\mathcal{B}}_{k+j}$ are of the same form as the matrices defined in (3.8) and (3.9) respectively.

3.1.4 Variants of the Subresultant Matrices

In Section 3.1.2 the sequence of subresultant matrices was introduced, and this section considers five variants of this sequence of matrices. These variants will be defined for the kth subresultant matrix only, and the definitions are to be extended to all matrices in the relevant sequence. The first four variants arise by including or excluding the diagonal matrices D^{-1}_{m+n-k} and $\hat{Q}_k$ in the definition of the subresultant matrices and are given by:

(i) $\{T_k(\hat{f}(x,y), \hat{g}(x,y))\}$

(ii) $\{D^{-1}_{m+n-k}T_k(\hat{f}(x,y), \hat{g}(x,y))\}$

(iii) $\{T_k(\hat{f}(x,y), \hat{g}(x,y))\hat{Q}_k\}$

(iv) $\{D^{-1}_{m+n-k}T_k(\hat{f}(x,y), \hat{g}(x,y))\hat{Q}_k\}$

The orthogonal matrices D^{-1}_{m+n-k} and $\hat{Q}_k$ pre and post multiply $T_k(\hat{f}(x), \hat{g}(x))$ respectively, and the rank of the subresultant matrices is theoretically unaffected by their inclusion. The fifth variant is constructed by the rearrangement of the variant $D^{-1}_{m+n-k}T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$ such that divisors common to each of the non-zero entries of the two partitions can be removed, which again does not theoretically affect the rank of the subresultant matrix.

Binomial terms in each non-zero entry of the kth subresultant matrix variant may cause the ratio of the entry of maximum magnitude to entry of minimum magnitude to be large. This can cause numerical problems and as a consequence some variants are more suited to further rank analysis than others. Of interest to this work, the singular values of the kth subresultant matrix for each subresultant variant can be significantly different.

The first four variants of the kth subresultant matrix are now defined:

1. The first variant of the kth subresultant matrix is given by

$$T_k\left(\hat{f}(x), \hat{g}(x)\right) = \left[\ T_{n-k}\left(\hat{f}(x)\right)\ \middle|\ T_{m-k}\left(\hat{g}(x)\right)\ \right].$$

Entries of the first partition $T_{n-k}(\hat{f}(x))$ are given by

$$T_{n-k}\left(\hat{f}(x)\right)_{(i+j+1,j+1)} = \begin{cases} \hat{a}_i\binom{m}{i} & i = 0, \ldots, m; \\ 0 & \text{otherwise,} \end{cases}$$

and entries of the second partition $T_{m-k}(\hat{g}(x))$ are of the form

$$T_{m-k}\Big(\hat{g}(x)\Big)_{(i+j+1,j+1)} = \begin{cases} \hat{b}_i\binom{n}{i} & i = 0,\ldots,n; \\ 0 & \text{otherwise.} \end{cases}$$

When the two polynomials $\hat{f}(x)$ and $\hat{g}(x)$ are of significantly different degree, entries of the two partitions $T_{n-k}(\hat{f}(x))$ and $T_{m-k}(\hat{g}(x))$ can be unbalanced due to the binomial terms $\binom{m}{i}$ and $\binom{n}{j}$ as in Example 3.1.2. These binomial terms are referred to as the coefficient multipliers of entries in $T_{n-k}(\hat{f}(x))$ and $T_{m-k}(\hat{g}(x))$.

Example 3.1.2. Consider two polynomials $\hat{f}(x)$ and $\hat{g}(x)$ of degrees $m = 20$ and $n = 5$ respectively. The non-zero entries in the first partition of $T_k(\hat{f}(x), \hat{g}(x))$ are scaled between 1 and $\binom{20}{10} = 184756$, while entries in the second partition are scaled by at most 10. This difference in scaling over the two partitions can cause issues in the computation of the numerical rank of the subresultant matrices.

□

2. The second variant of the kth subresultant matrix includes the diagonal matrix D^{-1}_{m+n-k} and is given by $D^{-1}_{m+n-k}T_k(\hat{f}(x), \hat{g}(x))$. Entries in the first partition $D^{-1}_{m+n-k}T_{n-k}(\hat{f}(x))$ are of the form

$$D^{-1}_{m+n-k}T_{n-k}\left(\hat{f}(x)\right)_{(i+j+1,j+1)} = \begin{cases} \frac{\hat{a}_i\binom{m}{i}}{\binom{m+n-k}{i+j}} & i = 0,\ldots,m;\ j = 0,\ldots,n-k, \\ 0 & \text{otherwise,} \end{cases}$$

and entries in the second partition $D^{-1}_{m+n-k}T_{m-k}(\hat{g}(x))$ are of the form

$$D^{-1}_{m+n-k}T_{m-k}\Big(\hat{g}(x)\Big)_{(i+j+1,j+1)} = \begin{cases} \frac{\hat{b}_i\binom{n}{i}}{\binom{m+n-k}{i+j}} & i = 0,\ldots,n;\ j = 0,\ldots,m-k, \\ 0 & \text{otherwise.} \end{cases}$$

The effect of the coefficient multipliers $\binom{m}{i}/\binom{m+n-k}{i+j}$ and $\binom{n}{i}/\binom{m+n-k}{j}$ is to scale the coefficients by a value between 0 and 1, but it is shown in Figure 3.3i of Example 3.3.1 that the scaling is not optimal and the entries in the middle columns of each partition are disproportionately small.

3. The third variant of the kth subresultant matrix includes the matrix $\hat{Q}_k$ and is given by $T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$. Entries in the first partition $T_{n-k}(\hat{f}(x))Q_{n-k}$ are given by

$$\Big(T_{n-k}\left(\hat{f}(x)\right)Q_{n-k}\Big)_{(i+j+1,j+1)} = \begin{cases} \hat{a}_i\binom{m}{i}\binom{n-k}{j} & i = 0,\ldots,m;\ j = 0,\ldots,n-k, \\ 0 & \text{otherwise,} \end{cases}$$

and entries in the second partition $T_{m-k}(\hat{g}(x))Q_{m-k}$ are given by

$$\Big(T_{m-k}\Big(\hat{g}(x)\Big)Q_{m-k}\Big)_{(i+j+1,j+1)} = \begin{cases} \hat{b}_i\binom{n}{i}\binom{m-k}{j} & i = 0,\ldots,n;\ j = 0,\ldots,m-k, \\ 0 & \text{otherwise.} \end{cases}$$

For large values of m and $n-k$, the product $\binom{m}{i} \times \binom{n-k}{j}$ is large when $i \approx \frac{m}{2}$ and $j \approx \frac{n-k}{2}$. The coefficient multipliers are therefore disproportionately large for entries in the middle columns of $C_{n-k}(f(x))$, particularly around the middle rows. Entries in the second partition $C_{m-k}(\hat{g}(x))$ are similarly scaled, and this effect is seen in Figure 3.3ii.

4. The fourth variant of the kth subresultant matrix is given by $D^{-1}_{m+n-k} T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$, which has already been described in Section 3.1.2. Entries in the first partition of the kth modified subresultant matrix are of the form

$$C_{n-k}\left(\hat{f}(x)\right)_{(i+j+1,j+1)} \begin{cases} \frac{\hat{a}_i\binom{m}{i}\binom{n-k}{j}}{\binom{m+n-k}{i+j}} & i = 0, \ldots, m;\ j = 0, \ldots, n-k, \\ 0 & \text{otherwise,} \end{cases}$$

and entries in the second partition $D^{-1}_{m+n-k} T_{m-k}(g(x))Q_{m-k}$ are given by

$$C_{m-k}\left(\hat{g}(x)\right)_{(i+j+1,j+1)} = \begin{cases} \frac{\hat{b}_i\binom{n}{i}\binom{m-k}{j}}{\binom{m+n-k}{i+j}} & i = 0, \ldots, n;\ j = 0, \ldots, m-k, \\ 0 & \text{otherwise.} \end{cases}$$

The effect of the coefficient multipliers is to scale the coefficients $\hat{a}_i$ for $i = 0, \ldots, m$ and $\hat{b}_i$ for $i = 0, \ldots, n$ by a multiplier between 0 and 1. The scaling achieved by the combination of these three binomial coefficients, shown in Figure 3.3iii, is an optimal form of scaling when compared with the first three variants.

The scaling of the non-zero entries of these four subresultant matrix variants will be further analysed in Section 3.3, and a set of examples will show which of the variants is optimal for the computation of the degree of the GCD. However, a fifth variant, a modified form of $D^{-1}_{m+n-k} T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$, is first considered.

The Fifth Variant of the Subresultant Matrices

Each of the non-zero entries of the kth subresultant matrix given by $D^{-1}_{m+n-k} T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$ contains three binomial coefficients, two of which are functions of k, and this presents two issues:

1. The computation of three possibly large binomial coefficients is necessary for each entry of the kth subresultant matrix $S_k(\hat{f}(x), \hat{g}(x))$, so for the polynomials $\hat{f}(x)$ and $\hat{g}(x)$ of degree m and n, the construction of the subresultant matrix $S_k(\hat{f}(x), \hat{g}(x))$ requires the evaluation of $3 \times [(m+1)(n-k+1) + (n+1)(m-k+1)]$ binomial coefficients.

2. Each of the subresultant matrices $S_k(\hat{f}(x), \hat{g}(x))$ for $k = 1, \ldots, \min(m, n)$ requires the evaluation of a different set of entries, since the (i, j)th entry of $S_k(\hat{f}(x), \hat{g}(x))$ differs from the (i, j)th entry of $S_{k-1}(\hat{f}(x), \hat{g}(x))$.

This section introduces a manipulation of the three binomial coefficients of the entries of $S_k(\hat{f}(x), \hat{g}(x))$. All three of the binomial coefficients for the non-zero entries in $S_k(\hat{f}(x), \hat{g}(x))$ are functions of i or j, and are therefore dependent on their position within

the matrix. It is shown that the entries of the subresultant matrix can be reduced to the product of two binomial terms dependent on i or j in the numerator and a third constant binomial coefficient in the denominator.

The kth subresultant matrix $S_k(\hat{f}(x), \hat{g}(x))$ is given by

$$\left[\begin{array}{ccc|ccc}
\frac{\hat{a}_0\binom{m}{0}\binom{n-k}{0}}{\binom{m+n-k}{0}} & & & \frac{\hat{b}_0\binom{n}{0}\binom{m-k}{0}}{\binom{m+n-k}{0}} & & \\
\frac{\hat{a}_1\binom{m}{1}\binom{n-k}{0}}{\binom{m+n-k}{1}} & \ddots & & \frac{\hat{b}_1\binom{n}{1}\binom{m-k}{0}}{\binom{m+n-k}{1}} & \ddots & \\
\vdots & \ddots & \frac{\hat{a}_0\binom{m}{0}\binom{n-k}{n-k}}{\binom{m+n-k}{n-k}} & \vdots & \ddots & \frac{\hat{b}_0\binom{n}{0}\binom{m-k}{m-k}}{\binom{m+n-k}{m-k}} \\
\vdots & & \frac{\hat{a}_1\binom{m}{1}\binom{n-k}{n-k}}{\binom{m+n-k}{n-k+1}} & \vdots & & \frac{\hat{b}1\binom{n}{1}\binom{m-k}{m-k}}{\binom{m+n-k}{m-k+1}} \\
\frac{\hat{a}_{m-1}\binom{m}{m-1}\binom{n-k}{0}}{\binom{m+n-k}{m-1}} & & \vdots & \frac{\hat{b}_{n-1}\binom{n}{n-1}\binom{m-k}{0}}{\binom{m+n-k}{n-1}} & & \vdots \\
\frac{\hat{a}_m\binom{m}{m}\binom{n-k}{0}}{\binom{m+n-k}{m}} & \ddots & \vdots & \frac{\hat{b}_n\binom{n}{n}\binom{m-k}{0}}{\binom{m+n-k}{n}} & \ddots & \vdots \\
 & \ddots & \frac{\hat{a}_{m-1}\binom{m}{m-1}\binom{n-k}{n-k}}{\binom{m+n-k}{m+n-k-1}} & & \ddots & \frac{\hat{b}_{n-1}\binom{n}{n-1}\binom{m-k}{m-k}}{\binom{m+n-k}{m+n-k-1}} \\
 & & \frac{\hat{a}_m\binom{m}{m}\binom{n-k}{n-k}}{\binom{m+n-k}{m+n-k}} & & & \frac{\hat{b}_n\binom{n}{n}\binom{m-k}{m-k}}{\binom{m+n-k}{m+n-k}}
\end{array}\right].$$

The non-zero entries of $C_{n-k}(\hat{f}(x))$ can by manipulation of the binomial coefficients be rearranged as

$$C_{n-k}\left(\hat{f}(x)\right)_{(i+j+1,j+1)} = \begin{cases} \frac{\hat{a}_i\binom{i+j}{i}\binom{m+n-k-i-j}{m-i}}{\binom{m+n-k}{m}} & i = 0, \ldots, m;\ j = 0, \ldots, n-k, \\ 0 & \text{otherwise,} \end{cases} \tag{3.12}$$

and by a similar rearrangement, entries of $C_{n-k}(\hat{g}(x))$ are given by

$$C_{m-k}\left(\hat{g}(x)\right)_{(i+j+1,j+1)} = \begin{cases} \frac{\hat{b}_i\binom{i+j}{i}\binom{m+n-k-i-j}{n-i}}{\binom{m+n-k}{n}} & i = 0, \ldots, n;\ j = 0, \ldots, m-k, \\ 0 & \text{otherwise.} \end{cases} \tag{3.13}$$

The kth subresultant matrix $S_k(\hat{f}(x), \hat{g}(x)) \in \mathbb{R}^{(m+n-k+1)\times(m+n-2k+2)}$ with rearranged entries is given by

$$\left[\begin{array}{ccc|ccc}
\frac{\hat{a}_0\binom{0}{0}\binom{m+n-k}{m}}{\binom{m+n-k}{m}} & & & \frac{\hat{b}_0\binom{0}{0}\binom{m+n-k}{n}}{\binom{m+n-k}{n}} & & \\
\frac{\hat{a}_1\binom{1}{1}\binom{m+n-k-1}{m-1}}{\binom{m+n-k}{m}} & \ddots & & \frac{\hat{b}_1\binom{1}{1}\binom{m+n-k}{n-1}}{\binom{m+n-k}{n}} & \ddots & \\
\vdots & \ddots & \frac{\hat{a}_0\binom{n-k}{0}\binom{m}{m}}{\binom{m+n-k}{m}} & \vdots & \ddots & \frac{\hat{b}_0\binom{0}{0}\binom{m-k}{0}}{\binom{m+n-k}{n}} \\
\vdots & & \frac{\hat{a}_1\binom{n-k+1}{1}\binom{m-1}{m-1}}{\binom{m+n-k}{m}} & \vdots & & \frac{\hat{b}_1\binom{n-1}{n-1}\binom{m-k+1}{1}}{\binom{m+n-k}{n}} \\
\frac{\hat{a}_{m-1}\binom{m-1}{m-1}\binom{n-k+1}{1}}{\binom{m+n-k}{m}} & & \vdots & \frac{\hat{b}_{n-1}\binom{n-1}{n-1}\binom{m-k+1}{1}}{\binom{m+n-k}{n}} & & \vdots \\
\frac{\hat{a}_m\binom{m}{m}\binom{n-k}{0}}{\binom{m+n-k}{m}} & \ddots & \vdots & \frac{\hat{b}_n\binom{n}{n}\binom{0}{0}}{\binom{m+n-k}{n}} & \ddots & \vdots \\
 & \ddots & \frac{\hat{a}_{m-1}\binom{m+n-k-1}{m-1}\binom{1}{1}}{\binom{m+n-k}{m}} & & \ddots & \frac{\hat{b}_{n-1}\binom{m+n-k-1}{n-1}\binom{1}{1}}{\binom{m+n-k}{n}} \\
 & & \frac{\hat{a}_m\binom{m+n-k}{m}\binom{0}{0}}{\binom{m+n-k}{m}} & & & \frac{\hat{b}_n\binom{m+n-k}{n}\binom{0}{0}}{\binom{m+n-k}{n}}
\end{array}\right],$$

and this rearrangement of the binomial terms has following interesting properties:

1. Consider the non-zero entries in the first partition of the kth subresultant matrix

with rearranged entries. The product of the first binomial coefficient in all of the non-zero entries is equal to the product of the second binomial coefficient in the numerator of all of the non-zero entries, that is,

$$\prod_{j=0}^{n-k}\prod_{i=0}^{m}\binom{i+j}{j} \equiv \prod_{j=0}^{n-k}\prod_{i=0}^{m}\binom{m+n-k-i-j}{n-k-j}.$$

This is of interest since the complexity of computing the geometric mean of non-zero entries in the two partitions of the subresultant matrices is reduced. The computation of the geometric mean is described in Section 3.4.

2. A simple expression for the computation of the arithmetic mean is deduced from the entries of the rearranged kth subresultant matrix. The sum of all non-zero entries in $S_k(\hat{f}(x), \hat{g}(x))$, containing any specified coefficient $\hat{a}_i$, is given by

$$\sum_{j=0}^{n-k}\hat{a}_i\frac{\binom{i+j}{i}\binom{m+n-k-i-j}{m-i}}{\binom{m+n-k}{m}} = \hat{a}_i\frac{\binom{m+n-k+1}{m+1}}{\binom{m+n-k}{m}} = \hat{a}_i\left(\frac{m+n-k+1}{m+1}\right)$$

and a similar expression is derived for entries containing the coefficients $\hat{b}_i$. This significantly reduces the effort required to compute the arithmetic mean of the non-zero entries of $C_{n-k}(\hat{f}(x))$ and $C_{m-k}(\hat{g}(x))$, which will be shown in Section 3.4.

The fifth variant of subresultant matrix is given by the removal of common denominators $\binom{m+n-k}{m}$ and $\binom{m+n-k}{n}$ from the partitions $C_{n-k}(\hat{f}(x))$ and $C_{m-k}(\hat{g}(x))$ of the kth subresultant matrix $S_k(\hat{f}(x), \hat{g}(x))$, and this is equivalent to scaling the polynomials $\hat{f}(x)$ and $\hat{g}(x)$ by the non-zero constants $\frac{1}{\binom{m+n-k}{m}}$ and $\frac{1}{\binom{m+n-k}{n}}$. The GCD of these scaled forms is equivalent to that of the original polynomials, that is,

$$\text{GCD}\left(\hat{f}(x), \hat{g}(x)\right) = GCD\left(\frac{\hat{f}(x)}{\binom{m+n-k}{m}}, \frac{\hat{g}(x)}{\binom{m+n-k}{n}}\right).$$

The subresultant matrix with denominators omitted is denoted by $\breve{S}_k(\hat{f}(x), \hat{g}(x))$

$$\breve{S}_k\left(\hat{f}(x), \hat{g}(x)\right) = \left[\ \breve{C}_{n-k}\left(\hat{f}(x)\right)\ \middle|\ \breve{C}_{m-k}\left(\hat{g}(x)\right)\ \right] \quad \text{for} \quad k = 1, \ldots, \min(m, n),$$

where the matrices $\breve{C}_{n-k}(\hat{f}(x))$ and $\breve{C}_{m-k}(\hat{g}(x))$ are given by

$$\breve{C}_{n-k}\left(\hat{f}(x)\right) = \binom{m+n-k}{n-k} \times C_{n-k}\left(\hat{f}(x)\right)$$
$$\breve{C}_{m-k}\left(\hat{g}(x)\right) = \binom{m+n-k}{m-k} \times C_{m-k}\left(\hat{g}(x)\right).$$

The rank of this new form of subresultant matrix $\breve{S}_k(\hat{f}(x), \hat{g}(x))$ is theoretically equal to the rank of $S_k(\hat{f}(x), \hat{g}(x))$

$$\text{rank}\left(\breve{S}_k\left(\hat{f}(x), \hat{g}(x)\right)\right) = \text{rank}\left(S_k\left(\hat{f}(x), \hat{g}(x)\right)\right).$$

Five variants of the subresultant matrix sequence have been described, and the scaling

of their entries has been considered. Poor scaling within the subresultant matrices is known to give bad results when attempting to determine the degree or coefficients of the GCD. The variant denoted $D_{m+n-k}^{-1}T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$ has optimal scaling amongst its entries, and numerical examples will show this in Section 3.3.

The rearrangement of the entries in the fourth variant of subresultant matrix reveals some interesting properties of these matrices. A common divisor to all non-zero entries in each partition is found, and properties which allow for fast methods of the computation of the arithmetic and geometric means have been described.

A set of examples in Section 3.3 will determine which of the five variants of subresultant matrix is optimal for computing the degree of the GCD. First, several methods for the computation of the degree of the GCD using subresultant matrices are described.

3.2 Methods for the Computation of the Degree of the GCD

It has already been stated that the computation of the degree of the GCD of two exact polynomials, or the AGCD of two inexact polynomials, reduces to the determination of the last singular or numerically singular subresultant matrix in the respective subresultant matrix sequences.

Alternatively, the degree of the GCD is given by the rank loss of $S_1(\hat{f}(x), \hat{g}(x))$ or the numerical rank loss of $S_1(f(x), g(x))$, but this method is typically weaker than considering all subresultant matrices.

This section describes several methods for the computation of the degree of the GCD or AGCD of two univariate polynomials f and g, which may be exactly or inexactly defined.

3.2.1 The Degree of the GCD by Singular Values

This section considers two methods for the computation of the degree of the GCD by methods which make use of the SVD of the Sylvester matrix and the set of subresultant matrices. Again, polynomials f and g of degrees m and n respectively have a GCD of degree t.

Degree Computation 1 (DC1) : The degree computation 1 (DC1) method for computing the degree of the GCD utilises the singular values of the first subresultant matrix, that is, the set $\{\sigma_{1,i} \mid i = 1, \ldots, m+n\}$. A method which uses the singular values of the Sylvester matrix was described in Section 1.8.1, but the method described here does not require any thresholds or prior knowledge of the noise added to the coefficients of f and g. Instead, a heuristic algorithm is used.

For exact polynomials, using infinite precision arithmetic, the Sylvester matrix has the rank $m+n-t$, however, in a floating point environment, the Sylvester matrix is typically of full rank, and has numerical rank $m+n-t$. The numerical rank, as defined in this thesis, is defined based on a *large* separation between numerically zero and non-zero singular values. This method is disregards the level of noise or machine precision and is based on observations only.

Since the singular values are in descending order, the first $(m+n-t)$ singular values $\{\sigma_{1,i} \mid i = 1, \ldots, (m+n-t)\}$ are non-zero, while the remaining t singular values

$\{\sigma_{1,i} \mid i = (m+n-t+1), \ldots, (m+n)\}$ are numerically zero.

Since the magnitude of the minimum singular values are of interest, let $\check{\rho}_i$ be the log of the singular value $\sigma_{1,i}$

$$\check{\rho}_i = \log_{10}(\sigma_{1,i}) \quad \text{for} \quad i = 1, \ldots, \min(m,n).$$

With the singular values ordered in decreasing size, a *large negative change* in magnitude from $\sigma_{1,i}$ to $\sigma_{1,i+1}$ indicates that $\sigma_{1,i}$ is non-zero while $\sigma_{1,i+1}$ is numerically zero. Let $\delta\check{\rho}_i$ denote the magnitude of change between $\sigma_{1,i}$ and $\sigma_{1,i+1}$ given by

$$\begin{aligned}\delta\check{\rho}_i &= \check{\rho}_i - \check{\rho}_{i+1} \\ &= \log_{10}(\sigma_{1,i}) - \log_{10}(\sigma_{1,i+1}) \\ &= \log_{10}\left(\frac{\sigma_{1,i}}{\sigma_{1,i+1}}\right) \quad \text{for} \quad i - 1, \ldots, (\min(m,n)-1)\end{aligned} \tag{3.14}$$

In this particular set of problems it is unlikely that any $\sigma_{1,i}$ is exactly equal to zero, since computations are performed in a floating-point environment and the polynomials f and g are inexact. Instead, a *large negative change* is expected between $\check{\rho}_{m+n-t}$ (a non-zero value) and $\check{\rho}_{m+n-t+1}$ (a numerically zero value). Therefore, $\delta\check{\rho}_i$ in (3.14) is *large and positive* for $i = m+n-t$. The numerical rank of the Sylvester matrix is given by

$$\begin{aligned}m+n-t &= \arg_i \max\{\delta\check{\rho}_i \mid i = 1, \ldots, (\min(m,n)-1)\} \\ &= \arg_i \max\left\{\log_{10}\left(\frac{\sigma_i}{\sigma_{i+1}}\right)\right\}\end{aligned}$$

and so t, the degree of the GCD, is given by

$$t = m + n - \arg_i \max\left\{\log_{10}\left(\frac{\sigma_i}{\sigma_{i+1}}\right)\right\}.$$

While this heuristic method works for many examples, it is possible that a maximum $\delta\check{\rho}_i$ is used to incorrectly identify the degree of the GCD. The term *large* is used within the context of all other $\delta\check{\rho}_i$, and while one of these values is necessarily larger than the others, it may be the case that its significance is exaggerated, and the degree of the GCD is incorrectly determined. In such cases it would be necessary to define some threshold to determine whether $\max\{\delta\check{\rho}_i\}$ is significant. These exceptions are described in more detail in Section 3.2.5.

Degree Computation 2 (DC2) : The degree computation 2 (DC2) method utilises the set of minimum singular values $\{\dot{\sigma}_k \mid k = 1, \ldots, \min(m,n)\}$ of the sequence of subresultant matrices $\{S_k(f,g) \mid k = 1, \ldots, \min(m,n)\}$, where $\dot{\sigma}_k$ is the minimum singular value of the kth subresultant matrix $S_k(f,g)$. Again, the magnitude of these values is of interest so let $\dot{\rho}_k$ be defined as

$$\dot{\rho}_k = \log_{10}(\dot{\sigma}_k) \quad \text{for} \quad k = 1, \ldots, \min(m,n) \tag{3.15}$$

and let $\delta\dot{\rho}_i$ denote the change in magnitude between $\dot{\sigma}_i$ and $\dot{\sigma}_{i+1}$ given by

$$\begin{aligned}\delta\dot{\rho}_i &= \dot{\rho}_{i+1} - \dot{\rho}_i \\ &= \log_{10}(\dot{\rho}_{i+1}) - \log_{10}(\dot{\rho}_i) \\ &= \log_{10}\left(\frac{\dot{\sigma}_{i+1}}{\dot{\sigma}_i}\right). \end{aligned} \tag{3.16}$$

The kth subresultant matrix $S_k(f,g)$ is numerically rank deficient for $k = 1,\ldots,t$, or of full rank for $k = (t+1),\ldots,\min(m,n)$. Therefore, the minimum singular values $\dot{\sigma}_k$ in the set $\{\dot{\sigma}_k \mid k = 1,\ldots,t\}$ are numerically zero, while the minimum singular values in the set $\{\,\dot{\sigma}_k \mid k = (t+1),\ldots,\min(m,n)\,\}$ are non-zero. A large positive change between $\dot{\rho}_t$ and $\dot{\rho}_{t+1}$ is expected, therefore $\delta\dot{\rho}_t$ is expected to be large and positive.

The degree of the GCD is therefore given by

$$\begin{aligned} t &= \arg_i \max\{\delta\dot{\rho}_i\} \\ &= \arg_i \max\{\dot{\rho}_{i+1} - \dot{\rho}_i\} \\ &= \arg_i \max\left\{\log_{10}\left(\frac{\dot{\sigma}_i}{\dot{\sigma}_{i+1}}\right)\right\}. \end{aligned} \tag{3.17}$$

3.2.2 The Degree of the GCD by QR Decomposition

The computation of the degree of the GCD by SVD is computationally expensive for polynomials in both the power basis and Bernstein basis. Instead, a method which utilises the QR decomposition can be used to similar effect, but with the advantage that updating and downdating can be applied for polynomials in the power basis. This does not extend to polynomials in the Bernstein basis, however this discussion is saved for Section 3.2.4.

The QR decomposition of the kth subresultant matrix is given by

$$S_k(f,g) = Q_k \begin{bmatrix} R_{1,k} \\ 0_{k-1} \end{bmatrix},$$

where the matrix Q_k[ii] $\in \mathbb{R}^{(m+n-k+1)\times(m+n-k+1)}$ is orthogonal, the matrix $R_{1,k} \in \mathbb{R}^{(m+n-k+1)\times(m+n-k+1)}$ is upper triangular and $0_{k-1} \in \mathbb{R}^{(k-1)\times(m+n-k+1)}$ is a zero matrix.

Since the rank of $S_k(f,g)$ is equal to the rank of $R_{1,k}$, two methods for the computation of the degree of the GCD can be derived from the entries of the upper triangular matrix $R_{1,k}$. Firstly, if a diagonal entry of $R_{1,k}$ is zero, then the matrix is rank deficient. However, in the context of floating-point computations and inexact polynomials it must be determined whether a value $0+\epsilon$ (for some minor perturbation ϵ) is numerically zero or non-zero. Secondly, a zero-row in $R_{1,k}$ is indicative of rank deficiency in both $R_{1,k}$ and $S_k(f,g)$. Again, it must be determined whether the rows of $R_{1,k}$ are numerically zero, in which case $R_{1,k}$ is rank deficient, or non-zero, in which case $R_{1,k}$ is of full rank.

[ii] Q_k is not to be confused with $\hat{Q}_k$ used in the definition of the kth subresultant matrix $D_{m+n-k}^{-1}T_k(f,g)\hat{Q}_k$ or the diagonal matrix Q_n used in polynomial convolution.

The two methods for the computation of the degree of the GCD derived from the QR decomposition are now described:

Degree Computation 3 (DC3) : The degree computation 3 (DC3) method makes use of the diagonal entries of $R_{1,k}$. The ratio of the maximum diagonal entry of $R_{1,k}$ to the minimum diagonal entry of R_k is given by

$$\tilde{\rho}_k = \log_{10}\left(\frac{\max\{\,|R_{1,k}(i,i)| \;\mid\; i = 1,\ldots,m+n-2k+2\,\}}{\min\{\,|R_{1,k}(i,i)| \;\mid\; i = 1,\ldots,m+n-2k+2\,\}}\right) \quad \text{for } k = 1,\ldots,\min(m,n).$$

The value $\tilde{\rho}_k$ is finite when $R_{1,k}$ is nonsingular, since $\min\{\,R_{1,k}(i,i) \mid i = 1,\ldots,m+n-k+1\}$ is non-zero. However, when $R_{1,k}$ is singular, $\tilde{\rho}_k$ is infinite, since $\min\{\,R_{1,k}(i,i) \mid i = 1,\ldots,m+n-k+1\,\} = 0$. When the polynomials f and g are inexact and the QR decomposition is computed in a floating-point environment, it can be assumed that none of the matrices $R_{1,k}$ are exactly singular. Instead, a large value $\tilde{\rho}_k$ is indicative of $R_{1,k}$ being numerically singular and the two polynomials having a common divisor of degree k. When $\tilde{\rho}_k$ is small then $R_{1,k}$ is nonsingular.

Let

$$\delta\tilde{\rho}_i = \tilde{\rho}_i - \tilde{\rho}_{i+1} \qquad \text{for} \quad i = 1,\ldots,\min(m,n)-1.$$

The maximum positive change occurs between $\tilde{\rho}_t$ and $\tilde{\rho}_{t+1}$ since $\tilde{\rho}_t$ is an infinite (or very large) value and $\tilde{\rho}_{t+1}$ is a finite (or significantly smaller) value. The degree of the GCD is therefore given by

$$\begin{aligned} t &= \arg_i \max\{\delta\tilde{\rho}_i\} \\ &= \arg_i \max\{\tilde{\rho}_i - \tilde{\rho}_{i+1}\}. \end{aligned}$$

Degree Computation 4 (DC4) : The degree computation 4 (DC4) method considers the norms of the rows of $R_{1,k}$ in the computation of the degree of the GCD. Let $R_{1,k}(i,:)$ denote the set of entries of the ith row of the matrix $R_{1,k}$. The ratio $\hat{\rho}_k$ is defined as the ratio of the maximum 2-norm to the minimum 2-norm of the rows $R_{1,k}(i,:)$, that is, $\hat{\rho}_k$ is given by

$$\hat{\rho}_k = \log_{10}\left(\frac{\max\{\,\|R_{1,k}(i,:)\|_2 \;\mid\; i = 1,\ldots,m+n-2k+2\}}{\min\{\,\|R_{1,k}(i,:)\|_2 \;\mid\; i = 1,\ldots,m+n-2k+2\,\}}\right), \quad \text{for } k = 1,\ldots,\min(m,n).$$

If a row of $R_{1,k}$ is numerically zero, then $\hat{\rho}_k$ is large since

$$\lim_{x\to\infty} \log_{10}(x) = \infty.$$

Therefore, the matrix $R_{1,k}$ is classified as numerically singular if $\hat{\rho}_k$ is large and nonsingular if $\hat{\rho}_k$ is significantly smaller than infinity. Let

$$\delta\hat{\rho}_i = \hat{\rho}_i - \hat{\rho}_{i+1} \qquad \text{for} \quad i = 1,\ldots,\min(m,n)-1.$$

It follows that t is given by

$$\begin{aligned} t &= \arg_i \max\{\delta\hat{\rho}_i\} \\ &= \arg_i \max\{\hat{\rho}_i - \hat{\rho}_{i+1}\}. \end{aligned}$$

For two polynomials in the power basis, the QR decomposition of each of the subresultant matrices is obtained in a more computationally efficient manner than the singular values by SVD. The QR decomposition of the first subresultant matrix is computed at a cost of n^3, and the QR decomposition of each subsequent subresultant matrix only requires two QR downdate operations at a cost of n^2, since each subsequent subresultant matrix is given by the removal of two columns and one row from the previous matrix. However, this does not extend to subresultant matrices of polynomials in Bernstein form since the entries in $D_{m+n-k}^{-1} T_k(f,g)\hat{Q}_k$ are different to the entries found in $D_{m+n-k-1}^{-1} T_{k+1}(f,g)\hat{Q}_{k+1}$.

3.2.3 The Degree of the GCD by Residuals

It has already been stated that the computation of the degree of the AGCD reduces to the determination of the index of the last numerically rank deficient subresultant matrix. When a subresultant matrix is rank deficient there is at least one column which lies in the space spanned by the remaining columns, and when a matrix is numerically rank deficient a column lies in the space spanned by the remaining columns with a minimal residual. Since this work deals with inexact polynomials defined in a floating-point environment, it is assumed that the subresultant matrices are never rank-deficient, but a subset are numerically rank deficient.

If the kth subresultant matrix $S_k(f,g)$ is numerically rank deficient, a column $\mathbf{c}_{k,q}$ of $S_k(f,g)$ lies in the space spanned by the remaining columns and the subresultant matrix with the qth column removed is denoted $A_{k,q}(f,g)$.

Degree Computation 5 (DC5) : The degree computation 5 (DC5) method uses a set of residuals associated with the subresultant matrices to determine whether they are singular or nonsingular. The residuals $\{\ r_{k,i}\ \mid\ i = 1,\ldots,m+n-2k+2\ \}$ are computed by the least squares solution of the approximate linear algebraic equations

$$A_{k,i}\,(f,g)\,\mathbf{x}_{k,i} \approx \mathbf{c}_{k,i} \quad \text{for} \quad i = 0,\ldots,m+n-2k+1$$

and are given by

$$r_{k,i} = \|\mathbf{c}_{k,i} - A_{k,i}\,(f,g)\,\mathbf{x}_{k,i}\| \quad \text{for} \quad i = 0,\ldots,m+n-2k+1. \tag{3.18}$$

An efficient method for the computation of $\mathbf{x}_{k,i}$ utilises the QR decomposition of $S_k(f,g)$, which can be updated for the removal of each column $\mathbf{c}_{k,i}$.

The minimum residual $\tilde{r}_k$ associated with the kth subresultant matrix is given by

$$\tilde{r}_k = \min\{r_{k,i} \mid i = 0,\ldots,m+n-2k+1\} \quad k = 1,\ldots,\min(m,n). \tag{3.19}$$

If $\tilde{r}_k$ is large, then the columns of $S_k(f,g)$ are linearly independent and $S_k(f,g)$ is of full rank (nonsingular). Alternatively, if $\tilde{r}_k$ is numerically zero, then there is a column in $S_k(f,g)$ which is nearly linearly dependent on the remaining columns and $S_k(f,g)$ is numerically singular.

Let $\breve{\rho}_k$ denote the log of the minimum residual $\tilde{r}_k$

$$\breve{\rho}_k = \log_{10}(\tilde{r}_k) \qquad \text{for} \quad k = 1, \ldots, \min(m,n)$$

and let

$$\begin{aligned}
\delta\breve{\rho}_i &= \breve{\rho}_{i+1} - \breve{\rho}_i \\
&= \log_{10}(\tilde{r}_{i+1}) - \log_{10}(\tilde{r}_i) \\
&= \log_{10}\left(\frac{\tilde{r}_{i+1}}{\tilde{r}_i}\right) \qquad \text{for} \quad k = 1, \ldots, \min(m,n) - 1,
\end{aligned}$$

then the degree of the GCD is given by

$$\begin{aligned}
t &= \arg_i \max\{\delta\breve{\rho}_i\} \\
&= \arg_i \max\{\breve{\rho}_{i+1} - \breve{\rho}_i\} \\
&= \arg_i \max\{\log_{10}(\tilde{r}_{i+1}) - \log_{10}(\tilde{r}_i)\} \\
&= \arg_i \max\left\{\frac{\tilde{r}_{i+1}}{\tilde{r}_i} \mid i = 1, \ldots, \min(m,n) - 1\right\}. \qquad (3.20)
\end{aligned}$$

That is, t is equal to the value of k for which the change between $\tilde{r}_k$ and $\tilde{r}_{k+1}$ is maximal, as this indicates that $S_k(f,g)$ is numerically singular while $S_{k+1}(f,g)$ is nonsingular.

Several methods for the computation of the degree of the GCD of two univariate polynomials have been considered. These measures are based on the SVD and the QR decomposition of the Sylvester matrix or the set of subresultant matrices. Examples will show that methods which make use of the complete set of subresultant matrices give better results than methods which only consider the Sylvester matrix. However, these methods are typically more computationally expensive.

It was stated that methods which make use of the QR decomposition are faster than computing the SVD for each subresultant matrix. This is due to the ability to update a QR decomposition to take into account row and column removals. It will now be shown why QR updating is not applicable to the sequence of subresultant matrices of polynomials in Bernstein form.

3.2.4 The QR Decomposition of the Sequence of Subresultant Matrices

Methods for computing the degree of the GCD based on the QR decomposition are preferred over methods based on SVD. This is because the QR decomposition can be updated to account for row and column deletion, whereas the SVD must be computed from scratch. Many algorithms for QR decomposition exist, such as Gram-Schmidt, Householder and Givens. These algorithms are described in detail in [36, Chapter 5.2]. The work in this

Algorithm 3: The degree of an AGCD of two polynomials using the method of residuals

Input: Two polynomials $f(x)$ and $g(x)$

begin
  for $k = 1, \ldots, \min(m, n)$ *% for each subresultant* $S_k(f, g)$ **do**
    Perform any preprocessing on the subresultant matrix
    for $j = 1, \ldots, m + n - 2k + 2$ *% for each column* j *of the subresultant* $S_k(f, g)$ **do**
      Define the removed column $\mathbf{c}_{k,j} \leftarrow S_k(f, g) \times M_j$
      Define the matrix of the remaining columns of $S_k(f, g)$ as $A_{k,j}(f, g)$
      Calculate the residual $r_{k,j}$ by the least squares solution of (3.18)
    end
    Get minimum residual $\tilde{r}_k$ of the set $\{r_{k,j}\}$
  end
  Using the set of minimal residuals $\{\tilde{r}_k\}$ compute the degree of the GCD
end

Output: t the degree of the AGCD of $f(x)$ and $g(x)$

thesis uses the standard MATLAB `qr()` function which makes use of Householder transformations.

The QR decomposition of $S_k(\hat{f}(x), \hat{g}(x))$ can be updated for row and column deletion, and this is typically faster than computing the QR decomposition from scratch. This is advantageous when considering the QR decomposition of the sequence of subresultant matrices containing coefficients of two polynomials in the power basis, and where $S_{k+1}(\hat{f}(x), \hat{g}(x))$ is given by the removal of two columns and one row from $S_k(\hat{f}(x), \hat{g}(x))$. The entries of the subresultant matrix $S_k(\hat{f}(x), \hat{g}(x))$ for two polynomials in Bernstein form differ from the entries of $S_{k+j}(\hat{f}(x), \hat{g}(x))$, and it was described in Section 3.1.3 how the matrix $S_{k+1}(\hat{f}(x), \hat{g}(x))$ can be obtained by a transformation of $S_k(\hat{f}(x), \hat{g}(x))$.

The QR decomposition of $S_{k+1}(\hat{f}(x), \hat{g}(x))$ reduces to the QR decomposition of

$$Q_{S_{k+1}} R_{S_{k+1}} = \mathcal{A}_{m+n-k} S_k \left(\hat{f}(x), \hat{g}(x) \right) \hat{\mathcal{B}}_k, \tag{3.21}$$

where $\mathcal{A}_{m+n-k}$ and $\hat{\mathcal{B}}_k$ are defined in Section 3.1.3. Given the QR decomposition of $S_k(\hat{f}(x), \hat{g}(x)) = Q_{S_k} R_{S_k}$, the QR decomposition of (3.21) should be simpler to compute than the QR decomposition from scratch.

Let the matrix $\mathcal{A}_{m+n-k}$ be written as

$$\begin{aligned} \mathcal{A}_{m+n-k} &= \mathcal{A}^*_{m+n-k} \mathcal{G}_A \\ &= \begin{bmatrix} \frac{m+n-k}{m+n-k} & & & \\ & \frac{m+n-k}{m+n-k-1} & & \\ & & \ddots & \\ & & & m+n-k \end{bmatrix} \left[\begin{array}{c|c} I_{m+n-k,m+n-k} & 0_{m+n-k,1} \end{array} \right], \end{aligned}$$

where $\mathcal{A}^*_{m+n-k} \in \mathbb{R}^{(m+n-k)\times(m+n-k)}$ is given by

$$\mathcal{A}^*_{m+n-k} = \operatorname{diag} \left[\begin{array}{cccc} \frac{m+n-k}{m+n-k}, & \frac{m+n-k}{m+n-k-1}, & \ldots, & \frac{m+n-k}{1} \end{array} \right].$$

Let $\hat{\mathcal{B}}_k$ be written as

$$\hat{\mathcal{B}}_k = \mathcal{J}_B \mathcal{B}_k^*$$

$$\begin{bmatrix} \mathcal{B}_{n-k} & \\ & \mathcal{B}_{m-k} \end{bmatrix} = \left[\begin{array}{c|c} I_{n-k,n-k} & 0_{n-k,m-k} \\ 0_{1,n-k} & 0_{1,m-k} \\ \hline 0_{m-k,n-k} & I_{m-k,m-k} \\ 0_{1,n-k} & 0_{1,m-k} \end{array}\right] \begin{bmatrix} B_{n-k}^* & \\ & B_{m-k}^* \end{bmatrix},$$

where $\mathcal{B}_{m-k}^* \in \mathbb{R}^{(m-k)\times(m-k)}$ and $\mathcal{B}_{n-k}^* \in \mathbb{R}^{(n-k)\times(n-k)}$ are given by

$$\mathcal{B}_{m-k}^* = \operatorname{diag}\left[\begin{array}{cccc} \frac{m-k}{m-k}, & \frac{m-k-1}{m-k}, & \ldots, & \frac{1}{m-k} \end{array}\right],$$
$$\mathcal{B}_{n-k}^* = \operatorname{diag}\left[\begin{array}{cccc} \frac{n-k}{n-k}, & \frac{n-k-1}{n-k}, & \ldots, & \frac{1}{n-k} \end{array}\right].$$

Then (3.21) is written as

$$\mathcal{A}_{m+n-k}^* \mathcal{J}_A Q_{S_k} R_{S_k} \mathcal{J}_B \begin{bmatrix} \mathcal{B}_{n-k}^* & \\ & \mathcal{B}_{m-k}^* \end{bmatrix}. \tag{3.22}$$

$\mathcal{J}_A$ and $\mathcal{J}_B$ have the effect of removing one row and two columns from $S_k(\hat{f}(x), \hat{g}(x))$, so the QR decomposition of $\mathcal{J}_A Q_{S_k} R_{S_k} \mathcal{J}_B$ can be obtained by QR update and the QR decomposition with updates is defined as

$$Q_2 R_2 = \left(\mathcal{J}_A Q_{S_k} R_{S_k} \mathcal{J}_B\right).$$

The expression (3.22) is therefore reduced to

$$\mathcal{A}_{m+n-k}^* Q_2 R_2 \mathcal{B}_k^*.$$

Since B_k^* is orthogonal, the matrix product $Q_2 R_2 B_k^*$ is given by $Q_2 R_3$, where R_3 is upper triangular and the ith column of R_3 is given by the ith column of R_2 multiplied by the ith diagonal entry of $\mathcal{B}_k^*$

$$\mathcal{A}_{m+n-k}^* Q_2 R_3.$$

This is almost in the form of a QR decomposition. However, the pre-multiplication by A_{m+n-k}^* must be considered and the QR decomposition of $A_{m+n-k}^* Q_2 R_3$ must be computed from scratch. This proves that the problem of determining the ranks of the subresultant matrices of two polynomials in Bernstein form is much more involved than the equivalent problem for two polynomials in the power basis.

3.2.5 Exceptions to the Method of Computing the Degree of the AGCD

The degree of the GCD of two polynomials can be computed by the five methods DC1 - DC5. Four of these methods, DC2 - DC5, are similar, in that each computes the degree

of the GCD as

$$t = \arg_i \max\{\, \delta\rho_i \mid i = 1, \ldots, \min(m,n) - 1 \,\}, \tag{3.23}$$

where $\delta\rho_i$ is defined for each of the methods. Under the following two conditions under which the degree of the GCD cannot be computed by these methods and it is necessary to employ alternative methods:

1. When the two polynomials f and g are coprime, the degree t of the GCD is equal to zero. However, the computed value of t given by $\arg_i \max\{\delta\rho_i\}$ and this is limited to the range $1, \ldots, \min(m,n) - 1$.

   In the square-free factorisation algorithm (Algorithm 1), the GCD of a polynomial and its derivatives is required. Only when f_i is square-free are the polynomials f_i and $f_i^{'}$ coprime, at which point the algorithm terminates.

2. The second condition under which the described methods fail to compute the degree of the GCD occurs when the GCD, $\hat{d}(x)$, is equal to one of the input polynomials to within a scalar multiplier, and $t = \min(m,n)$.

   This can occur in the square-free factorisation algorithm when f_i of degree m_i has only one root of multiplicity m_i. The GCD of f_i and its derivative $f_i^{'}$is equal to $f^{'}$.

These two situations can often be anticipated in the square-free factorisation algorithm, since bounds on the degree of each GCD can be determined before each computation. This will be described in more detail in Chapter 4.

For the computation of the GCD of two arbitrary polynomials, a threshold ϕ could instead be used such that

$$\begin{aligned} &\text{if} && \max_{k=1,\ldots,\min(m,n)-1} \{\delta\rho_k\} < \phi \\ &\text{then} && t = 0 \quad \text{or} \quad t = \min(m,n). \end{aligned}$$

It must then be determined whether all subresultant matrices $\{S_k(f,g)\}$ are singular ($t = 0$) or nonsingular ($t = \min(m,n)$), and this would require a threshold τ such that t is given by

$$t = \begin{cases} 0 & \text{If } \max\{\rho_k\} < \tau \\ \min(m,n) & \text{otherwise.} \end{cases}$$

It is possible that the degree t of the GCD cannot be determined by any metric described in this section, even with polynomials in preprocessed form. The set of values $\{\rho_k \mid k = 1, \ldots, \min(m,n)\}$ may span several orders of magnitude, with the first, ρ_1, indicative of a singular subresultant matrix and the last, $\rho_{\min(m,n)}$, indicative of a nonsingular matrix. If there is no significant change between any two values ρ_k and ρ_{k+1} then it is not possible to determine if a subset of the subresultant matrices are singular. In these cases, it may be necessary to consider a threshold τ, such that if $\rho_k > \tau$, then $S_k(f,g)$ is nonsingular.

The methods described above are all derivative of either the QR decomposition or the SVD of the set of subresultant matrices. In the following example, the singular values $\{\sigma_{k,i} \mid k = 1, \ldots, \min(m,n);\ i = 1, \ldots, m+n-2k+2\}$ and the diagonal entries $\{R_{1,k}(i,i) \mid k = 1, \ldots, \min(m,n);\ i = 1, \ldots, m+n-2k+2\}$ will be considered. It will be shown how both methods generally give the same result, but a different pattern amongst the two sets of values is observed.

Example 3.2.1. Consider the Bernstein form of the polynomials $\hat{f}(x)$ and $\hat{g}(x)$ of degrees $m = 39$ and $n = 42$ respectively, whose factorisations are given by

$$\begin{aligned}\hat{f}(x) &= (x - 5.8265747784)^7(x - 1.2435487954)^5(x - 0.85487987)(x - 0.26)^2 \\ &\quad (x - 0.217612343)^3(x - 0.157981)^9(x + 0.27564657)^5(x + 1.56)^7 \\ \hat{g}(x) &= (x - 1.2435487954)^5(x - 0.99102445)^5(x - 0.4579879)^9(x - 0.217612343)^3 \\ &\quad (x - 0.157981)^9(x + 0.12)^4(x + 1.56)^7.\end{aligned}$$

The GCD of $\hat{f}(x)$ and $\hat{g}(x)$, $\hat{d}_t(x)$ of degree $t = 24$, has a factorised form given by

$$\hat{d}_t(x) = (x - 1.2435487954)^5(x + 1.56)^7(x - 0.217612343)^3(x - 0.157981)^9.$$

Noise is added to the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$ such that the coefficients of the inexact polynomials $f(x)$ and $g(x)$ are given by

$$a_i = \hat{a}_i + r_{f,i}\hat{a}_i\epsilon_{f,i} \quad \text{and} \quad b_j = \hat{b}_j + r_{g,j}\hat{b}_j\epsilon_{g,j}, \tag{3.24}$$

where $\{\, r_{f,i} \mid i = 0, \ldots, m \,\}$ and $\{\, r_{g,j} \mid j = 0, \ldots, n \,\}$ are uniformly distributed random variables in the interval $[-1, 1]$, and $\{\, \epsilon_{f,i} \mid i = 0, \ldots, m \,\}$ and $\{\, \epsilon_{g,j} \mid j = 0, \ldots, n \,\}$ are uniformly distributed random variables in the interval $\left[10^{-12}, 10^{-10}\right]$. The polynomials $f(x)$ and $g(x)$ are preprocessed by methods which are described in a later section. The coefficients of the unprocessed polynomials $f(x)$ and $g(x)$, and the preprocessed polynomials $\tilde{f}_1(\omega)$ and $\tilde{g}_1(\omega)$ are plotted in Figure 3.1. The singular values $\{\sigma_{k,i}\}$ of the preprocessed

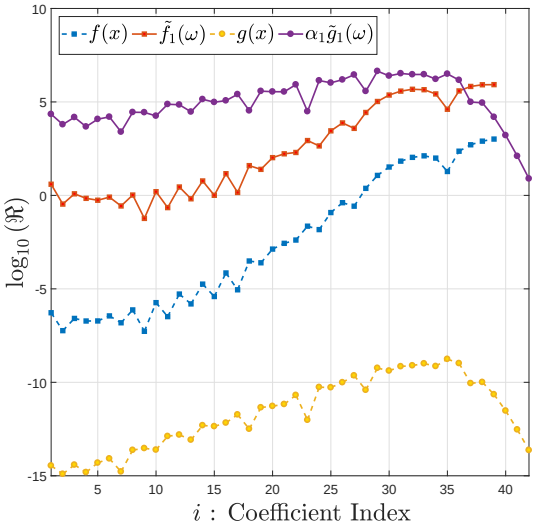
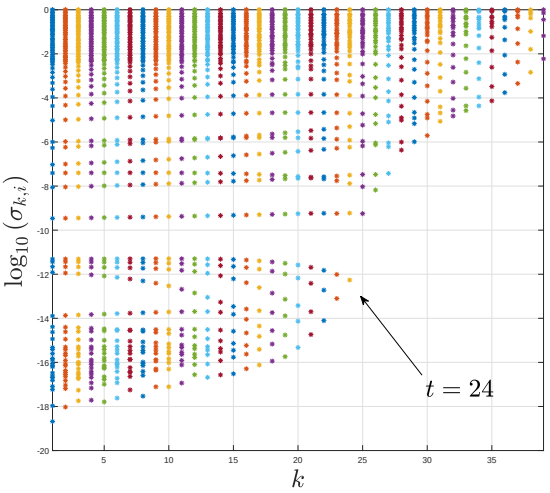
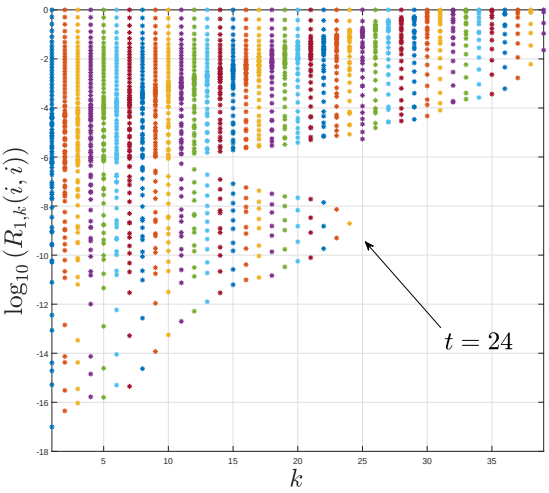


Figure 3.1: The coefficients of both the unprocessed polynomials $f(x)$ (■) and $g(x)$ (●) and the preprocessed polynomials $\tilde{f}_1(\omega)$ (■) and $\alpha_1\tilde{g}_1(\omega)$ (●) in Example 3.2.1

subresultant matrices $\{S_k(\tilde{f}_1(\omega), \alpha_1\tilde{g}_1(\omega))\}$ are plotted in Figure 3.2i. Diagonal entries of the matrices $\{R_{1,k}\}$ are plotted in Figure 3.2ii, where $R_{1,k}$ is obtained by the QR decomposition of the kth preprocessed subresultant matrix. Both methods reveal the degree of the GCD as $t = 24$, but have significantly different sets of values.



(i) The singular values $\{\sigma_{k,i}\}$ of the preprocessed subresultant matrices $\{S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega))\}$



(ii) The diagonal entries of the matrices $\{R_{1,k}\}$ obtained by the QR decomposition of subresultant matrices $\{S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega))\}$

Figure 3.2: The singular values of $\{S_k\}$ and the diagonal entries of $\{R_{1,k}\}$ from the QR decomposition of $\{S_k\}$ in Example 3.2.1

□

This section has considered methods for the computation of the degree of the GCD (or AGCD) of two univariate polynomials in Bernstein form. Several methods based on either the QR decomposition or the SVD were considered.

In Example 3.2.1 there is a clear separation between the numerically zero and non-zero singular values in the decomposition of each subresultant matrix S_k for $k = 1, \ldots, \min(m, n)$ in the subresultant matrix sequence. However, there is not a distinct separation between the numerically zero and non-zero diagonals of the matrices $\{R_{1,k}\}$ for $k = 1, \ldots, 12$. The remainder of this thesis makes frequent use of the singular value decomposition for the computation of the degree of the AGCD of two inexact polynomials.

3.3 The Optimal Variant of the Subresultant Matrices for the Computation of the Degree of the GCD

Five subresultant matrix variants were considered in Section 3.1.4, and methods for the computation of the degree of the GCD were considered in Section 3.2. Now, by a set of examples, it will be shown that $D_{m+n-k}^{-1}T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$ is typically the optimal variant of the subresultant matrix for use in the computation of the degree of the GCD and the approximation of its coefficients. The chosen examples are non trivial and include polynomials whose coefficients span many orders of magnitude, and are of signifcant degree. However, the examples included are not exhaustive, and the stated optimality is in relation to the examples shown, and should no rigorous proof of optimality is given.

Theoretically, the rank of each variant is equal, that is,

$$\begin{aligned}\operatorname{rank}\left(D_{m+n-k}^{-1}T_k\left(\hat{f}(x),\hat{g}(x)\right)\hat{Q}_k\right) &= \operatorname{rank}\left(D_{m+n-k}^{-1}T_k\left(\hat{f}(x),\hat{g}(x)\right)\right)\\ &= \operatorname{rank}\left(T_k\left(\hat{f}(x),\hat{g}(x)\right)\hat{Q}_k\right)\\ &= \operatorname{rank}\left(T_k\left(\hat{f}(x),\hat{g}(x)\right)\right)\\ &= \operatorname{rank}\left(\breve{S}_k\left(\hat{f}(x),\hat{g}(x)\right)\right).\end{aligned}$$

However, in practical problems, the numerical rank of these matrices can differ, and in this set of examples some variants are better suited to the computation of the degree of the GCD. This does not discount the other variants which under preprocessing and with infinite precision, are equally valid choices.

Example 3.3.1 considers scaling the coefficients of the polynomial $\hat{f}(x)$ in the first partition of each subresultant matrix variant due to their coefficient multipliers. The scaling effect of the coefficient multipliers across both partitions of the subresultant matrices are then considered in Example 3.3.2, where heat maps of the entries of $\hat{f}(x)$ and $\hat{g}(x)$ for each of the subresultant matrix variants are plotted. Finally, Example 3.3.3 considers the singular values of the set of subresultant matrices for each subresultant matrix variant as a method for the determination of the degree of the GCD of two polynomials.

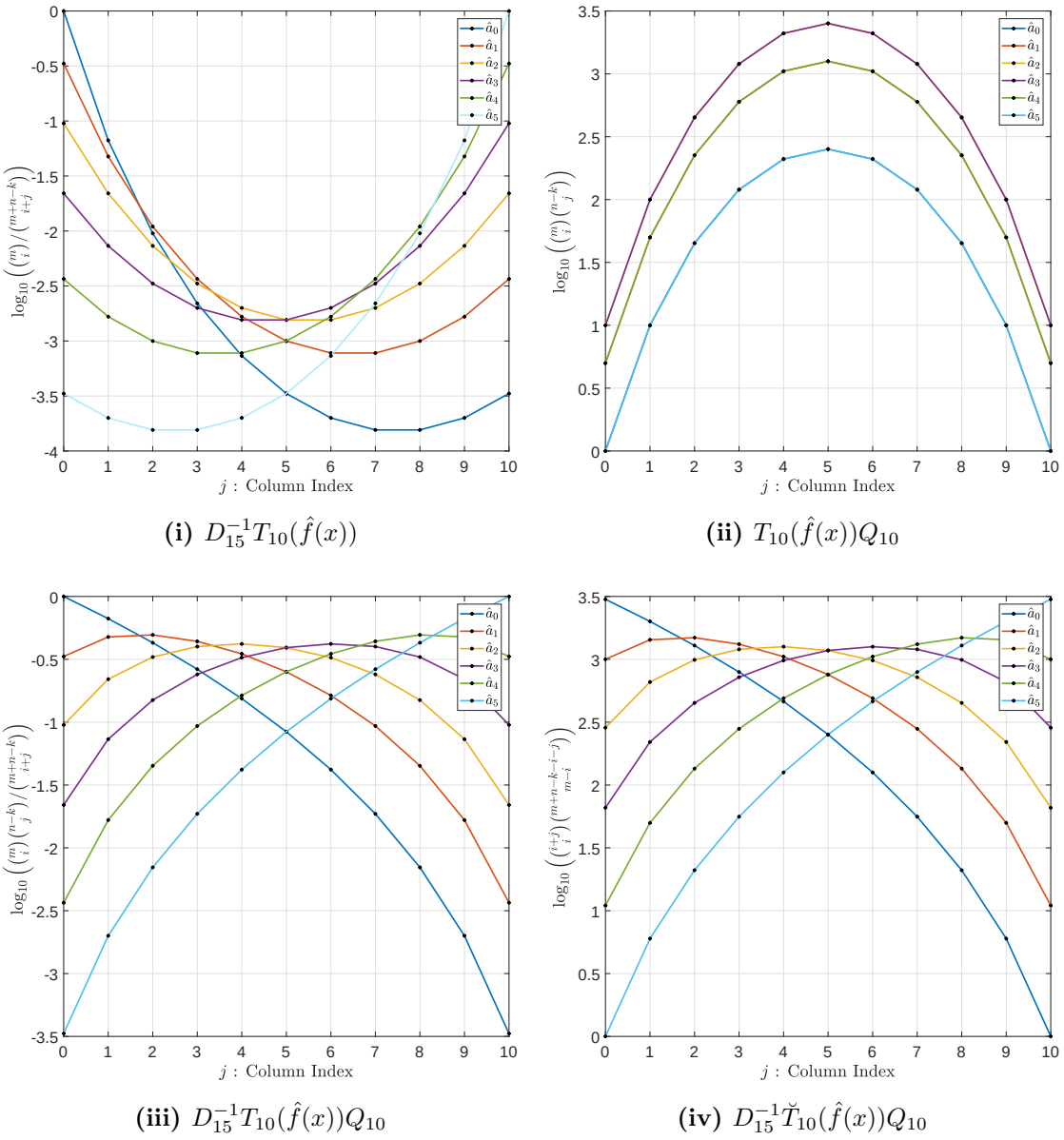
Example 3.3.1. Let $\hat{f}(x)$ and $\hat{g}(x)$ be polynomials of degree $m = 5$ and $n = 11$, and consider the scaling effect on the polynomial coefficients $\hat{a}_i$ due to the coefficient multipliers found in the entries in the first subresultant matrix for all variants. This example considers the scaling of six coefficients $\hat{a}_i$ for $i = 0, \ldots, 5$, in each of the eleven columns $\{\, c_j \mid j = 0, \ldots, 10 \,\}$ of the four subresultant matrix variants. The first variant given by $T_{n-k}(\hat{f}(x))$ is omitted in this example since the coefficient multiplier of entries containing $\hat{a}_i$ is equal for all columns. The four variants considered are (i) $D_{m+n-k}^{-1}T_{n-k}(\hat{f}(x))$, (ii) $T_{n-k}(\hat{f}(x))Q_{n-k}$, (iii) $D_{m+n-k}^{-1}T_{n-k}(\hat{f}(x))Q_{n-k}$ and (iv) $D_{m+n-k}^{-1}\breve{T}_{n-k}(\hat{f}(x))Q_{n-k}$.

In Figures 3.3i to 3.3iv the ith plotted line represents the scaling of coefficient $\hat{a}_i$ due to coefficient multipliers across each column of the four subresultant matrix variants. For instance, Figure 3.3i plots the coefficient multipliers which multiply any one coefficient $\hat{a}_i$ in $D_{m+n-k}^{-1}T_{n-k}(\hat{f}(x))$, that is, the coefficient multipliers $\left\{ \frac{\binom{m}{i}}{\binom{m+n-k}{i+j}} \mid j = 0, \ldots, n-k \right\}$ are plotted for each $\hat{a}_i$. From Figure 3.3i it can be seen that the entries in the middle columns are of significantly smaller magnitude than those in the surrounding columns, while in Figure 3.3ii the opposite is true and scaling due to the coefficient multipliers in the middle columns is significantly larger than in the remaining columns.

The optimal form of scaling is given by the coefficient multipliers in the variant $D_{m+n-k}^{-1}T_{n-k}(\hat{f}(x))Q_{n-k}$ as seen in Figure 3.3iii, where coefficients are scaled over the unit interval.

Example 3.3.2. This example considers the scaling (due to the coefficient multipliers) of the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$ in the entries of the fifth subresultant matrix for each of the four variants. In this example the arbitrary polynomials $\hat{f}(x)$ and $\hat{g}(x)$ of degrees $m = 5$ and $n = 15$ are considered.

Heat maps of the coefficient multipliers in the entries of the fifth subresultant matrix of the sets (i) $\{T_k(\hat{f}(x),\hat{g}(x))\}$, (ii) $\{D_{m+n-k}^{-1}T_k(\hat{f}(x),\hat{g}(x))\}$, (iii) $\{T_k(\hat{f}(x),\hat{g}(x))\hat{Q}_k\}$



(i) $D_{15}^{-1}T_{10}(\hat{f}(x))$ **(ii)** $T_{10}(\hat{f}(x))Q_{10}$

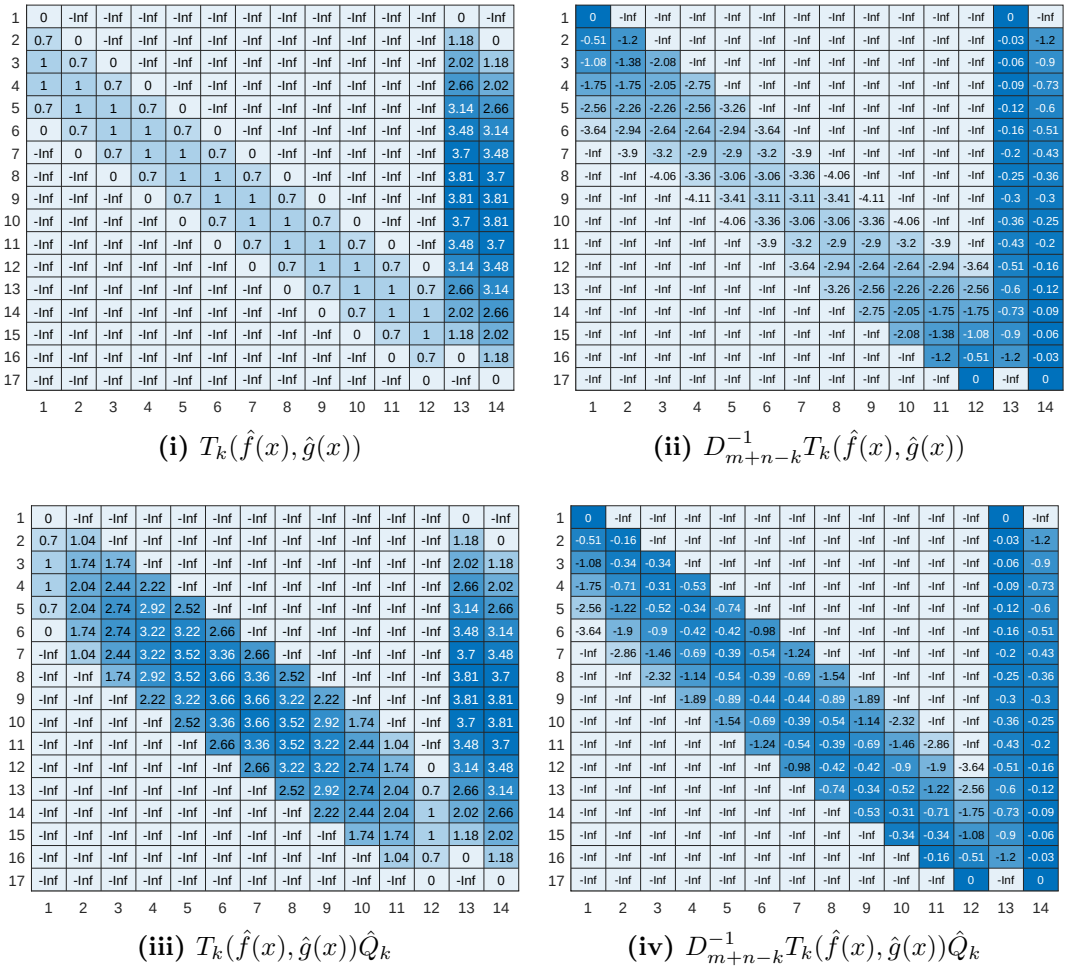
(iii) $D_{15}^{-1}T_{10}(\hat{f}(x))Q_{10}$ **(iv)** $D_{15}^{-1}\breve{T}_{10}(\hat{f}(x))Q_{10}$

Figure 3.3: Scaling of the coefficients $\{\,\hat{a}_i \mid i = 0, \ldots, 5\,\}$ in the first partition of four subresultant matrix variants where $k = 5$ in Example 3.3.1

and (iv) $\{D_{m+n-k}^{-1}T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k\}$. are plotted in Figure 3.4 In Figure 3.4i the coefficient multipliers in the second partition are significantly larger than the coefficient multipliers in the first partition for the variant $T_k(f(x), g(x))$, since $n >> m$. The coefficient multipliers in both partitions are of similar magnitude for $D_{m+n-k}^{-1}T_k(\hat{f}(x), \hat{g}(x))$ and $T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$, but Figure 3.4ii and Figure 3.4iii show that some entries in $D_{m+n-k}^{-1}T_k(\hat{f}(x), \hat{g}(x))$ and $T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$ are subject to more scaling than others because their coefficient multipliers are larger.

From Figure 3.4iv it can be seen that the optimal form of scaling is achieved by the subresultant matrix variant given by $D_{m+n-k}^{-1}T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$, where non-zero entries in both partitions are equivalently scaled.

Example 3.3.3. Consider the Bernstein forms of the exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ of



(i) $T_k(\hat{f}(x), \hat{g}(x))$ **(ii)** $D^{-1}_{m+n-k} T_k(\hat{f}(x), \hat{g}(x))$

(iii) $T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$ **(iv)** $D^{-1}_{m+n-k} T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$

Figure 3.4: The magnitude of the coefficient multipliers in the fifth subresultant matrix of the four subresultant matrix variants in Example 3.3.2

degrees $m = 25$ and $n = 17$, whose factorised forms are given by

$$\hat{f}(x) = (x - 1.46)^2(x - 1.37)^3(x - 0.82)^3(x - 0.75)^3(x - 0.56)^8(x - 0.1)^3(x + 0.27)^3$$
$$\hat{g}(x) = (x - 2.12)(x - 1.37)^3(x - 1.2)^3(x - 0.99)^4(x - 0.75)^3(x - 0.56)^8(x - 0.1)^2$$

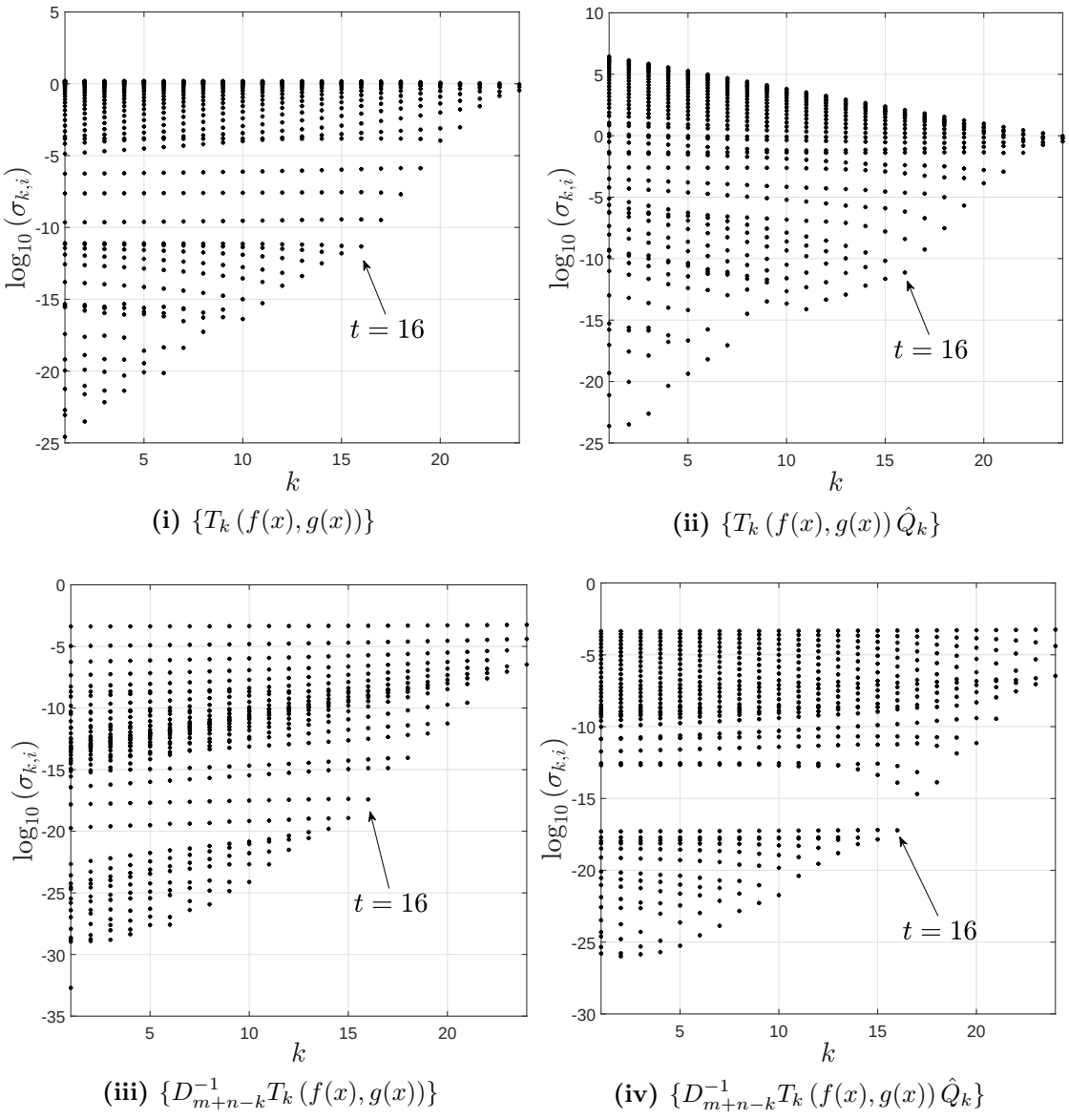
and whose GCD $\hat{d}(x, y)$ of degree $t = 16$ is given by

$$\hat{d}(x) = (x - 1.37)^3(x - 0.75)^3(x - 0.56)^8(x - 0.1)^2.$$

Random noise is added to the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$ such that the coefficients of the inexact polynomials $f(x)$ and $g(x)$ are given by (3.24), where $\{ r_{f,i} \mid i = 0, \ldots, m \}$ and $\{ r_{g,j} \mid j = 0, \ldots, n \}$ are uniformly distributed random variables in the interval $[-1, 1]$, and $\{ \epsilon_{f,i} \mid i = 0, \ldots, m \}$ and $\{ \epsilon_{g,j} \mid j = 0, \ldots, n \}$ are uniformly distributed random variables in the interval $\left[10^{-12}, 10^{-10}\right]$.

The singular values $\{ \sigma_{k,i} \mid k = 1, \ldots, \min(m, n); i = 1, \ldots, m + n - 2k + 2 \}$ of the sets of subresultant matrices (i) $\{T_k(f(x), g(x))\}$, (ii) $\{D^{-1}_{m+n-k} T_k(f(x), g(x))\}$, (iii) $\{T_k(f(x), g(x))\hat{Q}_k\}$ and (iv) $\{D^{-1}_{m+n-k} T_k(f(x), g(x))\hat{Q}_k\}$ are plotted in Figures 3.5i to 3.5iv.

Two methods, DC1 and DC2, can be used to determine the degree of the AGCD



(i) $\{T_k(f(x), g(x))\}$ **(ii)** $\{T_k(f(x), g(x))\,\hat{Q}_k\}$

(iii) $\{D_{m+n-k}^{-1} T_k(f(x), g(x))\}$ **(iv)** $\{D_{m+n-k}^{-1} T_k(f(x), g(x))\,\hat{Q}_k\}$

Figure 3.5: The singular values $\{\sigma_{k,i}\}$ of each subresultant matrix for each of the four subresultant matrix variants in Example 3.3.3

from the SVD of the set of subresultant matrices in this example. These methods were described in Section 3.2.1. From Figures 3.5i to 3.5iii, the degree of the AGCD cannot be determined by either of the DC1 or DC2 methods and there is no clear, discernible separation between the non-zero and numerically zero singular values of theses subresultant matrices. However, in Figure 3.5iv the non-zero and numerically zero singular values of the subresultant matrices $\{S_k(f(x), g(x))\}$ are shown to be well separated and the degree of the AGCD is correctly determined as $t = 16$ by DC1 due to the separation of the singular values of the first subresultant matrix.

In this example, the degree of the AGCD is not correctly determined by the DC2 method when only considering the set of minimum singular values $\{\dot{\rho}_k = \log_{10}(\dot{\sigma}_k)\}$ of any of the four variants, since $\max\{\delta\dot{\rho}_k\}$ (defined in (3.16)) is small for all $k = 1, \ldots, \min(m, n) - 1$.

□

Example 3.3.4. This example is similar to the previous example, but here the degree m of $\hat{f}(x)$ is significantly larger than the degree n of $\hat{g}(x)$, where $\hat{f}(x)$ and $\hat{g}(x)$ are given by

$$\begin{aligned}\hat{f}(x) &= (x-1.46)(x-0.82)^{30}(x-0.7515678)^2(x-0.37)^5(x-0.10122344)\times \\ &\quad (x+2.27564657)^{20} \\ \hat{g}(x) &= (x-1.2222222)(x-0.99102445)^5(x-0.7515678)^2(x-0.37)^5(x-0.12)^5,\end{aligned}$$

which are of degrees $m = 59$ and $n = 18$ respectively. The GCD $\hat{d}_t(x)$ of degree $t = 7$ is given by

$$\hat{d}_t(x) = (x-0.7515678)^2(x-0.37)^5.$$

Noise is added to the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$ as in (3.24), where $\{\, \epsilon_{f,i} \mid i = 0, \ldots, m \,\}$ and $\{\, \epsilon_{g,j} \mid j = 0, \ldots, n \,\}$ are uniformly distributed random variables in the interval $[10^{-10}, 10^{-9}]$.

The singular values of each of the sets of subresultant matrices (i) $\{T_k(f(x), g(x))\}$, (ii) $\{D^{-1}_{m+n-k} T_k(f(x), g(x))\}$, (iii) $\{T_k(f(x), g(x))\hat{Q}_k\}$ and (iv) $\{D^{-1}_{m+n-k} T_k(f(x), g(x))\hat{Q}_k\}$ are plotted in Figures 3.6i to 3.6iv.

The singular values of the set of subresultant matrices $\{T_k(f(x), g(x))\}$ are plotted in Figure 3.6i. From the set $\{\sigma_{k,i} \mid k = 1, \ldots, \min(m, n);\ i = 1, \ldots m + n - 2k + 2\}$, this variant is ill-suited to the purpose of computing the degree of the AGCD by analysis of the set of singular values. There is a clear separation between a subset of large singular values of magnitude in the interval $[10^5, 10^{10}]$ and the remaining singular values of magnitude less than one. However, this separation is due to poor scaling of the two partitions $C_{n-k}(f(x))$ and $C_{m-k}(g(x))$, and is not indicative of numerical rank deficiency in all of the subresultant matrices.

There is however some suggestion of separation between the numerically zero and non-zero singular values of the set of subresultant matrices $\{D^{-1}_{m+n-k} T_k(f(x), g(x))\}$ seen in Figure 3.6ii. Recall from the DC2 method that the degree of the AGCD is given by the index of the maximum change in magnitude of the minimum singular values, that is, the index of the maximum entry of the set $\{\delta\dot{\rho}_i \ i = 1, \ldots, \min(m, n) - 1\}$. Where each $\delta\dot{\rho}_i$, defined in (3.16), is given by $\delta\dot{\rho}_i = \dot{\rho}_{i+1} - \dot{\rho}_i$ and $\dot{\rho}_i = \log_{10}(\dot{\sigma}_i)$. The maximum entry of the set $\{\delta\dot{\rho}_i\}$ is given by $\delta\dot{\rho}_7$. Therefore, the degree of the AGCD is correctly determined to be $t = 7$. However, $\delta\dot{\rho}_7$ is only marginally larger than the other entries in the set $\{\delta\dot{\rho}_i\}$.

The subresultant matrices in the set $\{D^{-1}_{m+n-k} T_k(f(x), g(x))\hat{Q}_k\}$ have singular values with a more significant separation found between the set of non-zero and numerically zero values $\{\sigma_{k,i} \mid k = 1, \ldots, \min(m, n);\ i = 1, \ldots, m + n - 2k + 2\}$. These singular values are plotted in Figure 3.6iv.

The set of minimum singular values can be used for the computation of the degree of the GCD as described in the DC2 method. The minimum singular values are denoted $\{\dot{\sigma}_k\}$ The maximum entry in the set $\{\delta\dot{\rho}_i\}$ is given by $\delta\dot{\rho}_7$ and the degree of the AGCD is given by $t = 7$. There is also a separation between the numerically zero and non-zero singular values of the first subresultant matrix $S_1(f(x), g(x))$. By the DC1 method, the degree of the AGCD is given by the numerical rank loss of $S_1(f(x), g(x))$, which also gives the degree of the AGCD as $t = 7$. The degree of the AGCD can therefore be determined

by several methods which use the singular values of this set of subresultant matrices.

Finally, approximations of the coefficients of the cofactor polynomials $\hat{u}_t(x)$ and $\hat{v}_t(x)$ and the AGCD $\hat{d}_t(x)$ are computed from the tth subresultant matrix for each subresultant matrix variant. The errors between the approximations $u_t(x)$, $v_t(x)$ and $d_t(x)$ and the exact polynomials $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{d}_t(x)$ are given in Table 3.1. These approximations are obtained by the least squares solution to a system of equations derived from the tth subresultant matrix, a method which is described in Section 3.5. Given that the degree of the AGCD is correctly computed, both the variants of subresultant matrix given by $D_{m+n-t}^{-1}T_t(f(x),g(x))$ and $D_{m+n-t}^{-1}T_t(f(x),g(x))\hat{Q}_t$ return approximations which are of similar accuracy.

Method	$T_t(f,g)$	$D_{m+n-t}^{-1}T_t(f,g)$	$T_t(f,g)\hat{Q}_t$	$D_{m+n-t}^{-1}T_t(f,g)\hat{Q}_t$
Error $\hat{u}_t(x)$	-	$4.616148e-08$	-	$4.615582e-08$
Error $\hat{v}_t(x)$	-	$1.749660e-07$	-	$1.749654e-07$
Error $\hat{d}_t(x)$	-	$1.988072e-07$	-	$1.988315e-07$
Average Error	-	$1.399782e-07$	-	$1.399843e-07$

Table 3.1: Error in the approximations of $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{d}_t(x)$ in Example 3.3.4

Some precision has been lost when compared to the noise of the input polynomials, but methods such as preprocessing the polynomials and computing the low rank approximation of the subresultant matrix will yield improved results in later sections.

□

The five variants of the set of subresultant matrices are given by:

(i) $\{T_k(\hat{f}(x),\hat{g}(x))\}$,

(ii) $\{T_k(\hat{f}(x),\hat{g}(x))\hat{Q}_k\}$

(iii) $\{D_{m+n-k}^{-1}T_k(\hat{f}(x),\hat{g}(x))\}$

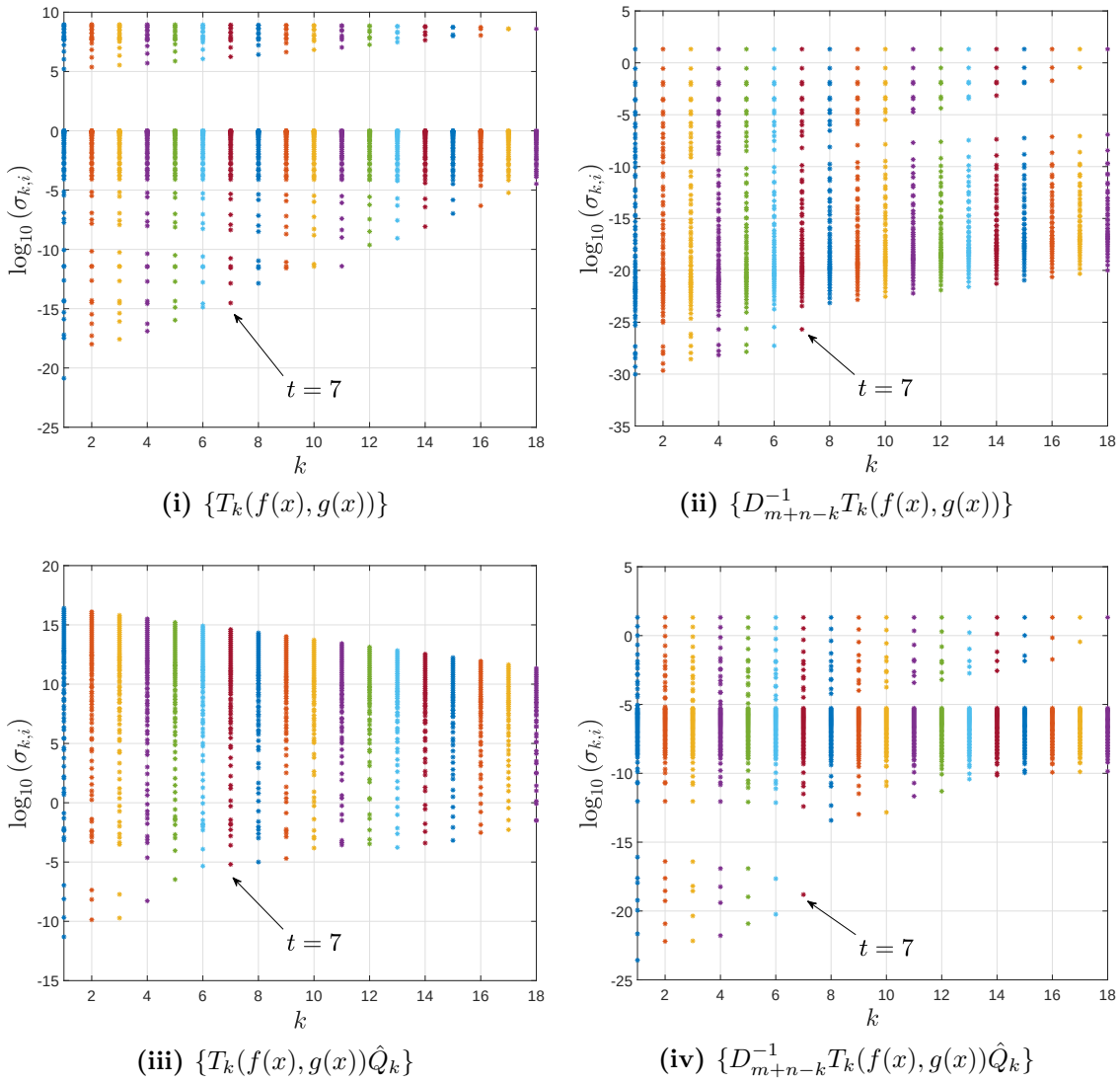
(iv) $\{D_{m+n-k}^{-1}T_k(\hat{f}(x),\hat{g}(x))\hat{Q}_k\}$

(v) $\{D_{m+n-k}^{-1}\tilde{T}_k(\hat{f}(x),\hat{g}(x))\hat{Q}_k\}$

for $k=1,\ldots,\min(m,n)$.

The SVD of the subresultant matrices of any of these variants can, in theory, be used to determine the degree of the GCD or AGCD. In practice, however, there are three situations in which some of the variants fail. These are (i) when noise is added to the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$, (ii) when the degrees of one or both of the polynomials are large and (iii) when the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$ span different ranges of magnitude.

It was shown in Example 3.3.3 that the optimal variant of the subresultant matrix sequence is given by $\{D_{m+n-k}^{-1}T_k(\hat{f}(x),\hat{g}(x))\hat{Q}_k\}$ due to the coefficient multipliers being of similar magnitude across all entries in both partitions of each of the subresultant matrices. This variant also exhibited the best separation between the numerically zero and non-zero singular values of its SVD.



(i) $\{T_k(f(x), g(x))\}$ **(ii)** $\{D^{-1}_{m+n-k} T_k(f(x), g(x))\}$

(iii) $\{T_k(f(x), g(x))\hat{Q}_k\}$ **(iv)** $\{D^{-1}_{m+n-k} T_k(f(x), g(x))\hat{Q}_k\}$

Figure 3.6: The singular values $\{\sigma_{k,i}\}$ of each subresultant matrix for each of the four subresultant matrix variants in Example 3.3.4

The ratio of the entry of maximum magnitude to the entry of minimum magnitude in the matrices of the subresultant matrix sequence $\{D^{-1}_{m+n-k} T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k\}$ may still be large. This could be caused by the coefficients of one polynomial being much larger than the coefficients of the other. Alternatively, the degree of one polynomial may be significantly larger than the other. It is therefore advantageous to preprocess the subresultant matrices and methods of preprocessing are considered in the next section.

3.4 Preprocessing the Subresultant Matrices

The previous section described how it is advantageous to compute the degree of the GCD of two polynomials $\hat{f}(x)$ and $\hat{g}(x)$ from the sequence of subresultant matrices of the form $\{S_k(\hat{f}(x), \hat{g}(x)) = D^{-1}_{m+n-k} T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k\}$. The polynomials $\hat{f}(x)$ and $\hat{g}(x)$ are preprocessed for use in each subresultant matrix, which typically results in a more reliable computation of the degree of the GCD and its coefficients with minimal additional computational complexity.

In this section three preprocessing stages are developed and applied to the polynomials

used in the subresultant matrices. These preprocessing operations have previously been developed for the subresultant matrices of two polynomials in the power basis in [69,70,72, 73], where preprocessing is only required once since the entries of each subresultant matrix are equal. However, the subresultant matrices $S_k(\hat{f}(x), \hat{g}(x))$ for $k = 1, \ldots, \min(m, n)$ of two polynomials in Bernstein form have entries which depend on k, and the preprocessing of each subresultant matrix is considered independently [9, 74]. This section summarises the cited work, as well as taking further steps to reduce computational complexity.

The method of preprocessing in this section is defined for exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$, but these same preprocessing operations can also be applied to inexact polynomials $f(x)$ and $g(x)$, as in the examples at the end of the chapter.

The three preprocessing operations are now briefly described before being considered in more detail:

1. **Normalisation by Mean of Non-Zero Entries of the Subresultant Matrix**

   The coefficients of $\hat{f}(x)$ and $\hat{g}(x)$ may span many orders of magnitude, and the entries of the partition $C_{n-k}(\hat{f}(x))$ may be much smaller or larger than the entries of $C_{m-k}(\hat{g}(x))$. It is therefore necessary to scale the polynomials by the arithmetic or geometric mean of their entries in their respective partitions of $S_k(\hat{f}(x), \hat{g}(x))$. The choice of normalisation method is discussed in [71], where the arithmetic mean, geometric mean and 2-norm are considered. Division of non-zero entries by their geometric mean is preferred, particularly in cases where the coefficients of the input polynomials span many orders of magnitude. On first inspection, the geometric mean and arithmetic mean of the non-zero entries in the partitions of the subresultant matrices have a significantly more complicated expression than an equivalent Sylvester matrix of two polynomials in the power basis. However, simplified expressions can be derived. The polynomials normalised by the mean of the non-zero entries in the kth subresultant matrix are denoted $\bar{f}_k(x)$ and $\bar{g}_k(x)$.

   Since the polynomial GCD is defined within scalar multipliers, the GCD of the normalised polynomials is equal to the GCD of the original forms:

   $$\mathrm{GCD}\left(\hat{f}(x), \hat{g}(x)\right) \sim \mathrm{GCD}\Big(\bar{f}_k(x), \bar{g}_k(x)\Big),$$

   where $\sim$ is an equivalence to within a non-zero scalar multiplier.

2. **Scaling the Partitions of the kth Subresultant Matrix**

   The polynomial GCD is defined to within a scalar constant, so $\mathrm{GCD}(\hat{f}(x), \hat{g}(x)) \sim \mathrm{GCD}(\hat{f}(x), \alpha\hat{g}(x))$. Similarly,

   $$\mathrm{rank}\left(S_k\left(\bar{f}_k(x), \bar{g}_k(x)\right)\right) = \mathrm{rank}\left(S_k\left(\bar{f}_k(x), \alpha\bar{g}_k(x)\right)\right),$$

   so both forms can be used in the computation of the degree t of the GCD. However, typically better results are obtained in the version where the two partitions are balanced due to the inclusion of an optimised α for $k = 1, \ldots, \min(m, n)$. The optimal value α_k is chosen such that the non-zero entries in $C_{n-k}(\bar{f}_k(x))$ and $C_{m-k}(\alpha_k\bar{g}_k(x))$ are of a similar order of magnitude. The computation α_k requires the solution of a linear programming problem for each subresultant matrix.

3. **Change of Independent Variable**

   The independent variable x is replaced by the independent variable ω using the substitution $x = \theta\omega$. The non-zero parameter $\theta \in \mathbb{R}$ is optimal when the coefficients of the polynomials $\tilde{f}_k(\omega)$ and $\tilde{g}_k(\omega)$ are of similar orders of magnitude, and the difference between the entry of maximum magnitude and entry of minimum magnitude in the *kth* subresultant matrix is minimised. As with α_k, the optimal value of θ_k is given by the solution of a linear programming problem, and both α_k and θ_k are therefore computed simultaneously using a combined linear programming problem.

3.4.1 Normalisation

The Arithmetic Mean of the Non-Zero Entries of the Subresultant Matrices

Preprocessing the polynomials $\hat{f}(x)$ and $\hat{g}(x)$ in each subresultant matrix in the sequence $S_k(\hat{f}(x), \hat{g}(x))$ is computationally expensive, but it is necessary since the entries of each subresultant matrix $S_k(\hat{f}(x), \hat{g}(x))$ are dependent on k. This section therefore focuses on efficient methods for of computing the arithmetic mean of non-zero entries contained in the two partitions of the subresultant matrices.

Let $\mathcal{A}_k(\hat{f}(x))$ denote the arithmetic mean of the non-zero entries in the first partition of the kth subresultant matrix, that is, the $(n-k)$th convolution matrix of the polynomial $\hat{f}(x)$.

Proposition 1. *The arithmetic mean of the non-zero entries in the first partition, $C_{n-k}(\hat{f}(x))$, of the kth subresultant matrix is given by*

$$\mathcal{A}_{n-k}\left(\hat{f}(x)\right) = \frac{1}{(m+1)(n-k+1)} \sum_{j=0}^{n-k} \sum_{i=0}^{m} \frac{\hat{a}_i \binom{m}{i} \binom{n-k}{j}}{\binom{m+n-k}{i-j}} \tag{3.25}$$

and this can be simplified to the form

$$\mathcal{A}_{n-k}\left(\hat{f}(x)\right) = \frac{m+n-k+1}{(m+1)^2(n-k+1)} \sum_{i=0}^{m} \hat{a}_i. \tag{3.26}$$

Proof. The $(n-k+1)$ entries of the $(i+1)$th diagonal of the matrix are of the form

$$\hat{a}_i \frac{\binom{i+j}{i}\binom{m+n-k-i-j}{m-i}}{\binom{m+n-k}{m}} \qquad \text{for} \qquad j = 0, \ldots, n-k.$$

The sum of these entries along each diagonal is given by

$$\begin{aligned} \sum_{j=0}^{n-k} \hat{a}_i \frac{\binom{i+j}{i}\binom{m+n-k-i-j}{m-i}}{\binom{m+n-k}{m}} &= \frac{(n-k+1)}{\binom{m+n-k}{m}} \sum_{j=0}^{n-k} \hat{a}_i \binom{i+j}{i}\binom{m+n-k-i-j}{m-i} \\ &= \hat{a}_i \frac{m+n-k+1}{m+1}, \end{aligned}$$

and therefore the computation of the arithmetic mean in (3.25) is simplified to

$$\begin{aligned}\mathcal{A}_k\left(\hat{f}(x)\right) &= \frac{1}{(m+1)(n-k+1)}\sum_{i=0}^{m}\hat{a}_i\frac{m+n-k+1}{m+1}\\ &= \frac{m+n-k+1}{(m+1)^2(n-k+1)}\sum_{i=0}^{m}\hat{a}_i.\end{aligned}$$

□

The expression (3.25) requires the evaluation of three binomial coefficients for each of the $(n-k+1)\times(m+1)$ non-zero entries, and this can have significant computational cost. This is reduced significantly by utilising the new expression for $\mathcal{A}_{n-k}(\hat{f}(x))$ in (3.26).

Corollary 1. *The arithmetic mean of the non-zero entries in the first partition, $C_{n-k-1}(\hat{f}(x))$, of the $(k+1)$th subresultant matrix $S_{k+1}(\hat{f}(x),\hat{g}(x))$ is given by*

$$\mathcal{A}_{n-k-1}\left(\hat{f}(x)\right) = \frac{(m+n-k)}{(m+1)^2(n-k+1)}\sum_{i=0}^{m}\hat{a}_i.$$

Alternatively, given $\mathcal{A}_{n-k}(\hat{f}(x))$ the arithmetic mean $\mathcal{A}_{n-k-1}(\hat{f}(x))$ is given by

$$\mathcal{A}_{n-k-1}\left(\hat{f}(x)\right) = \frac{(m+n-k)(n-k+1)}{(m+n-k+1)(n-k)}\times\mathcal{A}_{n-k}\left(\hat{f}(x)\right).$$

Corollary 2. *Given $\mathcal{A}_k(\hat{f}(x))$, the arithmetic mean of the non-zero entries of the first partition of the $(k+p)$th subresultant matrix is given by*

$$\mathcal{A}_{n-k-p}\left(\hat{f}(x)\right) = \frac{(m+n-k-p+1)(n-k+1)}{(n-k-p+1)(m+n-k+1)}\times\mathcal{A}_{n-k}\left(\hat{f}(x)\right),$$

which significantly reduces the complexity of computing the set of arithmetic means $\{\,\mathcal{A}_{n-k}(\hat{f}(x)) \mid k=1,\ldots,\min(m,n)\,\}$ and $\{\,\mathcal{A}_{m-k}(\hat{g}(x)) \mid k=1,\ldots,\min(m,n)\,\}$.

The Geometric Mean of Non-Zero Entries of the Subresultant Matrices

The first preprocessing operation normalises the polynomials $\hat{f}(x)$ and $\hat{g}(x)$ by the geometric means $\mathcal{G}_{n-k}(\hat{f}(x))$ and $\mathcal{G}_{m-k}(\hat{g}(x))$ of their entries in the kth subresultant matrix. This can be computationally expensive, since the geometric mean must be computed for each subresultant matrix, the entries of which are unique compared with any of the other subresultant matrices. For completeness, and in the pursuit of efficiency, a simplified expression for the computation of these geometric means is derived. An expression of the geometric mean of the non-zero entries in a partition of a subresultant matrix is given in [74], and this is now developed further.

The non-zero entries in the first partition, $C_{n-k}(\hat{f}(x))$, of the kth subresultant matrix $S_k(\hat{f}(x),\hat{g}(x))$ are given by

$$\frac{\left|\hat{a}_i\binom{m}{i}\binom{n-k}{0}\right|}{\binom{m+n-k}{i}}, \frac{\left|\hat{a}_i\binom{m}{i}\binom{n-k}{1}\right|}{\binom{m+n-k}{i+1}}, \ldots, \frac{\left|\hat{a}_i\binom{m}{i}\binom{n-k}{n-k}\right|}{\binom{m+n-k}{i+n-k}}$$

and their geometric mean as given by [74] is computed using the expression

$$\mathcal{G}_{n-k}\left(\hat{f}(x)\right) = \left(\prod_{i=0}^{m}\prod_{j=0}^{n-k}\frac{\left|\hat{a}_i\binom{m}{i}\binom{n-k}{j}\right|}{\binom{m+n-k}{i+j}}\right)^{\frac{1}{(n-k+1)(m+1)}}, \tag{3.27}$$

which is simplified to

$$\mathcal{G}_{n-k}\left(\hat{f}(x)\right) = \prod_{i=0}^{m}\left(\frac{\left(|\hat{a}_i|\binom{m}{i}\right)^{n-k+1}\prod_{j=0}^{n-k}\binom{n-k}{j}}{\prod_{j=0}^{n-k}\binom{m+n-k}{i+j}}\right)^{\frac{1}{(m+1)(n-k+1)}}.$$

The two partitions of the rearranged form of the kth subresultant matrix, defined in (3.12) and (3.13), give rise to a new, more efficient method of computing the geometric means of the non-zero entries of $C_{n-k}(\hat{f}(x))$ and $C_{m-k}(\hat{g}(x))$.
The absolute values of the non-zero entries of $C_{n-k}(\hat{f}(x))$ in the rearranged form are given by

$$\frac{\left|\hat{a}_i\binom{i}{i}\binom{m+n-k-i}{m-i}\right|}{\binom{m+n-k}{n-k}}, \frac{\left|\hat{a}_i\binom{i+1}{i}\binom{m+n-k-i-1}{m-i}\right|}{\binom{m+n-k}{m}}, \ldots, \frac{\left|\hat{a}_i\binom{i+n-k}{i}\binom{m-i}{m-i}\right|}{\binom{m+n-k}{m}}.$$

Proposition 2. *The geometric mean $\mathcal{G}_{n-k}(\hat{f}(x))$ of the non-zero entries of $C_{n-k}(\hat{f}(x))$ is given by*

$$\mathcal{G}_{n-k}\left(\hat{f}(x)\right) = \left(\prod_{i=0}^{m}\prod_{j=0}^{n-k}\frac{\left|\hat{a}_i\binom{i+j}{i}\binom{m+n-k-i-j}{m-i}\right|}{\binom{m+n-k}{m}}\right)^{\frac{1}{(n-k+1)(m+1)}} \tag{3.28}$$

and is reduced to the more computationally efficient expression

$$\mathcal{G}_{n-k}\left(\hat{f}(x)\right) = \frac{\left(\prod_{i=0}^{m}|\hat{a}_i|\right)^{\frac{1}{m+1}}\left[\prod_{j=0}^{n-k}\prod_{i=0}^{m}\binom{i+j}{j}\right]^{\frac{2}{(n-k+1)(m+1)}}}{\binom{m+n-k}{m}}.$$

Proof. The geometric mean in (3.28) can be considered in three parts. Let $\mathcal{G}_{n-k}(\hat{f}(x))$ be given by

$$\mathcal{G}_{n-k}\left(\hat{f}(x)\right) = \mathcal{A} \times \mathcal{B}_k \times \mathcal{C}_k,$$

where

$$\mathcal{A} = \left(\prod_{i=0}^{m} \prod_{j=0}^{n-k} |\hat{a}_i| \right)^{\frac{1}{(n-k+1)(m+1)}} = \left(\prod_{i=0}^{m} |\hat{a}_i| \right)^{\frac{1}{m+1}}, \tag{3.29}$$

$$\mathcal{B}_k = \left(\prod_{i=0}^{m} \prod_{j=0}^{n-k} \binom{i+j}{i} \binom{m+n-k-i-j}{m-i} \right)^{\frac{1}{(n-k+1)(m+1)}}, \tag{3.30}$$

$$\mathcal{C}_k = \left(\prod_{i=0}^{m} \prod_{j=0}^{n-k} \frac{1}{\binom{m+n-k}{m}} \right)^{\frac{1}{(n-k+1)(m+1)}} = \frac{1}{\binom{m+n-k}{m}}. \tag{3.31}$$

The expressions $\mathcal{A}$ and $\mathcal{C}_k$ are simplified above in (3.29) and (3.31), and the binomial terms in $\mathcal{B}_k$ have the property that

$$\prod_{j=0}^{n-k} \prod_{i=0}^{m} \binom{i+j}{j} \equiv \prod_{j=0}^{n-k} \prod_{i=0}^{m} \binom{m+n-k-i-j}{m-i},$$

such that (3.30) is simplified and given by

$$\mathcal{B}_k = \left(\prod_{j=0}^{n-k} \prod_{i=0}^{m} \binom{i+j}{j} \right)^{\frac{2}{(n-k+1)(m+1)}}. \tag{3.32}$$

Therefore, the geometric mean in (3.28) can be simplified to

$$\mathcal{G}_{n-k}\left(\hat{f}(x)\right) = \frac{\left[\prod_{i=0}^{m} |\hat{a}_i|\right]^{\frac{1}{m+1}} \left[\prod_{j=0}^{n-k} \prod_{i=0}^{m} \binom{i+j}{j}\right]^{\frac{2}{(n-k+1)(m+1)}}}{\binom{m+n-k}{m}}. \tag{3.33}$$

□

Similarly, the geometric mean of the entries of $C_{m-k}(\hat{g}(x))$ is given by

$$\mathcal{G}_{m-k}\Big(\hat{g}(x)\Big) = \frac{\left(\prod_{i=0}^{n} \left|\hat{b}_i\right|\right)^{\frac{1}{n+1}} \left(\prod_{j=0}^{m-k} \prod_{i=0}^{n} \binom{i+j}{j}\right)^{\frac{2}{(m-k+1)(n+1)}}}{\binom{m+n-k}{n}}.$$

The computation of the geometric means $\mathcal{G}_{n-k}(\hat{f}(x))$ and $\mathcal{G}_{m-k}(\hat{g}(x))$ of each of the subresultant matrices is required for the preprocessing of the polynomials $\{\, \hat{f}_k(x) \mid k = 1, \ldots, \min(m,n) \,\}$ and $\{\, \hat{g}_k(x) \mid k = 1, \ldots, \min(m,n) \,\}$. Computing $\mathcal{G}_{n-k}(\hat{f}(x))$ and $\mathcal{G}_{m-k}(\hat{g}(x))$ using the expression in (3.27) is computationally expensive, and the rearrangement developed in this section goes some way to reducing the complexity. An expression is now derived for the computation of $\mathcal{G}_{n-k-1}(\hat{f}(x))$ given the component parts of the geometric mean $\mathcal{G}_{n-k}(\hat{f}(x))$, and this further reduces the computational complexity.

Proposition 3. *The geometric mean $\mathcal{G}_{n-k-1}(\hat{f}(x))$ is given by*

$$\mathcal{G}_{n-k-1}\left(\hat{f}(x)\right) = \frac{\left[\prod_{i=0}^{m} \hat{a}_i\right]^{\frac{1}{m+1}} \left[\prod_{j=0}^{n-k-1} \prod_{i=0}^{m} \binom{i+j}{j}\right]^{\frac{2}{(n-k)(m+1)}}}{\binom{m+n-k-1}{m}}.$$

Given $\mathcal{A}$, $\mathcal{B}_k$ and $\mathcal{C}_k$, defined in (3.29,3.30,3.31), the geometric mean $\mathcal{G}_{n-k-1}(\hat{f}(x))$ can be expressed as

$$\mathcal{G}_{n-k-1}\left(\hat{f}(x)\right) = \frac{(m+n-k)\mathcal{A} \times \mathcal{B}_k^{\frac{n-k+1}{n-k}} \times \mathcal{C}_k}{(n-k)\left(\prod_{i=0}^{m}\binom{i+n-k}{n-k}\right)^{\frac{2}{(n-k)(m+1)}}}.$$

Proof. The geometric mean $\mathcal{G}_{n-k-1}(\hat{f}(x))$ is given by

$$\mathcal{G}_{n-k-1}\left(\hat{f}(x)\right) = \mathcal{A} \times \mathcal{B}_{k+1} \times \mathcal{C}_{k+1}.$$

Firstly, $\mathcal{B}_{k+1}$ is given by

$$\mathcal{B}_{k+1} = \left[\prod_{j=0}^{n-k-1}\prod_{i=0}^{m}\binom{i+j}{j}\right]^{\frac{2}{(n-k)(m+1)}} = \frac{\left[\prod_{j=0}^{n-k}\prod_{i=0}^{m}\binom{i+j}{j}\right]^{\frac{2}{(n-k)(m+1)}}}{\left[\prod_{i=0}^{m}\binom{i+n-k}{n-k}\right]^{\frac{2}{(n-k)(m+1)}}},$$

which in terms of $\mathcal{B}_k$ (defined in (3.32)) is given by

$$\mathcal{B}_{k+1} = \frac{\mathcal{B}_k^{\frac{n-k+1}{n-k}}}{\left[\prod_{i=0}^{m}\binom{i+n-k}{n-k}\right]^{\frac{2}{(n-k)(m+1)}}}. \tag{3.34}$$

Secondly, $\mathcal{C}_{k+1}$ is given by

$$\mathcal{C}_{k+1} = \frac{1}{\binom{m+n-k-1}{m}},$$

which when written in terms of $\mathcal{C}_k$ (defined in (3.31)), is given by

$$\mathcal{C}_{k+1} = \frac{m+n-k}{n-k}\,\mathcal{C}_k. \tag{3.35}$$

Given (3.34) and (3.35), $\mathcal{G}_{n-k-1}(\hat{f}(x))$ is given by

$$\mathcal{G}_{n-k-1}\left(\hat{f}(x)\right) = \frac{(m+n-k)\mathcal{A} \times \mathcal{B}_k^{\frac{n-k+1}{n-k}} \times \mathcal{C}_k}{(n-k)\left(\prod_{i=0}^{m}\binom{i+n-k}{n-k}\right)^{\frac{2}{(n-k)(m+1)}}}.$$

□

This rearranged form is computationally more efficient than the previously described approach. The geometric means of the non-zero entries of the two partitions of the *kth* subresultant matrix are denoted $\mathcal{G}_{n-k}(\hat{f}(x))$ and $\mathcal{G}_{m-k}(\hat{g}(x))$ respectively. The coefficients of polynomials in the kth subresultant matrix are replaced by the coefficients of normalised

polynomials $\bar{f}_k(x)$ and $\bar{g}_k(x)$, where

$$\bar{f}_k(x) = \sum_{i=0}^{m} \bar{a}_i \binom{m}{i} (1-x)^{m-i} x^i, \quad \text{where} \quad \bar{a}_i = \frac{\hat{a}_i}{\mathcal{G}_{n-k}\left(\hat{f}(x)\right)},$$
$$\bar{g}_k(x) = \sum_{i=0}^{n} \bar{b}_i \binom{n}{i} (1-x)^{n-i} x^i, \quad \text{where} \quad \bar{b}_i = \frac{\hat{b}_i}{\mathcal{G}_{m-k}\left(\hat{g}(x)\right)}.$$

3.4.2 Computing the Optimal Values of α and θ

Having reduced the complexity of the first preprocessing operation, the optimal values of α and θ from the second and third preprocessing operations are now computed by solving a linear programming problem.

The second preprocessing operation scales the entries of the second partition of the kth subresultant matrix by α_k and the third preprocessing operation replaces the independent variable x with $\theta\omega$, where ω is a new independent variable. Values of α and θ are optimally chosen such that the ratio of entry of maximum magnitude to entry of minimum magnitude in the matrix $S_k(\tilde{f}_k(\omega), \alpha_k \tilde{g}_k(\omega))$ is minimised.

The three preprocessing operations yield the polynomials

$$\ddot{f}_k(\theta,\omega) = \sum_{i=0}^{m} \bar{a}_i \theta^i \binom{m}{i} (1-\theta\omega)^{m-i} \omega^i,$$
$$\alpha\ddot{g}_k(\theta,\omega) = \alpha \sum_{i=0}^{n} \bar{b}_i \theta^i \binom{n}{i} (1-\theta\omega)^{n-i} \omega^i,$$

where α and θ are to be optimised.

The general expression for a non-zero entry of the first partition, $C_{n-k}(\ddot{f}_k(\theta,\omega))$, of the subresultant matrix $S_k(\hat{f}(x), \hat{g}(x))$ is given by

$$C_{n-k}\left(\ddot{f}_k(\theta,\omega)\right)_{(i+j+1,\ j+1)} = \begin{cases} \dfrac{\bar{a}_i \theta^i \binom{i+j}{i}\binom{m+n-k-i-j}{m-i}}{\binom{m+n-k}{m}} & i = 0,\ldots,m;\ j = 0,\ldots,n-k, \\ 0 & \text{otherwise,} \end{cases}$$

and the general expression of a non-zero entry in the second partition, $C_{m-k}(\alpha\ddot{g}_k(\theta,\omega))$, is given by

$$C_{m-k}\left(\alpha\ddot{g}(\theta,\omega)\right)_{(i+j+1,\ j+1)} = \begin{cases} \dfrac{\alpha\bar{b}_i \theta^i \binom{i+j}{i}\binom{m+n-k-i-j}{n-i}}{\binom{m+n-k}{n}} & i = 0,\ldots,n; \quad j = 0,\ldots,m-k, \\ 0 & \text{otherwise.} \end{cases}$$

It is convenient to define the sets of non-zero entries in the two partitions as $\mathscr{P}_{1,k}(\theta)$ and $\mathscr{P}_{2,k}(\alpha,\theta)$ respectively, where

$$\mathscr{P}_{1,k}(\theta) = \left\{ \frac{\left|\bar{a}_i \theta^i \binom{i+j}{i}\binom{m+n-k-i-j}{m-i}\right|}{\binom{m+n-k}{m}} \mid i = 0,\ldots,m,\ j = 0,\ldots,n-k. \right\}$$
$$\mathscr{P}_{2,k}(\alpha,\theta) = \left\{ \frac{\left|\alpha\bar{b}_i \theta^i \binom{i+j}{i}\binom{m+n-k-i-j}{n-i}\right|}{\binom{m+n-k}{n}} \mid i = 0,\ldots,n,\ j = 0,\ldots,m-k \right\}.$$

The minimisation problem described above can be written as

$$(\alpha_k, \theta_k) = \arg\min_{\alpha,\theta} \left\{ \frac{\max\{\max\{\mathscr{P}_{1,k}(\theta)\}, \max\{\mathscr{P}_{2,k}(\alpha,\theta)\}\}}{\min\{\min\{\mathscr{P}_{1,k}(\theta)\}, \min\{\mathscr{P}_{2,k}(\alpha,\theta)\}\}} \right\} \tag{3.36}$$

for each subresultant matrix S_k for $k = 1, \ldots, \min(m, n)$.
The minimisation (3.36) can be written as

$$\begin{array}{lll}
\text{Minimise} & \dfrac{u}{v} & \\
\text{Subject to} & & \\
u & \geq & \dfrac{\left|\bar{a}_i\theta^i\binom{i+j}{i}\binom{m+n-k-i-j}{m-i}\right|}{\binom{m+n-k}{m}} \quad i=0,\ldots,m; \quad j=0,\ldots,n-k, \\
u & \geq & \dfrac{\left|\alpha\bar{b}_i\theta^i\binom{i+j}{i}\binom{m+n-k-i-j}{n-i}\right|}{\binom{m+n-k}{n}} \quad i=0,\ldots,n; \quad j=0,\ldots,m-k, \\
v & \leq & \dfrac{\left|\bar{a}_i\theta^i\binom{i+j}{i}\binom{m+n-k-i-j}{m-i}\right|}{\binom{m+n-k}{m}} \quad i=0,\ldots,m; \quad j=0,\ldots,n-k, \\
v & \leq & \dfrac{\left|\alpha\bar{b}_i\theta^i\binom{i+j}{i}\binom{m+n-k-i-j}{n-i}\right|}{\binom{m+n-k}{n}} \quad i=0,\ldots,n; \quad j=0,\ldots,m-k, \\
v & > & 0, \\
\alpha_k & > & 0, \\
\theta_k & > & 0,
\end{array} \tag{3.37}$$

and using the set of transformations

$$U = \log_{10}(u), \quad V = \log_{10}(v), \quad \bar{\phi} = \log_{10}(\theta), \quad \bar{\mu} = \log_{10}(\alpha),$$
$$\bar{\alpha}_{i,j} = \log_{10}\left(\frac{\left|\bar{a}_i\binom{i+j}{i}\binom{m+n-k-i-j}{m-i}\right|}{\binom{m+n-k}{m}}\right) \quad \text{and} \quad \bar{\beta}_{i,j} = \log_{10}\left(\frac{\left|\bar{b}_i\binom{i+j}{j}\binom{m+n-k-i-j}{n-i}\right|}{\binom{m+n-k}{n}}\right)$$

can be written as

$$\begin{array}{llllll}
\text{Minimise} & U - V & & & & \\
\text{subject to} & & & & & \\
U & -i\bar{\phi} & & \geq & \bar{\alpha}_{i,j} & i=0,\ldots,m; \quad j=0,\ldots,n-k, \\
U & -i\bar{\phi} & -\bar{\mu} & \geq & \bar{\beta}_{i,j} & i=0,\ldots,n; \quad j=0,\ldots,m-k, \\
-V & +i\bar{\phi} & & \geq & -\bar{\alpha}_{i,j} & i=0,\ldots,m; \quad j=0,\ldots,n-k, \\
-V & +i\bar{\phi} & +\bar{\mu} & \geq & -\bar{\beta}_{i,j} & i=0,\ldots,n; \quad j=0,\ldots,m-k.
\end{array} \tag{3.38}$$

Since j only appears on the right-hand side of these inequalities, the values $\bar{\mathscr{m}}_{1,i}$ and $\bar{m}_{1,i}$ are used to denote the maximum and minimum entries in the set $\{\, \bar{\alpha}_{i,j} \mid j = 0, \ldots, n-k \,\}$. Similarly, $\bar{\mathscr{m}}_{2,j}$ and $\bar{m}_{2,j}$ are used to denote the maximum and minimum entries in the set $\{\, \bar{\beta}_{i,j} \,\}$ for $j = 0, \ldots, m-k$ and are given by

$$\begin{array}{lcll}
\bar{\mathscr{m}}_{1,i} & = & \max_{j=0,\ldots,n-k}\{\bar{\alpha}_{i,j}\} & \text{for} \quad i=0,\ldots,m, \\
\bar{\mathscr{m}}_{2,i} & = & \max_{j=0,\ldots,m-k}\{\bar{\beta}_{i,j}\} & \text{for} \quad i=0,\ldots,n, \\
\bar{m}_{1,i} & = & \min_{i=0,\ldots,n-k}\{\bar{\alpha}_{i,j}\} & \text{for} \quad i=0,\ldots,m, \\
\bar{m}_{2,i} & = & \min_{i=0,\ldots,m-k}\{\beta_{i,j}\} & \text{for} \quad i=0,\ldots,n.
\end{array} \tag{3.39}$$

The location of the maximum entry of each coefficient $\bar{a}_i$ in the kth subresultant matrix

is given by $(i+j+1, j+1)$, where j is given by

$$j = \left[i \times \frac{n-k}{m} \right].$$

The location of the minimum entry of the coefficient $\bar{a}_i$ is given by $(i+j+1, j+1)$, where

$$j = \begin{cases} 0 & i < \left[\frac{m}{2}\right], \\ 0 \text{ or } n-k & i = \frac{m}{2}, \\ n-k & \text{otherwise.} \end{cases}$$

Given this, the determination of $\bar{\mathcal{M}}_{1,i}$, $\bar{\mathcal{M}}_{2,i}$, $\bar{m}_{1,i}$ and $\bar{m}_{2,i}$ is of reduced complexity since the locations of the maximum and minimum entries are already known. The minimisation problem in (3.38) can now be written as

$$\begin{aligned} &\text{Minimise} \quad U - V \\ &\text{subject to} \\ &\begin{array}{rrrcll} U & & -i\bar{\phi} & \geq & \bar{\mathcal{M}}_{1,i} & i = 0, \ldots, m, \\ U & & -i\bar{\phi} \;\; -\bar{\mu} & \geq & \bar{\mathcal{M}}_{2,i} & i = 0, \ldots, n, \\ & -V & +i\bar{\phi} & \geq & -\bar{m}_{1,i} & i = 0, \ldots, m, \\ & -V & +i\bar{\phi} \;\; +\bar{\mu} & \geq & -\bar{m}_{2,i} & i = 0, \ldots, n. \end{array} \end{aligned}$$

In matrix form, this is given by

$$\text{Minimise} \quad \begin{bmatrix} 1 & -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} U \\ V \\ \bar{\phi}_k \\ \bar{\mu}_k \end{bmatrix} \quad \text{subject to} \quad A \begin{bmatrix} U \\ V \\ \bar{\phi}_k \\ \bar{\mu}_k \end{bmatrix} \geq \mathbf{b}. \tag{3.40}$$

The matrix $A \in \mathbb{R}^{(2m+2n+4)\times 4}$ in (3.40) is given by

$$A = \begin{bmatrix} \bar{\mathcal{A}}_1, & \bar{\mathcal{A}}_2, & \bar{a}_1, & \bar{a}_2 \end{bmatrix}^T,$$

where matrices $\bar{\mathcal{A}}_1$ and $\bar{a}_1 \in \mathbb{R}^{(m+1)\times 4}$ are given by

$$\bar{\mathcal{A}}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 1 & 0 & -2 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & -m & 0 \end{bmatrix}, \quad \bar{a}_1 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & -1 & 2 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & -1 & m & 0 \end{bmatrix}$$

and $\bar{\mathscr{A}}_2$ and $\bar{a}_2 \in \mathbb{R}^{(n+1)\times 4}$ are given by

$$\bar{\mathscr{A}}_2 = \begin{bmatrix} 1 & 0 & 0 & -1 \\ 1 & 0 & -1 & -1 \\ 1 & 0 & -2 & -1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & -n & -1 \end{bmatrix}, \quad \bar{a}_2 = \begin{bmatrix} 0 & -1 & 0 & 1 \\ 0 & -1 & 1 & 1 \\ 0 & -1 & 2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & -1 & n & 1 \end{bmatrix}.$$

The vector $\mathbf{b} \in \mathbb{R}^{2m+2n+4}$ in (3.40) is given by

$$\mathbf{b} = \left[\begin{array}{ccc|ccc|ccc|ccc} \bar{m}_{1,0}, & \ldots, & \bar{m}_{1,m} & \bar{m}_{2,0}, & \ldots, & \bar{m}_{2,n} & -\bar{m}_{1,0}, & \ldots, & -\bar{m}_{1,m} & -\bar{m}_{2,0}, & \ldots, & -\bar{m}_{2,n} \end{array} \right]^T,$$

where $\bar{m}_{i,j}$ and $\bar{m}_{i,j}$ are defined in (3.39).

The optimal values α_k and θ_k are given by $10^{\bar{\mu}}$ and $10^{\bar{\phi}}$ respectively, and the resulting preprocessed polynomials are given by

$$\tilde{f}_k(\omega) = \sum_{i=0}^{m} \bar{a}_i \theta_k^i \binom{m}{i} (1-\theta_k\omega)^{m-i}\,\omega^i, \quad \text{where} \quad \bar{a}_i = \frac{\hat{a}_i}{\mathcal{G}_{n-k}\Big(\hat{f}(x)\Big)},$$

$$\alpha_k\tilde{g}_k(\omega) = \alpha_k \sum_{i=0}^{n} \bar{b}_i \theta_k^i \binom{n}{i} (1-\theta_k\omega)^{n-i}\,\omega^i, \quad \text{where} \quad \bar{b}_i = \frac{\hat{b}_i}{\mathcal{G}_{m-k}\Big(\hat{g}(x)\Big)}.$$

The following examples consider the scaling of the coefficients of two preprocessed polynomials, and consequently the scaling of the entries of the sequence of subresultant matrices. The computation of the degree of the GCD is also considered in these examples, by analysing the singular values of each subresultant matrix for (i) unprocessed and (ii) preprocessed polynomials.

Example 3.4.1. This example considers the effect of preprocessing on the coefficients of the polynomials $\hat{f}(x)$ and $\hat{g}(x)$, whose preprocessed forms are given by $\tilde{f}_k(\omega)$ and $\alpha_k\tilde{g}_k(\omega)$. The scaling of entries in the first unprocessed and preprocessed subresultant matrix is also considered.

Consider the Bernstein form of exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ of degrees $m = 4$ and $n = 4$, whose factorised forms are given by

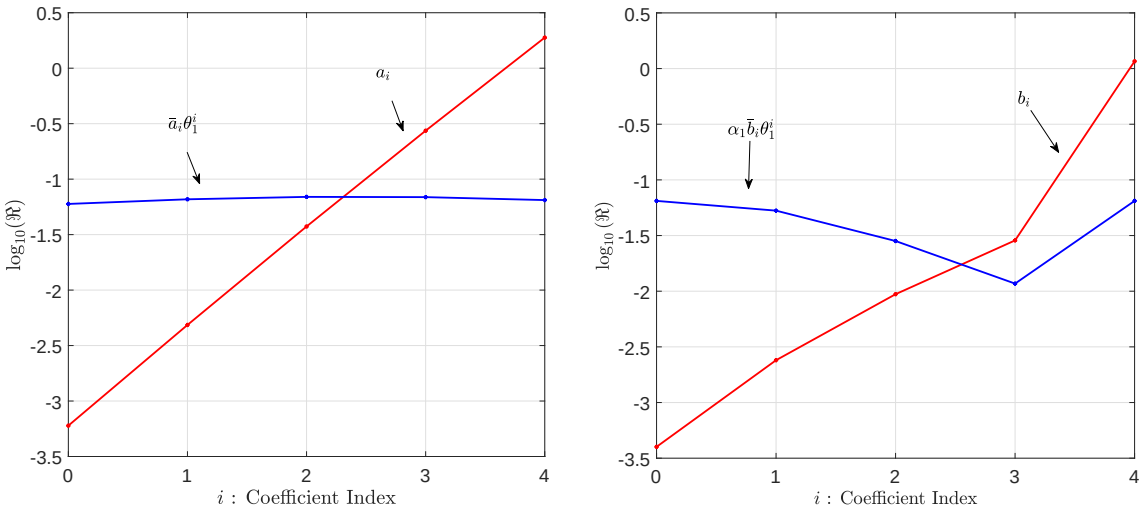
$$\hat{f}(x) = (x+0.1)^2(x+0.2)(x+0.3)$$
$$\hat{g}(x) = (x-0.2)(x+0.1)^2(x+0.2),$$

and whose GCD $\hat{d}_t(x)$ of degree $t = 2$ is given by

$$\hat{d}_t(x) = (x+0.1)^2.$$

Noise is added to the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$ to give the inexact polynomials $f(x)$ and $g(x)$, whose coefficients are given by (3.24), where $\{r_{f,i}\}$ and $\{r_{g,j}\}$ are uniformly distributed random variables in the interval $[-1, 1]$, and the set of values $\{\epsilon_{f,i}\}$ and $\{\epsilon_{g,j}\}$ are uniformly distributed random variables in the interval $\left[10^{-12}, 10^{-10}\right]$.

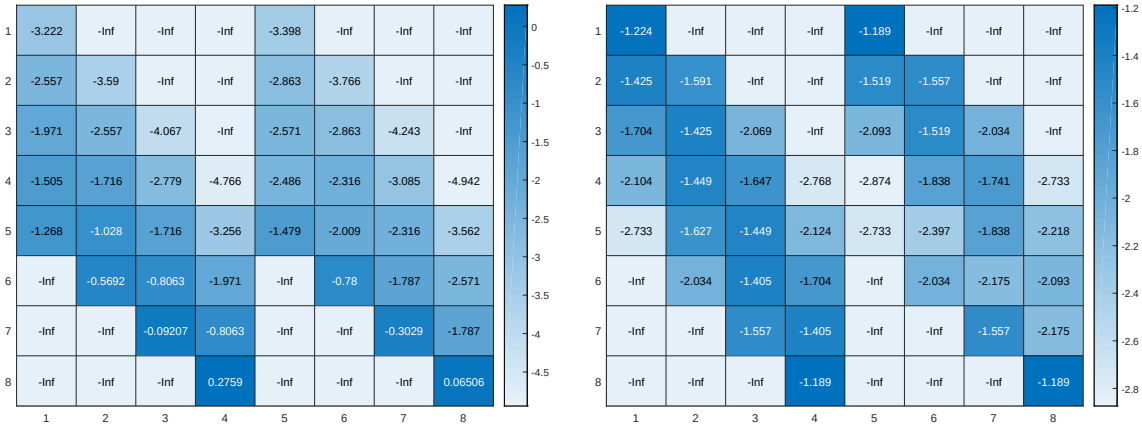
The polynomials $f(x)$ and $g(x)$ are preprocessed such that the preprocessed polynomi-



(i) Coefficients of $f(x)$ (●) and $\tilde{f}_1(\omega)$ (■)

(ii) Coefficients of $g(x)$ (●) and $\alpha_1\tilde{g}_1(\omega)$ (■)

Figure 3.7: The coefficients of both the unprocessed polynomials $f(x)$ and $g(x)$ and the preprocessed polynomials $\tilde{f}_1(\omega)$ and $\alpha_1\tilde{g}_1(\omega)$ in Example 3.4.1



(i) Entries in $S_1(f(x), g(x))$

(ii) Entries in $S_1(\tilde{f}_1(\omega), \alpha_1\tilde{g}_1(\omega))$

Figure 3.8: The absolute values of the entries of the (i) unprocessed and (ii) preprocessed Sylvester matrices in Example 3.4.1

als are given by $\tilde{f}_1(\omega)$ and $\alpha_1\tilde{g}_1(\omega)$ where $\alpha_1 = 0.6408$ and $\theta_1 = 0.1362$. The coefficients of the unprocessed and preprocessed polynomials are plotted in Figure 3.7, where it can be seen that the coefficients of the unprocessed polynomials span four orders of magnitude while the coefficients of the preprocessed polynomials only span one order of magnitude.

The overall effect of this scaling is seen in Figure 3.8, where heat maps of the log of the entries of the (i) unprocessed and (ii) preprocessed Sylvester matrix are shown. It is observed that the non-zero entries in the preprocessed subresultant matrix (Figure 3.8ii) span many fewer orders of magnitude than the unprocessed Sylvester matrix (Figure 3.8i). □

Example 3.4.2. This example considers the scaling of the polynomials $f(x)$ and $g(x)$, and the computation of the degree of the GCD by minimum singular values of the (i) unprocessed and (ii) preprocessed subresultant matrices.

Consider the Bernstein form of the exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ of degrees $m = 21$

and $n = 20$, whose factorised forms are given by

$$\hat{f}(x) = (x - 1.46)^2(x - 1.37)^3(x - 0.82)^3(x - 0.75)^3(x - 0.56)^4(x - 0.1)^3(x + 0.27)^3$$
$$\hat{g}(x) = (x - 2.12)(x - 1.37)^3(x - 1.2)^3(x - 0.99)^4(x - 0.75)^3(x - 0.56)^4(x - 0.1)^2.$$

The GCD $\hat{d}(x)$ of degree $t = 12$ in factorised form is given by

$$\hat{d}(x) = (x - 0.75)^3 (x - 0.1)^2 (x - 0.56)^4 (x - 1.37)^3 .$$

Noise is added to the coefficients of exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ such that the coefficients of the inexact polynomials $f(x)$ and $g(x)$ are given by (3.24), where the set of values $\{r_{f,i}\}$ and $\{r_{g,j}\}$ are uniformly distributed random variables in the interval $[-1, 1]$, and $\{\epsilon_{f,i}\}$ and $\{\epsilon_{g,j}\}$ are uniformly distributed random variables in the interval $\left[10^{-12}, 10^{-10}\right]$.

The polynomials $f(x)$ and $g(x)$ are preprocessed for each of the k subresultant matrices, thereby producing the sets $\{\, \tilde{f}_i(\omega) \mid i = 1, \ldots, 20\}$ and $\{\, \alpha_i \tilde{g}_i(\omega) \mid i = 1, \ldots, 20\}$ such that the preprocessed subresultant matrices are given by $\{\, S_k(\tilde{f}_k(\omega), \alpha_k \tilde{g}_k(\omega)) \mid k = 1, \ldots, 20\,\}$.

Coefficients of the unprocessed polynomials $f(x)$ and $g(x)$, and preprocessed polynomials $\tilde{f}_1(\omega)$ and $\alpha_1 \tilde{g}(\omega)$ are plotted in Figure 3.9. The coefficients of the unprocessed pair of polynomials span ≈ 12.5 orders of magnitude, while the coefficients of the pair of preprocessed polynomials only span ≈ 7 orders of magnitude.

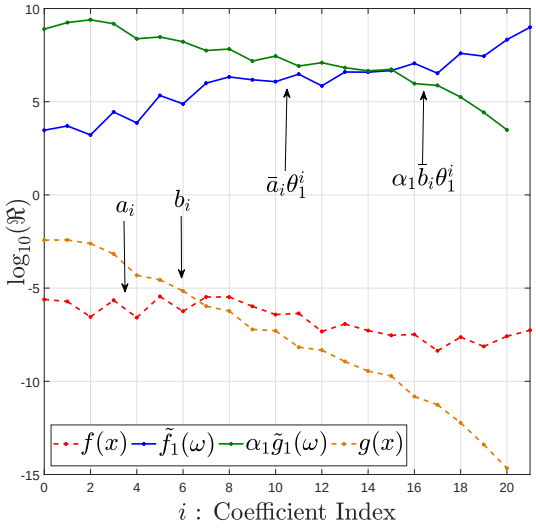


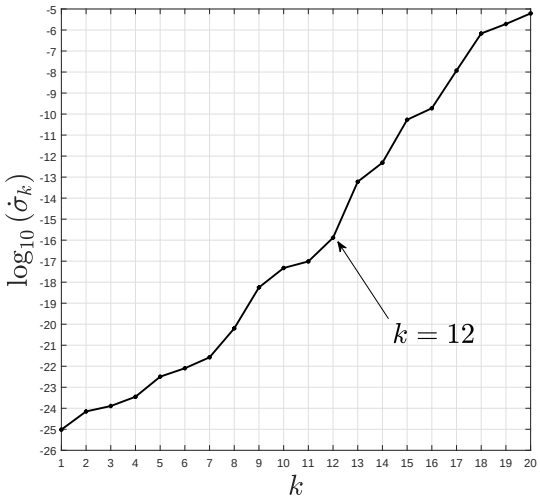
Figure 3.9: The coefficients of both the unprocessed polynomials $f(x)$ (●) and $g(x)$ (●) and the preprocessed polynomials $\tilde{f}_1(\omega)$ (●) and $\alpha_1\tilde{g}_1(\omega)$ (●) in Example 3.4.2

The degree of the GCD is computed using DC2, that is, the method of degree computation using minimum singular values described in Section 3.2.1. The sets of minimum singular values $\{\, \dot{\sigma}_k \mid k = 1, \ldots, 20\,\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices are plotted in Figure 3.10i and Figure 3.10ii respectively.

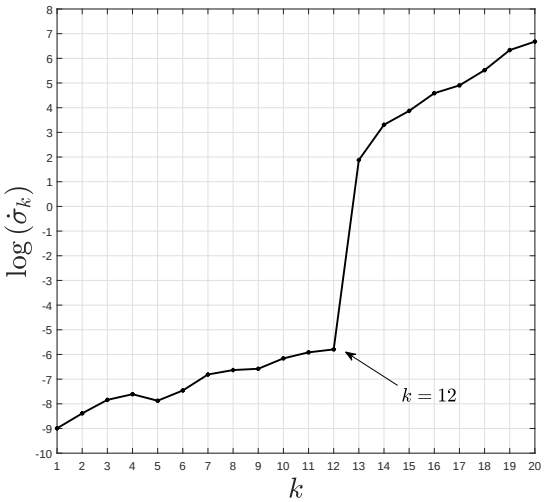
The degree of the GCD can with confidence be determined from the minimum singular values of the preprocessed subresultant matrices (Figure 3.10ii). There is a clear separation between the set of values $\{\, \dot{\rho}_i = \log_{10}(\dot{\sigma}_i) \mid i = 1, \ldots, 12\,\}$ and the set $\{\, \dot{\rho}_i = \log_{10}(\dot{\sigma}_i) \mid i = 13, \ldots, 20\}$. The two separate sets of numerically zero and non-zero minimum singular values are indicative of rank deficient and full rank subresultant matrices respectively.

The maximum change in the set $\{\dot{\rho}_k\}$ occurs between $k = 12$ and $k = 13$, that is, $\delta\dot{\rho}_{12}$ is maximal in the set $\{\delta\dot{\rho}_i \mid i = 1, \ldots, \min(m, n) - 1\}$. The degree of the GCD is therefore correctly identified as $t = 12$.

However, there is not a clear separation amongst the minimum singular values $\{\dot{\rho}_k = \log_{10}(\dot{\sigma}_k) \mid k = 1, \ldots, 20\}$ of the unprocessed subresultant matrices $\{S_k(f(x), g(x))\}$ in Figure 3.10i. By the same method, the index of maximum change in $\{\dot{\rho}_i\}$ is given by $i = 12$, but the value $\delta\dot{\rho}_{12}$ is not significantly larger than the other entries in the set $\{\delta\dot{\rho}_i\}$.



(i) The minimum singular values $\{\dot{\sigma}_k\}$ of the unprocessed subresultant matrices $\{S_k(f(x), g(x))\}$



(ii) The minimum singular values $\{\dot{\sigma}_k\}$ of the preprocessed subresultant matrices $\{S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega))\}$

Figure 3.10: The minimum singular values $\{\dot{\sigma}_k\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 3.4.2

□

Example 3.4.3. This example considers the scaling of the polynomial coefficients and the computation of the degree of the GCD using the complete set of all singular values for the (i) unprocessed and (ii) preprocessed subresultant matrices. The degrees of $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{d}(x)$, given by $m = 42$, $n = 39$ and $t = 28$, are significantly larger than in previous examples.

Consider the Bernstein form of the exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$, whose factorised forms are given by

$$\begin{aligned}\hat{f}(x) &= (x - 5.56)^8(x - 1.46)^4(x - 1.37)^3(x - 1.2435487954)^2(x - 0.82)^3\times \\ &\quad (x - 0.7515678)^{15}(x - 0.10122344)^4(x + 2.27564657)^3 \\ \hat{g}(x) &= (x - 5.56)^8(x - 2.12)^4(x - 1.37)^3(x - 1.2435487954)^2(x - 1.2222222)^3\times \\ &\quad (x - 0.99102445)^4(x - 0.7515678)^{15},\end{aligned}$$

and whose GCD $\hat{d}(x)$ of degree $t = 28$ is given by

$$\hat{d}(x) = (x - 5.56)^8(x - 1.37)^3(x - 1.2435487954)^2(x - 0.7515678)^{15}.$$

Random noise is added to the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$ in the same way as in the

previous example, and $\{\epsilon_{f,i}\}$ and $\{\epsilon_{g,j}\}$ are uniformly distributed random variables in the interval $\left[10^{-12}, 10^{-10}\right]$.

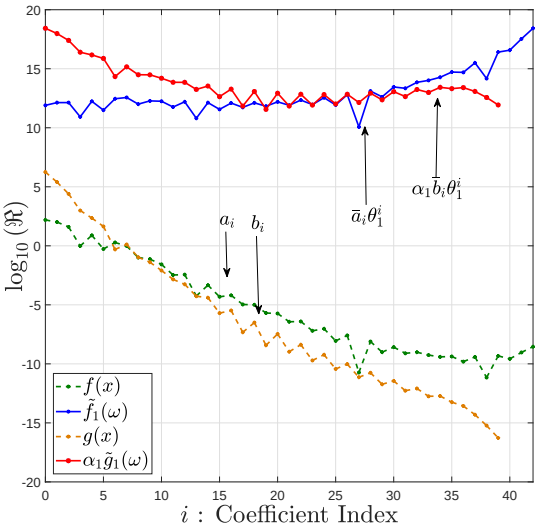


Figure 3.11: The coefficients of both the unprocessed polynomials $f(x)$ (●) and $g(x)$ (●) and the preprocessed polynomials $\tilde{f}_1(\omega)$ (●) and $\alpha_1\tilde{g}_1(\omega)$ (●) in Example 3.4.3

In Figure 3.11 the coefficients of the unprocessed and preprocessed polynomials $f(x)$, $g(x)$, $\tilde{f}_1(\omega)$ and $\alpha_1\tilde{g}_1(\omega)$ are plotted. The coefficients of the unprocessed polynomials $f(x)$ and $g(x)$ span approximately 20 orders of magnitude, while the coefficients of the preprocessed polynomials $\tilde{f}_1(\omega)$ and $\alpha_1\tilde{g}_1(\omega)$ only span ≈ 10 orders of magnitude.

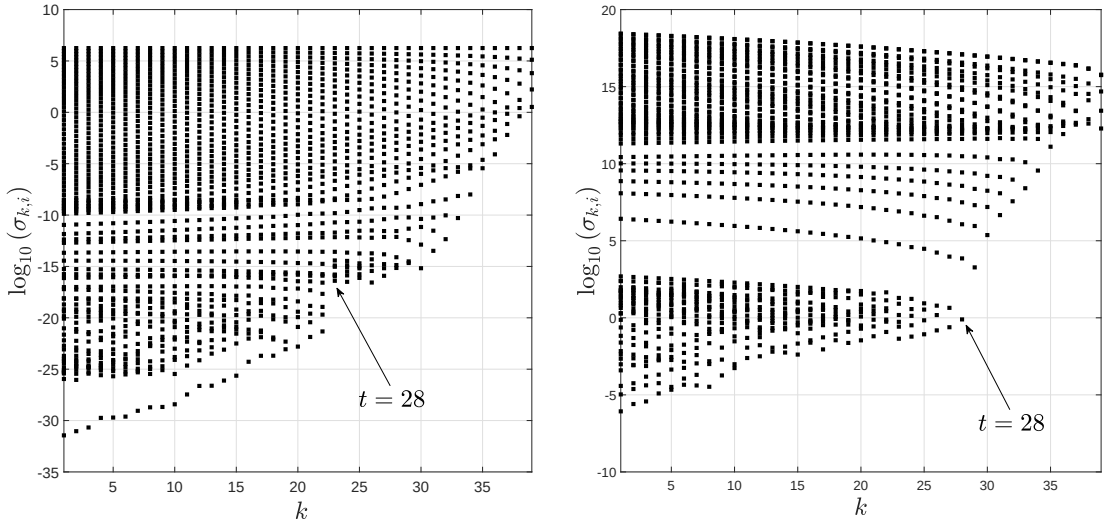
In Figure 3.12 the two complete sets of singular values of the (i) unprocessed and (ii) preprocessed subresultant matrices are plotted. It is not possible to determine the degree of the AGCD from the singular values of the unprocessed subresultant matrices $\{S_k(f(x), g(x))\}$ shown in Figure 3.12i. There is no clear separation between the numerically zero and non-zero singular values. Both DC1 and DC2 fail to compute the degree of the GCD.

However, in Figure 3.12ii there is a clear separation between the numerically zero and non-zero singular values of $\{S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega))\}$. By both methods DC1 and DC2, the set of subresultant matrices $\{\, S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega)) \mid k = 1, \ldots, 28 \,\}$ are identified as numerically singular, while the remaining subresultant matrices $\{\, S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega)) \mid k = 29, \ldots 39 \,\}$ are full rank. The degree of the AGCD is therefore correctly determined to be given by $t = 28$.

Results using the sets of singular values from the Sylvester matrix and the sequence of subresultant matrices are now compared with the singular values of the preprocessed Bézoutian matrix $\tilde{B}(f, g)$ obtained using the method described in [74]. The degree of the GCD of two polynomials is given by the rank loss of the Bézoutian matrix $B(f, g)$.

In Figure 3.13 the normalised singular values $\{\sigma_i/\sigma_1 \mid i = 1, \ldots, \max(m, n)\}$ of the Bézoutian matrix are plotted on a logarithmic scale. The degree of the GCD can be computed by a method similar to DC1. The SVD of the Bézoutian matrix $B(f, g)$ has $\max(m, n)$ singular values. The rank of $B(f, g)$ is given by the number of non-zero singular values and the degree of the GCD is given by $\max(m, n) - \text{rank}(B(f, g))$.

The degree of the GCD cannot, however, be determined from this set of singular values. Since it is known that the degree of the GCD is given by $t = 28$, a large negative change



(i) The singular values $\{\sigma_{k,i}\}$ of the unprocessed subresultant matrices $\{S_k(f(x), g(x))\}$

(ii) The singular values $\{\sigma_{k,i}\}$. of the preprocessed subresultant matrices $\{S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega))\}$

Figure 3.12: The singular values $\{\sigma_{k,i}\}$ of the sets of (i) unprocessed and (ii) preprocessed subresultant matrices in Example 3.4.3

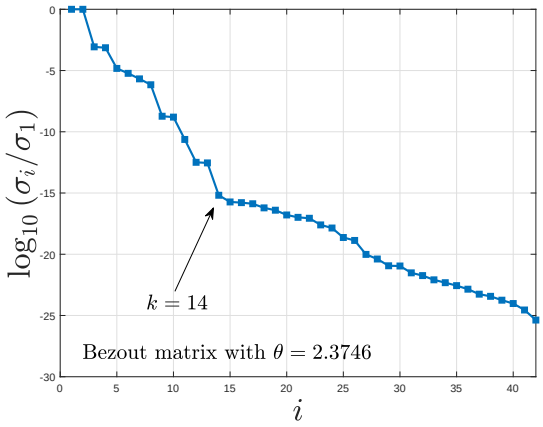


Figure 3.13: The normalised singular values $\{\sigma_i/\sigma_1\}$ of the Bernstein-Bézoutian matrix $\tilde{B}(f, g)$

would be expected between $\log_{10}(\sigma_{14}/\sigma_1)$ and $\log_{10}(\sigma_{15}/\sigma_1)$ to indicate that σ_{14} is the last numerically non-zero singular value.

□

Example 3.4.4. In Example 3.3.3 it was shown that $D_{m+n-k}^{-1}T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$ was the optimal variant of subresultant matrix for the computation of the degree of the GCD. The polynomials $f(x)$ and $g(x)$ for this example are given by adding noise to the coefficients of the Bernstein form of the exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ defined in Example 3.3.3

$$\hat{f}(x) = (x - 1.46)^2(x - 1.37)^3(x - 0.82)^3(x - 0.75)^3(x - 0.56)^8(x - 0.1)^3(x + 0.27)^3$$
$$\hat{g}(x) = (x - 2.12)(x - 1.37)^3(x - 1.2)^3(x - 0.99)^4(x - 0.75)^3(x - 0.56)^8(x - 0.1)^2.$$

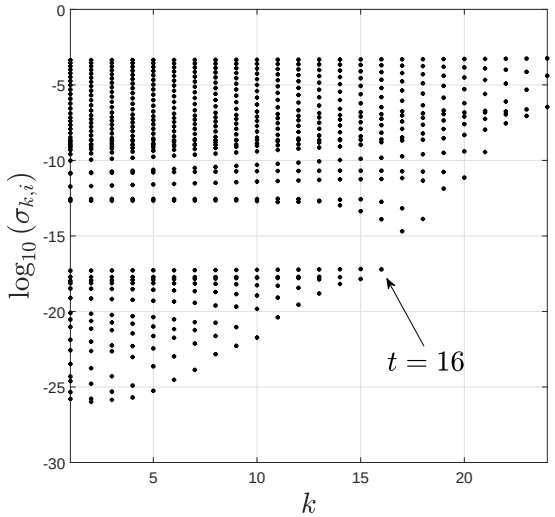
Again, $\{\epsilon_{f,i}\}$ and $\{\epsilon_{g,j}\}$ are uniformly distributed random variables in the interval $[10^{-12}, 10^{-10}]$. The inexact polynomials $f(x)$ and $g(x)$ are preprocessed to produce the sets of polynomials

$\{\tilde{f}_k(\omega) \mid k = 1, \ldots, \min(m,n)\}$ and $\{\alpha_k \tilde{g}_k(\omega) \mid k = 1, \ldots, \min(m,n)\}$.

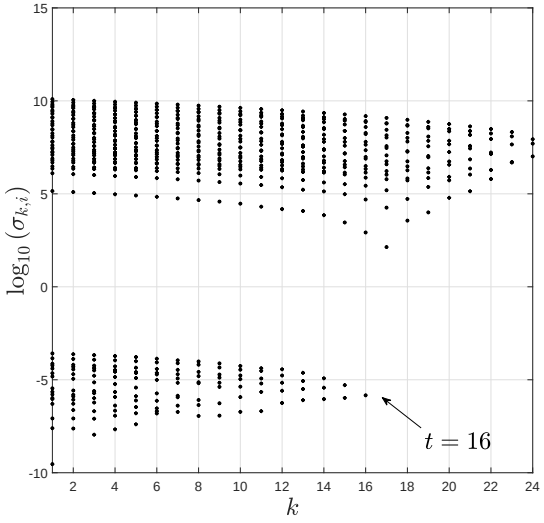
The singular values $\{\sigma_{k,i}\}$ of the (i) unprocessed subresultant matrices $\{S_k(f(x), g(x))\}$ and (ii) preprocessed subresultant matrices $\{S_k(\tilde{f}_k(\omega), \alpha_k \tilde{g}_k(\omega))\}$ are plotted in Figure 3.14i and Figure 3.14ii respectively. In Figure 3.14i there is a clear separation between the numerically zero and non-zero singular values of each subresultant matrix. However, the last numerically zero singular value of $S_{16}(f(x), g(x))$ is similar to the smallest non-zero singular value of $S_{17}(f(x), g(x))$, that is, $\dot{\rho}_{16} \approx \dot{\rho}_{17}$ where $\dot{\rho}_i = \log_{10}(\dot{\sigma}_i)$.

This is easier to see in Figure 3.15, where only the minimum singular values of each subresultant matrix are plotted. Consider the use of the the method DC2 for the computation of the degree of the GCD by the set of minimum singular values as described in Section 3.2.1. Let $\delta\dot{\rho}_i$ be the change between any two consecutive $\dot{\rho}_i$ and $\dot{\rho}_{i+1}$ as defined in (3.16). Since the degree of the GCD is given by $t = 16$, theoretically, $\max\{\delta\rho_i\}$ is given by $\delta\dot{\rho}_{16}$. While this is true, $\delta\dot{\rho}_{16}$ is not significantly larger than all other $\delta\dot{\rho}_i$ in the set $\{\dot{\rho}_i\}$, that is, the change between $\dot{\rho}_{16}$ and $\dot{\rho}_{17}$ is not significantly larger than the change between any other consecutive $\dot{\rho}_i$ and $\dot{\rho}_{i+1}$.

This is in contrast to the clearly defined separation between the rank deficient and full rank preprocessed subresultant matrices whose singular values are plotted in Figure 3.14ii and whose minimimum singular values are plotted in Figure 3.15ii. The degree of the GCD can be computed by (i) the set of singular values of the first subresultant matrix, (ii) the maximum change in magnitude of the minimum singular values of each subresultant matrix or (iii) observation of the complete set of singular values of all subresultant matrices.



(i) The singular values $\{\sigma_{k,i}\}$ of the unprocessed subresultant matrices $\{S_k(f(x), g(x))\}$

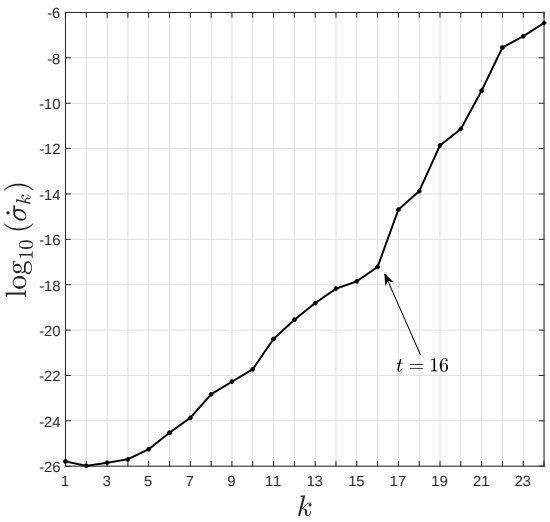


(ii) The singular values $\{\sigma_{k,i}\}$ of the preprocessed subresultant matrices $\{S_k(\tilde{f}_k(\omega), \alpha_k \tilde{g}_k(\omega))\}$

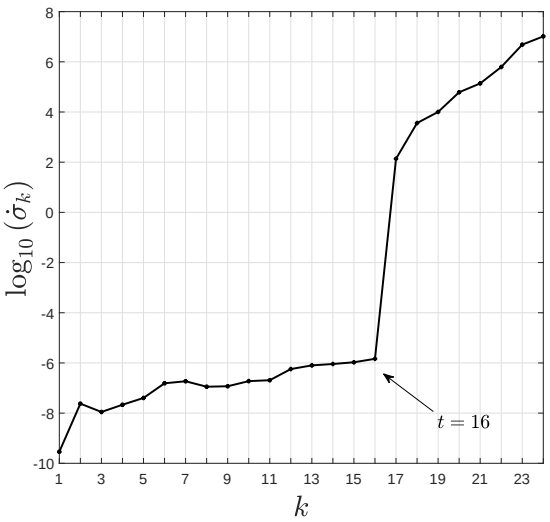
Figure 3.14: The singular values $\{\sigma_{k,i}\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 3.4.4

□

Example 3.4.5. Consider the Bernstein form of the exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ of



(i) The minimum singular values $\{\dot{\sigma}_k\}$ of the unprocessed subresultant matrices $\{S_k(f(x), g(x))\}$



(ii) The minimum singular values $\{\dot{\sigma}_k\}$ of the preprocessed subresultant matrices $\{S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega))\}$

Figure 3.15: The minimum singular values $\{\dot{\sigma}_k\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 3.4.4

degrees $m = 35$ and $n = 31$, whose factorisations are given by

$$\hat{f}(x) = (x - 1.46)^2(x - 1.37)^3(x - 0.82)^3(x - 0.75)^{10}(x - 0.56)^8(x - 0.1)^6(x + 0.27)^3$$
$$\hat{g}(x) = (x - 2.12)(x - 1.37)^3(x - 1.2)^3(x - 0.99)^4(x - 0.75)^{10}(x - 0.56)^8(x - 0.1)^2.$$

The factorised form of the GCD $\hat{d}(x)$ of degree $t = 23$ is given by

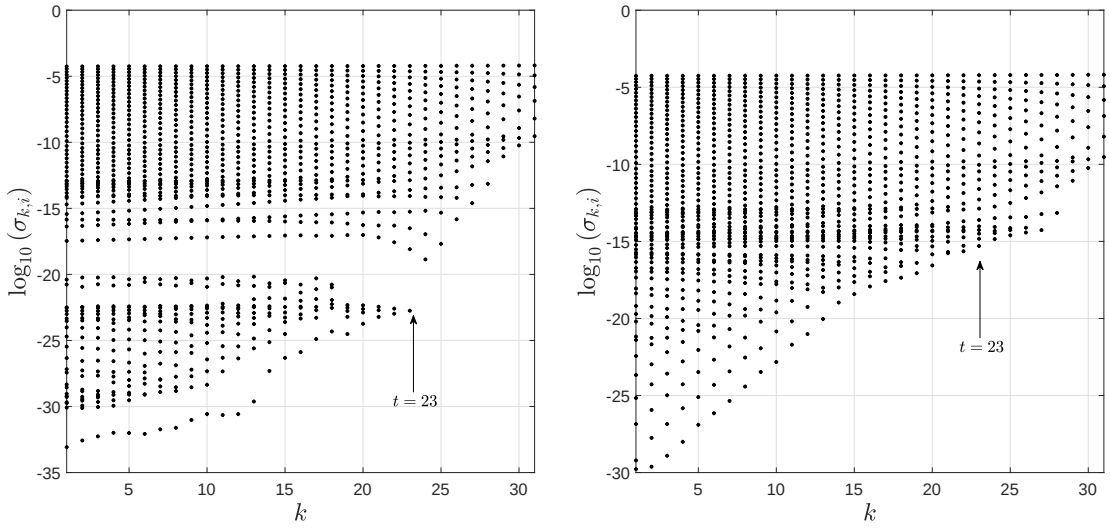
$$\hat{d}(x) = (x - 1.37)^3(x - 0.75)^{10}(x - 0.56)^8(x - 0.1)^2.$$

Noise is added to the coefficients of the exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$, and the coefficients of the inexact polynomials $f(x)$ and $g(x)$ are given by (3.24), where $\{\epsilon_{f,i}\}$ and $\{\epsilon_{g,j}\}$ are initially set equal to 10^{-12} and in the second experiment set equal to 10^{-4}. The sets of values $\{r_{f,i}\}$ and $\{r_{g,j}\}$ are uniformly distributed random variables in the interval $[-1, 1]$.

This example shows that the degree of the GCD of two polynomials can be computed by analysis of the singular values of the set of subresultant matrices, and that in the presence of noise this computation is less likely to succeed. However, preprocessing the polynomials typically allows for the recovery of the degree of the GCD where the use of unprocessed polynomials would otherwise fail.

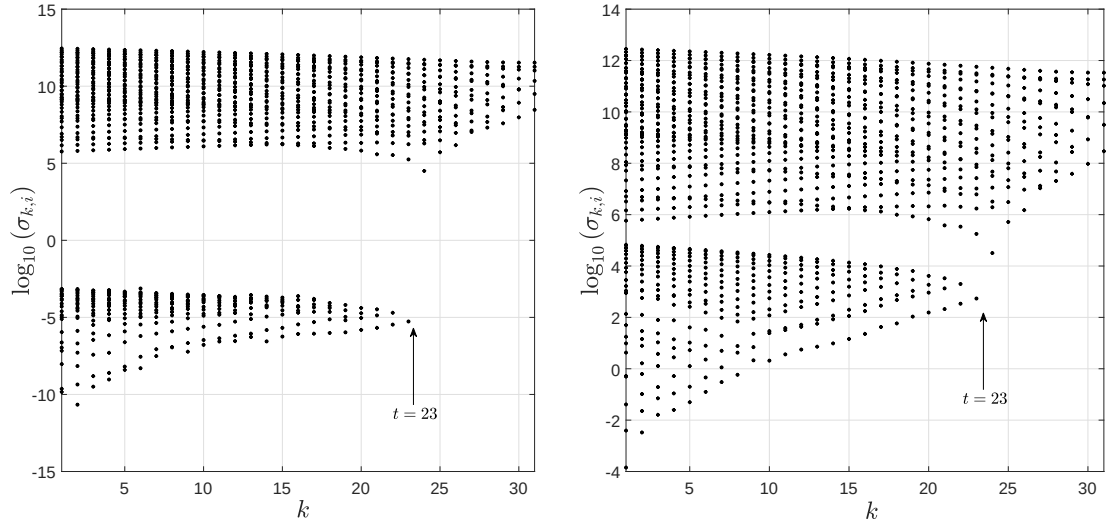
The sets of singular values of each subresultant matrix of the (i) unprocessed and (ii) preprocessed polynomials are computed at two noise levels. The singular values of the subresultant matrices $\{S_k(f(x), g(x))\}$ containing the coefficients of the unprocessed polynomials are plotted in Figure 3.16. Meanwhile, the singular values of the set of subresultant matrices $\{S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega))\}$ containing the coefficients of the preprocessed polynomials are plotted in Figure 3.17. From these graphs, it can be seen that the singular values of the subresultant matrices of preprocessed polynomials give a clearer indication of the degree of the GCD even at the highest noise levels, while there is minimal separation

between the non-zero and numerically zero singular values of the subresultant matrices of unprocessed polynomials even at low noise levels.



(i) $\{\epsilon_{f,i}\} = \{\epsilon_{g,j}\} = 10^{-12}$ **(ii)** $\{\epsilon_{f,i}\} = \{\epsilon_{g,j}\} = 10^{-4}$

Figure 3.16: The singular values $\{\sigma_{k,i}\}$ of the unprocessed subresultant matrices $\{S_k(f(x), g(x))\}$ for noise at levels (i) 10^{-12} and (ii) 10^{-4} in Example 3.4.5



(i) $\{\epsilon_{f,i}\} = \{\epsilon_{g,j}\} = 10^{-12}$ **(ii)** $\{\epsilon_{f,i}\} = \{\epsilon_{g,j}\} = 10^{-4}$

Figure 3.17: The singular values $\{\sigma_{k,i}\}$ of the preprocessed subresultant matrices $\{S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega))\}$ for noise at levels (i) 10^{-12} and (ii) 10^{-4} in Example 3.4.5

The singular values of the preprocessed Bézoutian matrix $\tilde{B}(f, g)$ are plotted in Figure 3.18, where f and g are inexact polynomials with $\{\epsilon_{f,i}\} = \{\epsilon_{g,j}\} = 10^{-4}$. The Bézoutian matrix has $\max(m, n)$ singular values, where the subset $\{\sigma_k \mid 1, \ldots, \max(m, n) - t\}$ are non-zero and the subset $\{\sigma_k \mid \max(m, n) - t + 1 \ldots \max(m, n)\}$ are numerically zero. The degree of the GCD is therefore given by the rank loss of $\tilde{B}(f, g)$ $t = \max(m, n) - k^*$, where k^* is the index of the last non-zero singular value.

Since it is known that the degree of the GCD is given by $t = 23$, it would be expected that the last non-zero singular value would be σ_{12} and the first numerically zero singular value would be σ_{13}.

As in earlier examples, let the change in magnitude of the log of the singular values be denoted $\{\delta\rho_i = \rho_i - \rho_{i+1} \mid i = 1, \ldots, \max(m,n)\}$, where $\rho_i = \log_{10}(\sigma_i)$. Then the degree of the AGCD is given by

$$\arg_i \max\{\, \delta\rho_i \mid i = 1, \ldots, \max(m,n)\,\} \tag{3.41}$$

While the change between the 12th and 13th singular value, $\delta\rho_{12}$, is significant, a larger change is given between the 2nd and 3rd singular values ($\delta\rho_2$), that is $\delta\rho_2 > \delta\rho_{12}$. Consequently, the degree of the AGCD is incorrectly determined.

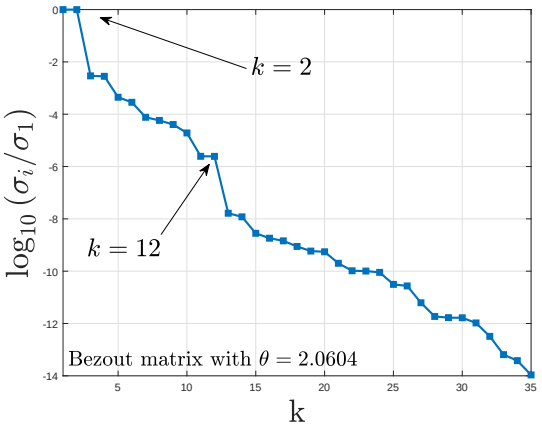


Figure 3.18: The normalised singular values $\{\sigma_i/\sigma_1\}$ of the preprocessed Bézoutian matrix $B(f,g)$ in Example 3.4.5

□

Example 3.4.6. Consider the Bernstein form of the exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$, whose factorisations are given by

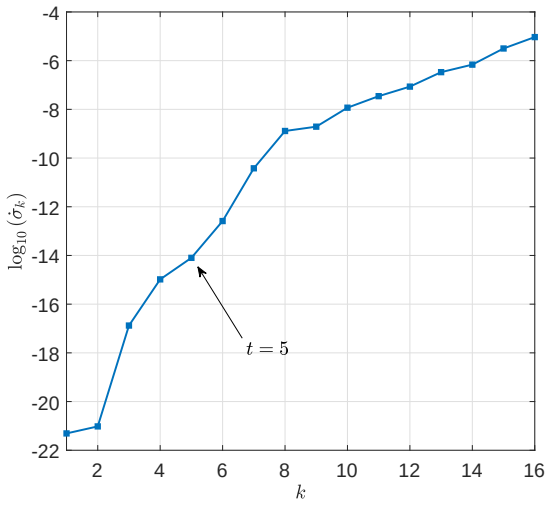
$$\hat{f}(x) = (x-3.4)^3(x-2.5)^3(x-0.8)^2(x-0.7)^3(x-0.5)^2(x-0.3)^2(x-0.1)^4$$
$$\hat{g}(x) = (x-1.1)^3(x-0.9)^4(x-0.85)^4(x-0.8)^2(x-0.1)^3$$

and whose GCD $\hat{d}(x)$ has the factorisation

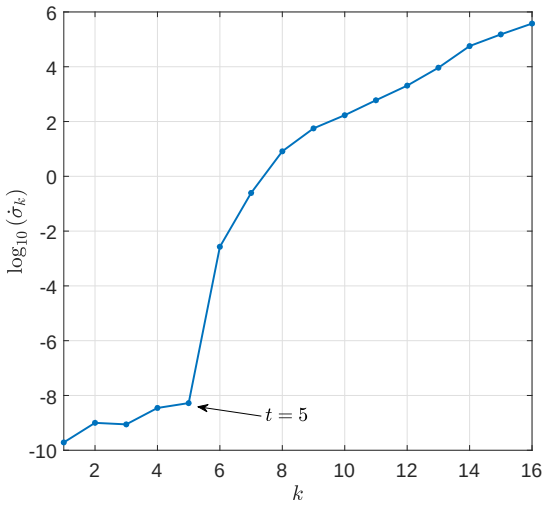
$$\hat{d}(x) = (x-0.8)^2(x-0.1)^3.$$

Variable noise is added to the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$ and the inexact polynomials $f(x)$ and $g(x)$ have the coefficients given by (3.24), where $\{\epsilon_{f,i}\}$ and $\{\epsilon_{g,j}\}$ are uniformly distributed random variables in the interval $\left[10^{-12}, 10^{-10}\right]$, and $\{r_{f,i}\}$ and $\{r_{g,j}\}$ are uniformity distributed random variables in the interval $[-1,1]$.

The set of minimum singular values $\{\,\dot{\sigma}_i \mid i = 1, \ldots, 16\,\}$ of the subresultant matrices of the unprocessed polynomials $\{\,S_k(f(x), g(x)\,\}$ are shown in Figure 3.19i. From this set of values, the degree of the AGCD is incorrectly computed by DC2, which identifies the degree of the AGCD as $t = 2$. However, better results are obtained from the preprocessed subresultant matrices. The set of minimum singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices of the preprocessed polynomials, $\{S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega))\}$, are shown in Figure 3.19ii. There is a distinct separation between the numerically zero and non-zero minimum sin-



(i) The minimum singular values $\{\dot{\sigma}_k\}$ of the unprocessed subresultant matrices $\{S_k(f(x), g(x))\}$



(ii) The minimum singular values $\{\dot{\sigma}_k\}$ of the preprocessed subresultant matrices $\{S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega))\}$

Figure 3.19: The minimum singular values $\{\dot{\sigma}_k\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 3.4.6

gular values. The degree of the GCD is given by $\max(\{\delta\dot{\rho}_i\})$ and is correctly identified as $t = 5$.

□

Section 3.4 has considered the computation of the degree of the GCD of two polynomials, $\hat{f}(x)$ and $\hat{g}(x)$. Three preprocessing operations have been introduced, and efficient methods for their computation have been developed.

The numerical rank of each matrix in the set of subresultant matrices of two preprocessed polynomials has been shown to be better defined than the numerical rank of each of the subresultant matrices of the equivalent unprocessed polynomials.

It has been shown by Example 3.4.2 and Example 3.4.3 that preprocessing inexact polynomials allows the recovery of the AGCD degree by analysis of the preprocessed subresultant matrices, where the subresultant matrices of the equivalent unprocessed polynomials would otherwise fail.

The methods described in this work have been compared with a method which makes use of the Bézoutian matrix [72], which has typically failed for examples with high levels of noise.

The second stage of computing the AGCD of two polynomials is to compute its coefficients. Using the tth subresultant matrix, the coefficients of the cofactor polynomials $\hat{u}_t(x)$ and $\hat{v}_t(x)$ and the coefficients of the GCD $\hat{d}_t(x)$ can be approximated. Methods for the computation of these approximations are described in the next section.

3.5 The Coefficients of Cofactor Polynomials and Matrix Low Rank Approximations

The previous section focused on the computation of the degree t of the AGCD of two univariate polynomials, $f(x)$ and $g(x)$, and showed that satisfactory results are obtained

when the correct subresultant matrix variant is considered and the polynomials are preprocessed. This section now considers an initial approximation of the coefficients of the GCD from the subresultant matrix $S_t(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$.

3.5.1 The Coefficients of Cofactor Polynomials by Least Squares

The rank of the tth subresultant matrix $S_t(\hat{f}(x), \hat{g}(x)) \in \mathbb{R}^{(m+n-t+1)\times(m+n-2t+2)}$ for the exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ is equal to $m+n-2t+1$, that is, the matrix has a rank loss of one. Therefore, there is exactly one column of the matrix which defines the linear dependence of all other columns. However, this property does not extend to the tth subresultant matrix of the inexact preprocessed polynomials $S_t(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$, which is of full rank. Instead, $S_t(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$ is considered to be numerically rank deficient, and one of its columns lies in the space spanned by the remaining columns with minimal residual.

The columns of $S_t(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$ are given by the set of vectors

$$\{\mathbf{c}_{t,j} \mid j = 0, \ldots, m+n-2t+1\}$$

such that the tth subresultant matrix is given by

$$S_t\left(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega)\right) = \left[\begin{array}{cccc|ccc} \mathbf{c}_{t,0}, & \ldots, & \mathbf{c}_{t,n-t}, & & \mathbf{c}_{t,n-t+1}, & \ldots, & \mathbf{c}_{t,m+n-2t+1} \end{array}\right].$$

To determine which column lies in the space spanned by the others with minimum error, each column is removed in turn to form the set of approximate equations

$$\{ A_{t,j}\left(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega)\right)\mathbf{x}_{t,j} \approx \mathbf{c}_{t,j} \mid j = 0, \ldots, m+n-2t+1\},$$

where $A_{t,j}(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$ is the tth subresultant matrix $S_t(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$ with the column $\mathbf{c}_{t,j}$ removed, and the associated residuals are given by

$$r_{t,j} = \left\|\mathbf{c}_{t,j} - A_{t,j}\left(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega)\right)\mathbf{x}_{t,j}\right\| \qquad \text{for} \quad j = 0, \ldots, m+n-2t+1.$$

The set of residuals associated with these approximate equations is denoted $\{ r_{t,j} \mid j = 0, \ldots, m+n-2t+1\}$. Let q be the index of the column removed with minimal residual

$$q = \arg_j \min\{ r_{t,j} \mid j = 1, \ldots, m+n-2t+2\},$$

then it is the column $\mathbf{c}_{t,q}$ which is most likely to be nearly linearly dependent on the other columns. The approximate equation of interest for the low rank approximation problem is therefore given by

$$A_{t,q}\left(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega)\right)\mathbf{x}_{t,q} \approx \mathbf{c}_{t,q}. \tag{3.42}$$

Let M_q be defined as

$$M_q = \left[\begin{array}{ccccccc} \mathbf{e}_1, & \mathbf{e}_2, & \ldots, & \mathbf{e}_{q-1}, & \mathbf{e}_{q+1}, & \ldots, & \mathbf{e}_{m+n-2t+2} \end{array}\right],$$

where $\mathbf{e}_i \in \mathbb{R}^{m+n-2t+2}$ is the ith unit basis vector, then the approximation (3.42) can be written as

$$\left(D_{m+n-t}^{-1} T_t\left(\tilde{f}_t(\omega), \alpha_t \tilde{g}_t(\omega)\right) \hat{Q}_t\right) M_q \mathbf{x}_{t,q} \approx \left(D_{m+n-t}^{-1} T_t\left(\tilde{f}_t(\omega), \alpha_t \tilde{g}_t(\omega)\right)\right) \mathbf{e}_q. \qquad (3.43)$$

The vector $\mathbf{x}_{t,q} \in \mathbb{R}^{(m+n-2t+1)}$ in (3.42), given by

$$\mathbf{x}_{t,q} = \left[\begin{array}{cccc} x_0, & x_1, & \ldots, & x_{m+n-2t} \end{array}\right]^T \in \mathbb{R}^{m+n-2t+1},$$

is obtained by a simple least squares based method.

The insertion of '-1' into the qth position of the vector $\mathbf{x}_{t,q}$ gives the vector $\mathbf{x}_t \in \mathbb{R}^{(m+n-2t+2)}$

$$\mathbf{x}_t = \left[\begin{array}{ccccccc} x_0, & x_1, & \ldots, & x_{q-2}, & -1, & x_{q-1}, & \ldots, & x_{m+n-2t+1} \end{array}\right]^T$$

such that

$$S_t\left(\tilde{f}_t(\omega), \alpha_t \tilde{g}_t(\omega)\right) \mathbf{x}_t \approx \mathbf{0}.$$

The vector $\mathbf{x}_t$ contains the coefficients of the cofactor polynomials $\tilde{v}_t(\omega)$ and $\tilde{u}_t(\omega)$

$$\mathbf{x}_t = \left[\begin{array}{cccc|ccc} \bar{v}_0, & \bar{v}_1\theta, & \ldots, & \bar{v}_{n-t}\theta^{n-t}, & -\bar{u}_0, & -\bar{u}_1\theta, & \ldots & -\bar{u}_{m-t}\theta^{m-t} \end{array}\right].$$

Having computed approximations of the coefficients of the cofactor polynomials $\tilde{v}_t(\omega)$ and $\tilde{u}_t(\omega)$, the coefficients of the polynomial $\tilde{d}_t(\omega)$ are approximated as the least squares solution of the system of equations

$$\left(\left[\begin{array}{cc} D_m^{-1} & 0 \\ 0 & D_n^{-1} \end{array}\right] \left[\begin{array}{c} T_t\left(\tilde{u}_t(\omega)\right) \\ T_t\left(\tilde{v}_t(\omega)\right) \end{array}\right] Q_t\right) \tilde{\mathbf{d}}_t \approx \left[\begin{array}{c} \tilde{\mathbf{f}} \\ \alpha \tilde{\mathbf{g}} \end{array}\right]$$

and the vector $\tilde{\mathbf{d}}_t \in \mathbb{R}^{t+1}$ is given by

$$\tilde{\mathbf{d}}_t = \left[\begin{array}{cccc} d_0, & d_1\theta, & \ldots, & d_t\theta^t \end{array}\right]^T,$$

where $d_i\theta^i$ are the coefficients of the polynomial $\tilde{d}_t(\omega)$.

The least squares solution only gives an approximation of the polynomials $\tilde{u}_t(\omega)$ and $\tilde{v}_t(\omega)$ since $\tilde{f}_t(\omega)$ and $\alpha_t\tilde{g}_t(\omega)$ are subject to noise. However, structure can be added to the subresultant matrix $S_t(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$, such that a low rank approximation is obtained. The polynomials $\tilde{f}_t(\omega) + \delta\tilde{f}_t(\omega)$ and $\tilde{g}_t(\omega) + \delta\tilde{g}_t(\omega)$ are obtained by perturbing $\tilde{f}_t(\omega)$ and $\tilde{g}_t(\omega)$ such that $S_t(\tilde{f}_t + \delta\tilde{f}_t, \alpha_t(\tilde{g}_t + \delta\tilde{g}_t))$ is a low rank approximation of $S_t(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t)$. This added structure results in improved approximations of the cofactor polynomials and the coefficients of the GCD. This method of determining the low rank approximation is now described.

3.5.2 The Coefficients of Cofactor Polynomials by STLN

A structured low rank approximation can be obtained by structured total least norm (STLN) algorithms [55], specifically applied to the Sylvester matrix [40, 44, 68]. In [69] the linear problem of computing minimal perturbations of the coefficients of f and g is transformed to a non-linear problem by also considering perturbations of α_t and θ_t which have previously been determined to be the optimal values in preprocessing of the tth subresultant matrix.

In this section a method of structured non-linear total least norm (SNTLN) is developed for the computation of a low rank approximation of the tth subresultant matrix $S_t(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$, from which the coefficients of the cofactor polynomials $\tilde{u}_t(\omega)$ and $\tilde{v}_t(\omega)$ are computed.

The approximate equation (3.42) is replaced by an exact equation given by the addition of a structured matrix $B_{t,q}$ and vector $\mathbf{h}_{t,q}$, such that an exact equation is given by

$$(A_{t,q} + B_{t,q})\,\mathbf{x}_{t,q} = \mathbf{c}_{t,q} + \mathbf{h}_{t,q}. \tag{3.44}$$

The matrix $B_{t,q}$ and vector $\mathbf{h}_{t,q}$ are structured in the same way as $A_{t,q}$ and $\mathbf{c}_{t,q}$ respectively, and contain coefficients of polynomials to be added to the polynomials $\tilde{f}_t(\omega)$ and $\alpha_t\tilde{g}_t(\omega)$. The matrix $B_{t,q}$ and vector $\mathbf{h}_{t,q}$ are not unique, since there are infinitely many sets of perturbations such that the polynomials $\tilde{f}_t(\omega) + \delta\tilde{f}_t(\omega)$ and $\tilde{g}_t(\omega) + \delta\tilde{g}_t(\omega)$ have a non-constant polynomial GCD. It is therefore necessary to constrain the problem to find the minimal perturbations $\delta\tilde{f}(\omega)$ and $\delta\tilde{g}(\omega)$ and the problem is solved by least squares with equality (LSE)

$$\min\{\|B_{t,q}\|^2 + \|\mathbf{h}_{t,q}\|^2\} \quad \text{such that} \quad (A_{t,q} + B_{t,q})\,\mathbf{x}_{t,q} = \mathbf{c}_{t,q} + \mathbf{h}_{t,q}. \tag{3.45}$$

As stated above, the solution to (3.44) is not unique, but (3.45) ensures that the polynomials $\tilde{f}_t(\omega)$ and $\tilde{g}_t(\omega)$ are perturbed by the minimum amount to induce a common divisor of degree t, such that the problem has a unique solution.

The Subresultant Matrix, Structured Perturbations and a Low Rank Approximation Method

The polynomials $\tilde{f}_t(\omega)$ and $\alpha_t\tilde{g}_t(\omega)$ are highly likely to be coprime, and these are perturbed to induce a non-constant common divisor in their perturbed forms. The polynomials added to $\tilde{f}_t(\omega)$ and $\alpha_t\tilde{g}_t(\omega)$ are given by

$$\tilde{s}_t(\omega) = \sum_{i=0}^{m} z_{1,i}\phi^i \binom{m}{i} (1-\phi\omega)^{m-i}\omega^i$$

$$\text{and} \qquad \beta\tilde{t}_t(\omega) = \beta\sum_{i=0}^{n} z_{2,i}\phi^i \binom{n}{i} (1-\phi\omega)^{n-i}\omega^i$$

respectively.

The matrix $B_t(\phi, \beta, \mathbf{z})$ is used to describe the structured matrix containing the coefficients of the polynomials $\tilde{s}(\phi, \omega)$ and $\beta\tilde{t}(\phi, \omega)$, which is equivalent in structure to

$S_t(\tilde{f}(\phi,\omega),\beta\tilde{g}(\phi,\omega))$ and is given by

$$B_t\left(\beta,\phi,\mathbf{z}\right)=D_{m+n-t}^{-1}F_t\left(\beta,\phi,\mathbf{z}\right)\hat{Q}_t\in\mathbb{R}^{(m+n-t+1)\times(m+n-2t+2)}.$$

The matrices $D_{m+n-t}^{-1}\in\mathbb{R}^{(m+n-t)\times(m+n-t)}$ and $\hat{Q}_t\in\mathbb{R}^{(m+n-2t+2)\times(m+n-2t+2)}$ are already defined in (3.6) and (3.7) respectively, and the matrix $F_t(\beta,\phi,\mathbf{z})$ is given by

$$\begin{aligned}F_t(\beta,\phi,\mathbf{z})&=\left[\begin{array}{c|c}T_{n-t}\left(\tilde{s}(\phi,\omega)\right)&T_{m-t}\left(\beta\tilde{t}(\phi,\omega)\right)\end{array}\right]\\&=\left[\begin{array}{ccc|ccc}z_{1,0}\binom{m}{0}&&&\beta z_{2,0}\binom{n}{0}&&\\z_{1,1}\phi\binom{m}{1}&\ddots&&\beta z_{2,1}\phi\binom{n}{1}&\ddots&\\\vdots&\ddots&z_0\binom{m}{0}&\vdots&\ddots&\beta z_{2,0}\binom{n}{0}\\z_{1,m}\phi^m\binom{m}{m}&\ddots&z_{1,1}\phi^1\binom{m}{1}&\beta z_{2,n}\phi^n\binom{n}{n}&\ddots&\beta z_{2,1}\phi^1\binom{n}{1}\\&\ddots&\vdots&&\ddots&\vdots\\&&z_{1,m}\phi^m\binom{m}{m}&&&\beta z_{2,n}\phi^n\binom{n}{n}\end{array}\right].\end{aligned}$$

The perturbation of the coefficients of $\tilde{f}(\phi,\omega)$ and $\alpha_t\tilde{g}(\phi,\omega)$ implies that the equation (3.44) is given by

$$\left(D_{m+n-t}^{-1}\left(T_t+F_t\right)\hat{Q}_t\right)M_{t,}\mathbf{x}_{t,q}=\left(D_{m+n-t}^{-1}\left(T_t+F_t\right)\hat{Q}_t\right)\mathbf{e}_q.\tag{3.46}$$

A change in notation is required since (3.46) is a non-linear equation solved iteratively, and the variables to be determined are β, ϕ and $\mathbf{z}$. The initial values are given by

$$\beta^{(0)}=\alpha_t,\quad\phi^{(0)}=\theta_t\quad\text{and}\quad\mathbf{z}^{(0)}=\mathbf{0}_{m+n+2}.\tag{3.47}$$

These variables are now included in the arguments of the vectors and matrices in the expression (3.46), which is now given by

$$\left(D_{m+n-t}^{-1}\left(T_t\left(\beta,\phi\right)+F_t\left(\beta,\phi,\mathbf{z}\right)\right)\hat{Q}_t\right)M_{t,q}\mathbf{x}_{t,q}=\mathbf{c}_{t,q}\left(\beta,\phi\right)+\mathbf{h}_{t,q}\left(\beta,\phi,\mathbf{z}\right),\tag{3.48}$$

where

$$\mathbf{c}_{t,q}(\beta,\phi)=D_{m+n-t}^{-1}T_t\left(\beta,\phi\right)\hat{Q}_t\mathbf{e}_q\quad\text{and}\quad\mathbf{h}_{t,q}(\beta,\phi,\mathbf{z})=D_{m+n-t}^{-1}F_t\left(\beta,\phi,\mathbf{z}\right)\hat{Q}_t\mathbf{e}_q.$$

The variable β is included in the arguments of $\mathbf{c}_{t,q}$ and $\mathbf{h}_{t,q}$, but it is possible that these vectors may not be functions of β. The remainder of this section assumes that $\mathbf{c}_{t,q}$ is a column removed from the second partition of the tth subresultant matrix, and $n-t+1\leq q\leq m+n-2t+1$. Alternatively, if $\mathbf{c}_{t,q}$ is from the first partition and $0\leq q\leq n-t$, the dependence of $\mathbf{c}_{t,q}$ and $\mathbf{h}_{t,q}$ on β is removed

$$\begin{array}{llll}\mathbf{c}_{t,q}=\mathbf{c}_{t,q}(\phi)&\mathbf{h}_{t,q}=\mathbf{h}_{t,q}(\phi,\mathbf{z})&\text{if}&0\leq q\leq n-t\\\mathbf{c}_{t,q}=\mathbf{c}_{t,q}(\beta,\phi)&\mathbf{h}_{t,q}=\mathbf{h}_{t,q}(\beta,\phi,\mathbf{z})&\text{if}&n-t+1\leq q\leq m+n-2t+1.\end{array}$$

Equation (3.48) is non-linear and is solved by using the Newton-Raphson method. The

residual associated with an approximate solution of this is given by

$$\begin{aligned}\mathbf{r}\left(\beta,\phi,\mathbf{x}_{t,q},\mathbf{z}\right) &= \mathbf{c}_{t,q}\left(\beta,\phi\right)+\mathbf{h}_{t,q}\left(\beta,\phi,\mathbf{z}\right)\\ &\quad-\left(D_{m+n-t}^{-1}\left(T_t\left(\beta,\phi\right)+F_t\left(\beta,\phi,\mathbf{z}\right)\right)\hat{Q}_t\right)M_{t,q}\mathbf{x}_{t,q}\end{aligned}\tag{3.49}$$

and the computation of the vectors $\mathbf{z}$ and $\mathbf{x}_{t,q}$ and β and ϕ such that $\mathbf{r}(\beta,\phi,\mathbf{x}_{t,q},\mathbf{z})=0$ yields an LSE. The residual $\tilde{\mathbf{r}}$ is defined by

$$\begin{aligned}\tilde{\mathbf{r}} &= \mathbf{r}\left(\beta+\delta\beta,\,\phi+\delta\phi,\,\mathbf{x}_{t,q}+\delta\mathbf{x}_{t,q},\,\mathbf{z}+\delta\mathbf{z}\right)\\ &= \mathbf{c}_{t,q}\left(\beta+\delta\beta,\,\phi+\delta\phi\right)+\mathbf{h}_{t,q}\left(\beta+\delta\beta,\,\phi+\delta\phi,\,\mathbf{z}+\delta\mathbf{z}\right)\\ &\quad-\left(D_{m+n-t}^{-1}\left(T_t\left(\beta+\delta\beta,\,\phi+\delta\phi\right)+F_t\left(\beta+\delta\beta,\,\phi+\delta\phi,\,\mathbf{z}+\delta\mathbf{z}\right)\right)\hat{Q}_t\right)M_{t,q}(\mathbf{x}_{t,q}+\delta\mathbf{x}_{t,q}),\end{aligned}$$

which to first order is given by

$$\begin{aligned}\tilde{\mathbf{r}} = \mathbf{r}\left(\beta,\phi,\mathbf{x}_{t,q},\mathbf{z}\right) &-\left(D_{m+n-t}^{-1}\left(\frac{\partial T_t}{\partial\phi}+\frac{\partial F_t}{\partial\phi}\right)\hat{Q}_tM_{t,q}\mathbf{x}_{t,q}-\left(\frac{\partial\mathbf{c}_{t,q}}{\partial\phi}+\frac{\partial\mathbf{h}_{t,q}}{\partial\phi}\right)\right)\delta\phi\\ &-\left(D_{m+n-t}^{-1}\left(\frac{\partial T_t}{\partial\beta}+\frac{\partial F_t}{\partial\beta}\right)\hat{Q}_tM_{t,q}\mathbf{x}_{t,q}-\left(\frac{\partial\mathbf{c}_{t}}{\partial\beta}+\frac{\partial\mathbf{h}_{t,q}}{\partial\beta}\right)\right)\delta\beta\\ &-\left(D_{m+n-t}^{-1}\left(T_t+F_t\right)\hat{Q}_tM_{t,q}\right)\delta\mathbf{x}_{t,q}+\sum_{i=0}^{m+n+1}\frac{\partial\mathbf{h}_{t,q}}{\partial z_i}\delta z_i\\ &-\left(D_{m+n-t}^{-1}\sum_{i=0}^{m+n+1}\frac{\partial F_t}{\partial z_i}\delta z_i\right)\hat{Q}_tM_{t,q}\mathbf{x}_{t,q}.\end{aligned}\tag{3.50}$$

The terms $\frac{\partial T_t}{\partial\phi}$, $\frac{\partial F_t}{\partial\phi}$, $\frac{\partial\mathbf{c}_{t,q}}{\partial\phi}$, $\frac{\partial\mathbf{h}_{t,q}}{\partial\phi}$, $\frac{\partial T_t}{\partial\beta}$, $\frac{\partial F_t}{\partial\beta}$, $\frac{\partial\mathbf{c}_{t,q}}{\partial\beta}$ and $\frac{\partial\mathbf{h}_{t,q}}{\partial\beta}$ are easily computed.

The final two terms in (3.50) first require some manipulation. Firstly, a matrix-vector product which defines $\mathbf{h}_{t,q}$ is found, and secondly, due to the commutative nature of multiplication, the product $B_t(\tilde{s}(\omega),\beta\tilde{t}(\omega))\mathbf{x}_{t,q}$ is rearranged as a matrix containing the entries of the vector $\mathbf{x}_{t,q}$ and a vector containing coefficients of the polynomials $\bar{s}(x)$ and $\bar{t}(x)$.

Expressing $\mathbf{h}_{t,q}$ as a Matrix-Vector Product

A column $\mathbf{h}_{t,j}$ for $j=0,\ldots,m+n-2t+1$ of the structured matrix $B_t(\beta,\phi,\mathbf{z})$ can be written as a matrix-vector product, where the matrix consists of entries containing ϕ^i and the vectors $\mathbf{s}$ and $\mathbf{t}$ contain the coefficients of the polynomials $\bar{s}(x)$ and $\bar{t}(x)$.

If j is in the interval $[0,n-t]$, then the vector $\mathbf{h}_{t,j}$ is a column in the first partition of the structured matrix $B_t(\beta,\phi,\mathbf{z})$ and is of the form

$$\mathbf{h}_{t,j}=\left[\begin{array}{ccccccc}0_j, & \frac{\bar{z}_{1,0}\binom{m}{0}\binom{n-t}{j}}{\binom{m+n-t}{j}}, & \frac{\bar{z}_{1,1}\phi\binom{m}{1}\binom{n-t}{j+1}}{\binom{m+n-t}{j+1}}, & \cdots & \frac{\bar{z}_{1,m}\phi^m\binom{m}{m}\binom{n-t}{j+1}}{\binom{m+n-t}{m+j+1}}, & 0_{n-t-j}\end{array}\right]^T,$$

which can be written as

$$\mathbf{h}_{t,j}=D_{m+n-t}^{-1}P_t\left[\begin{array}{c}\mathbf{s}\\ \mathbf{t}\end{array}\right].\tag{3.51}$$

The diagonal matrix $D_{m+n-t}^{-1}\in\mathbb{R}^{(m+n-t+1)\times(m+n-t+1)}$ has the same structure as (2.10)

and the matrix P_t is given by

$$P_t = \binom{n-t}{j} \begin{bmatrix} 0_{j,\,m+1} & 0_{j,\,n+1} \\ I_{m+1,\,m+1} & 0_{m+1,\,n+1} \\ 0_{n-t-j,\,m+1} & 0_{n-t-j,\,n+1} \end{bmatrix} \begin{bmatrix} \phi_m & \\ & \phi_n \end{bmatrix} \begin{bmatrix} Q_m & \\ & Q_n \end{bmatrix},$$

where $I_{m+1,m+1} \in \mathbb{R}^{(m+1)\times(m+1)}$ is the $(m+1)$th identity matrix $I_{m+1} \in \mathbb{R}^{(m+1)\times(m+1)}$. The matrices $\phi_m \in \mathbb{R}^{(m+1)\times(m+1)}$ and $\phi_n \in \mathbb{R}^{(n+1)\times(n+1)}$ are diagonal matrices given by

$$\phi_m = \operatorname{diag}\begin{bmatrix} 1, & \phi, & \ldots, & \phi^m \end{bmatrix} \quad \text{and} \quad \phi_n = \operatorname{diag}\begin{bmatrix} 1, & \phi, & \ldots, & \phi^n \end{bmatrix}.$$

The vectors $\bar{\mathbf{s}} \in \mathbb{R}^{(m+1)}$ and $\bar{\mathbf{t}} \in \mathbb{R}^{(n+1)}$ contain the coefficients $\{\bar{z}_{1,i}\}$ and $\{\bar{z}_{2,i}\}$ respectively and are given by

$$\mathbf{s} = \begin{bmatrix} z_{1,0} & z_{1,1} & \ldots & z_{1,m} \end{bmatrix}^T \in \mathbb{R}^{m+1},$$
$$\mathbf{t} = \begin{bmatrix} z_{2,0} & z_{2,1} & \ldots & z_{2,n} \end{bmatrix}^T \in \mathbb{R}^{n+1}.$$

Example 3.5.1. This example considers the expression of a column of the structured matrix $B_t(\beta, \phi, \mathbf{z})$ as a matrix-vector product, where the vector contains the set of coefficients of $\bar{s}(x)$ and $\bar{t}(x)$.

Consider the system where $m = 3$, $n = 5$ and $t = 2$. The third column of the matrix $B_3(\beta, \phi, \mathbf{z})$ is given by

$$\mathbf{h}_{2,2} = \begin{bmatrix} 0 & 0 & \frac{z_{1,0}\binom{3}{0}\binom{3}{2}}{\binom{6}{2}} & \frac{z_{1,1}\phi\binom{3}{1}\binom{3}{2}}{\binom{6}{3}} & \frac{z_{1,2}\phi^2\binom{3}{2}\binom{3}{2}}{\binom{6}{4}} & \frac{z_{1,3}\phi^3\binom{3}{3}\binom{3}{2}}{\binom{6}{5}} & 0 \end{bmatrix}^T$$

and can be written as the matrix-vector product

$$\mathbf{c}_{2,2} = D_6^{-1} P_2 \begin{bmatrix} \mathbf{s} \\ \mathbf{t} \end{bmatrix},$$

where the matrix $D_6^{-1} \in \mathbb{R}^{7\times 7}$

$$D_6^{-1} = \operatorname{diag}\begin{bmatrix} \frac{1}{\binom{6}{0}}, & \frac{1}{\binom{6}{1}}, & \frac{1}{\binom{6}{2}}, & \frac{1}{\binom{6}{3}}, & \frac{1}{\binom{6}{4}}, & \frac{1}{\binom{6}{5}}, & \frac{1}{\binom{6}{6}} \end{bmatrix}.$$

The matrix $P_2 \in \mathbb{R}^{7\times 10}$ is given by

$$P_2 = \binom{3}{2} \times \left[\begin{array}{cccc|cccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline \binom{3}{0} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \binom{3}{1}\phi & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \binom{3}{2}\phi^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \binom{3}{3}\phi^3 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right],$$

and $\mathbf{s} \in \mathbb{R}^4$ and $\mathbf{t} \in \mathbb{R}^6$ are vectors of coefficients given by

$$\mathbf{s} = \begin{bmatrix} z_{1,0}, & z_{1,1}, & z_{1,2}, & z_{1,3} \end{bmatrix}^T \quad \text{and} \quad \mathbf{t} = \begin{bmatrix} z_{2,0}, & z_{2,1}, & z_{2,2}, & z_{2,3}, & z_{2,4}, & z_{2,5} \end{bmatrix}^T.$$

□

Now suppose instead that the $\hat{\jmath}$th column which lies in the second partition of the subresultant matrix is removed. The vector $\mathbf{h}_{t,\hat{\jmath}}$ has the form

$$\mathbf{h}_{t,\hat{\jmath}} = \begin{bmatrix} 0_{\hat{\jmath}}, & \beta\bar{z}_{2,0}\frac{\binom{n}{0}\binom{m-t}{\hat{\jmath}}}{\binom{m+n-t}{\hat{\jmath}}}, & \beta\bar{z}_{2,1}\phi\frac{\binom{n}{1}\binom{m-t}{\hat{\jmath}}}{\binom{m+n-t}{\hat{\jmath}+1}}, & \ldots, & \beta\bar{z}_{2,n}\phi^n\frac{\binom{n}{n}\binom{m-t}{\hat{\jmath}}}{\binom{m+n-t}{n+\hat{\jmath}}}, & 0_{m-t-\hat{\jmath}} \end{bmatrix},$$

and can be written as the matrix-vector product

$$\mathbf{h}_{t,\hat{\jmath}} = \beta D^{-1}_{m+n-t} P_t \begin{bmatrix} \bar{\mathbf{s}} \\ \bar{\mathbf{t}} \end{bmatrix}.$$

The matrix P_t is given by

$$P_t = \binom{m+n-t}{\hat{\jmath}+1} \begin{bmatrix} 0_{\hat{\jmath},\,m+1} & 0_{\hat{\jmath},\,n+1} \\ 0_{n+1,\,m+1} & I_{n+1,\,n+1} \\ 0_{m-t-\hat{\jmath},\,m+1} & 0_{m-t-\hat{\jmath},\,n+1} \end{bmatrix} \begin{bmatrix} \phi_m & \\ & \phi_n \end{bmatrix} \begin{bmatrix} Q_m & \\ & Q_n \end{bmatrix},$$

where $I_{n+1,n+1} \in \mathbb{R}^{(n+1)\times(n+1)}$ is the $(n+1)$th identity matrix. The vector $\mathbf{h}_{t,j} \in \mathbb{R}^{m+n-t+1}$ is given by

$$\mathbf{h}_t = \beta D^{-1}_{m+n-t} P_t \mathbf{z}.$$

The penultimate term in (3.50) can therefore be rearranged as

$$\sum_{i=0}^{m+n+1} \frac{\partial \mathbf{h}_{t,q}}{\partial z_i} \delta z_i = \beta D^{-1}_{m+n-t} P_t \delta \mathbf{z}.$$

The Matrix-Vector Product Rearrangement

The last term in (3.50) is also simplified by noting that the matrix-vector product $D^{-1}_{m+n-t} F_t \hat{Q}_t M_{t,q} \mathbf{x}_{t,q}$ can be written as a matrix-vector product $D^{-1}_{m+n-t} Y_t(x_{t,q})\mathbf{z}$. The equation given by

$$S_t\left(\tilde{f}(\omega), \beta\tilde{g}(\omega)\right) M_{t,q}\mathbf{x}_{t,q} = \mathbf{c}_{t,q}$$

or

$$D^{-1}_{m+n-t} \begin{bmatrix} T_{n-k}\left(\tilde{f}(\omega)\right) \;\middle|\; T_{m-k}\left(\beta\tilde{g}(\omega)\right) \end{bmatrix} \begin{bmatrix} Q_{n-t} & \\ & Q_{m-t} \end{bmatrix} M_{t,q}\mathbf{x}_{t,q} = \mathbf{c}_{t,q} \tag{3.52}$$

can be written as a matrix whose non-zero entries contain $\{x_i \mid i = 0, \ldots, m+n-2t\}$ and a vector in terms of $\bar{f}(x)$ and $\bar{g}(x)$.

The matrix-vector product $M_{t,q}\mathbf{x}_{t,q}$ in (3.52) is denoted $\bar{\mathbf{x}}$ and is given by inserting a zero

into the qth position of $\mathbf{x}_{t,q}$, that is, $\bar{\mathbf{x}} \in \mathbb{R}^{m+n-2t+2}$ is given by

$$\bar{\mathbf{x}} = \begin{bmatrix} \bar{\mathbf{x}}_1, & \bar{\mathbf{x}}_2 \end{bmatrix}^T,$$

where either $\bar{\mathbf{x}}_1$ or $\bar{\mathbf{x}}_2$ contains the introduced zero entry. Assuming that $q < m - t + 1$, the vectors $\bar{\mathbf{x}}_1$ and $\bar{\mathbf{x}}_2$ are given by

$$\begin{aligned} \bar{\mathbf{x}}_1 &= \begin{bmatrix} x_0 & x_1 & \dots & x_{q-1} & 0 & x_q & \dots & x_{m-t-1} \end{bmatrix}^T \in \mathbb{R}^{m-t+1} \\ \bar{\mathbf{x}}_2 &= \begin{bmatrix} x_{m-t-1} & x_{m-t} & \dots & x_{m+n-2t} \end{bmatrix}^T \in \mathbb{R}^{n-t+1}. \end{aligned}$$

For simplicity, vector $\bar{\mathbf{x}}$ is now redefined as

$$\bar{\mathbf{x}} = \begin{bmatrix} \bar{\mathbf{x}}_1 \\ \bar{\mathbf{x}}_2 \end{bmatrix} = \begin{bmatrix} \tilde{x}_0 & \tilde{x}_1, & \dots, & \tilde{x}_{m-t}, \end{bmatrix} \begin{bmatrix} \tilde{x}_{m-t+1}, & \tilde{x}_{m-t+2}, & \dots, & \tilde{x}_{m+n-2t+1} \end{bmatrix}^T \in \mathbb{R}^{m+n-2t+2},$$

where the entries $\{\tilde{x}_0, \dots, \tilde{x}_{m+n-2t+1}\}$ are the set of entries $\{x_0, \dots, x_{m+n-2t}\}$ with a zero inserted.

The matrix-vector product in (3.52) can be written as the product of a matrix which has entries containing coefficients $\{\tilde{x}_i\}$ and a vector containing coefficients of the polynomials $\bar{f}(x)$ and $\bar{g}(x)$

$$D_{m+n-t}^{-1} \begin{bmatrix} T_m(\bar{x}_1) \end{bmatrix} \begin{bmatrix} T_n(\beta, \bar{x}_2) \end{bmatrix} \begin{bmatrix} \phi_m & \\ & \phi_n \end{bmatrix} \begin{bmatrix} Q_m & \\ & Q_n \end{bmatrix} \begin{bmatrix} \bar{\mathbf{f}} \\ \bar{\mathbf{g}} \end{bmatrix}.$$

The matrix $T_m(\bar{x}_1) \in \mathbb{R}^{(t+1)\times(m+1)}$ is given by

$$T_m(\bar{x}_1) = \begin{bmatrix} \tilde{x}_0\binom{m-t}{0} & & & \\ \tilde{x}_1\binom{m-t}{1} & \tilde{x}_0\binom{m-t}{0} & & \\ \vdots & \tilde{x}_1\binom{m-t}{1} & \ddots & \\ \tilde{x}_{m-t}\binom{m-t}{m-t} & \vdots & \ddots & \tilde{x}_0\binom{m-t}{0} \\ & \tilde{x}_{m-t}\binom{m-t}{m-t} & & \tilde{x}_1\binom{m-t}{1} \\ & & \ddots & \vdots \\ & & & \tilde{x}_{m-t}\binom{m-t}{m-t} \end{bmatrix}$$

and $T_n(\beta, \bar{x}_2) \in \mathbb{R}^{(t+1)\times(n+1)}$ is given by

$$T_n(\beta, \bar{x}_2) = \begin{bmatrix} \beta\tilde{x}_{m-t+1}\binom{n-t}{0} & & & \\ \beta\tilde{x}_{m-t+2}\binom{n-t}{1} & \beta\tilde{x}_{m-t+1}\binom{n-t}{0} & & \\ \vdots & \beta\tilde{x}_{m-t+2}\binom{n-t}{1} & \ddots & \\ \beta\tilde{x}_{m+n-2t+1}\binom{n-t}{n-t} & \vdots & \ddots & \beta\tilde{x}_{m-t+1}\binom{n-t}{0} \\ & \beta\tilde{x}_{m+n-2t+1}\binom{n-t}{n-t} & & \beta\tilde{x}_{m-t+1}\binom{n-t}{1} \\ & & \ddots & \vdots \\ & & & \beta\tilde{x}_{m+n-2t+1}\binom{n-t}{n-t} \end{bmatrix}.$$

Given the rearrangement above, the expression

$$D_{m+n-t}^{-1} F_t \left(\tilde{s}(\omega), \beta \tilde{t}(\omega) \right) \hat{Q}_t M_{t,q} \mathbf{x}_{t,q} \tag{3.53}$$

is similarly rearranged, so that the final term in (3.50) can be written in an alternative form

$$D_{m+n-t}^{-1} \left[\begin{array}{c|c} T_m(\bar{x}_1) & T_n(\beta \bar{x}_2) \end{array} \right] \left[\begin{array}{cc} \phi_m & \\ & \phi_n \end{array} \right] \left[\begin{array}{cc} Q_m & \\ & Q_n \end{array} \right] \left[\begin{array}{c} \mathbf{s} \\ \mathbf{t} \end{array} \right] = \mathbf{h}_{t,q}, \tag{3.54}$$

and the matrix $Y_t(\bar{x}_1, \bar{x}_2)$ is given by

$$Y_t(\bar{x}_1, \bar{x}_2) = \left[\begin{array}{c|c} T_m(\bar{x}_1) & T_n(\bar{x}_2) \end{array} \right] \left[\begin{array}{cc} \phi_m & \\ & \phi_n \end{array} \right] \left[\begin{array}{cc} Q_m & \\ & Q_n \end{array} \right].$$

The differentiation of (3.54) with respect to $\mathbf{z}$ yields

$$D_{m+n-t}^{-1} Y_t(\beta, \phi, \mathbf{x}_{t,q}) \, \delta \mathbf{z} = \sum_{i=0}^{m+n+1} \left(\frac{\partial F_t}{\partial z_i} \delta z_i \right) \hat{Q}_t M_{t,q} \mathbf{x}_{t,q}$$

and thus (3.50) simplifies to

$$\begin{aligned} \tilde{\mathbf{r}} = \mathbf{r}(\beta, \phi, \mathbf{x}_{t,q}, \mathbf{z}) &- \left(D_{m+n-t}^{-1} \left(\frac{\partial T_t}{\partial \phi} + \frac{\partial F_t}{\partial \phi} \right) \hat{Q}_t M_q \mathbf{x}_{t,q} - \left(\frac{\partial \mathbf{c}_{t,q}}{\partial \phi} + \frac{\partial \mathbf{h}_{t,q}}{\partial \phi} \right) \right) \delta \phi \\ &- \left(D_{m+n-t}^{-1} \left(\frac{\partial T_t}{\partial \beta} + \frac{\partial F_t}{\partial \beta} \right) \hat{Q}_t M_{t,q} \mathbf{x}_{t,q} - \left(\frac{\partial \mathbf{c}_{t,q}}{\partial \beta} + \frac{\partial \mathbf{h}_{t,q}}{\partial \beta} \right) \right) \delta \beta \\ &- \left(D_{m+n-t}^{-1} (T_t + F_t) \hat{Q}_t M_{t,q} \right) \delta \mathbf{x}_{t,q} - D_{m+n-t}^{-1} (Y_t - P_t) \, \delta \mathbf{z} \end{aligned}$$

to first order. The jth iteration in the Newton-Raphson method for the calculation of β, ϕ, $\mathbf{x}_{t,q}$ and $\mathbf{z}$ is obtained from

$$\left[\begin{array}{cccc} H_{\mathbf{z}} & H_{\mathbf{x}_{t,q}} & H_\beta & H_\phi \end{array} \right]^{(j)} \left[\begin{array}{c} \delta \mathbf{z} \\ \delta \mathbf{x}_{t,q} \\ \delta \beta \\ \delta \phi \end{array} \right]^{(j)} = \mathbf{r}^{(j)}, \tag{3.55}$$

where $\mathbf{r}^{(j)} = \mathbf{r}^{(j)}(\beta, \phi, \mathbf{x}_{t,q}, \mathbf{z})$

$$\begin{aligned} H_{\mathbf{z}} &= D_{m+n-t}^{-1} (Y_t - P_t) \in \mathbb{R}^{(m+n-t+1) \times (m+n+2)}, \\ H_{\mathbf{x}_{t,q}} &= D_{m+n-t}^{-1} (T_t + F_t) \hat{Q}_t M_{t,q} \in \mathbb{R}^{(m+n-t+1) \times (m+n-2t+1)}, \\ H_\beta &= D_{m+n-t}^{-1} \left(\frac{\partial T_t}{\partial \beta} + \frac{\partial F_t}{\partial \beta} \right) \hat{Q}_t M_{t,q} \mathbf{x}_{t,q} - \left(\frac{\partial \mathbf{c}_{t,q}}{\partial \beta} + \frac{\partial \mathbf{h}_{t,q}}{\partial \beta} \right) \in \mathbb{R}^{m+n-t+1}, \\ H_\phi &= D_{m+n-t}^{-1} \left(\frac{\partial T_t}{\partial \phi} + \frac{\partial F_t}{\partial \phi} \right) \hat{Q}_t M_{t,q} \mathbf{x}_{t,q} - \left(\frac{\partial \mathbf{c}_{t,q}}{\partial \phi} + \frac{\partial \mathbf{h}_{t,q}}{\partial \phi} \right) \in \mathbb{R}^{m+n-t+1}. \end{aligned}$$

The values β, ϕ, $\mathbf{x}_{t,q}$ and $\mathbf{z}$ at the jth iteration are given by

$$\mathbf{y}^{(j)} = \mathbf{y}^{(j-1)} + \delta\mathbf{y}^{(j)}, \quad \begin{bmatrix} \mathbf{z}, \\ \mathbf{x}_{t,q}, \\ \beta, \\ \phi \end{bmatrix}^{(j)}, \text{ and } \quad \delta\mathbf{y}^{j} = \begin{bmatrix} \delta\mathbf{z}, \\ \delta\mathbf{x}_{t,q}, \\ \delta\beta, \\ \delta\phi \end{bmatrix}^{(j)}. \tag{3.56}$$

The initial value of $\mathbf{z}$ is the zero vector $\mathbf{z}^{(0)} = \mathbf{0}$, since $\mathbf{z}$ is the vector of the perturbations of the coefficients of the inexact polynomials $\tilde{f}(\omega)$ and $\tilde{g}(\omega)$, and the initial perturbations are zero. The initial values of ϕ and β are given by $\phi^{(0)} = \theta_t$ and $\beta^{(0)} = \alpha_t$. Finally, the initial value of $\mathbf{x}_{t,q}^{(0)}$ is calculated from (3.49), where $\mathbf{r} = \mathbf{h}_{t,q} = 0$ and $F_t = 0$

$$\mathbf{x}_{t,q}^{(0)} = \arg\min_{\mathbf{w}} \| \left(D_{m+n-t}^{-1} T_t\left(\alpha_0, \theta_0\right) \hat{Q}_t M_{t,q} \right) \mathbf{w} - \mathbf{c}_{t,q}(\alpha_0, \theta_0) \|.$$

Equation (3.55) is under-determined and has infinitely many solutions. It can therefore be written as

$$C^{(j)} \delta\mathbf{y}^{(j)} = \mathbf{r}^{(j)},$$

where $C^{(j)}$ is given by

$$C^{(j)} = \begin{bmatrix} H_{\mathbf{z}}, & H_{\mathbf{x}_{t,q}}, & H_{\beta}, & H_{\phi} \end{bmatrix}^{(j)}.$$

It is necessary to seek the solution such that the values $\delta\mathbf{z}$, $\delta\beta$ and $\delta\phi$ are all minimised, that is, where the vector $\delta\mathbf{y}^{(j)}$ is minimised. The magnitude of difference between $\mathbf{y}^{(j)}$ and the initial estimate $\mathbf{y}^{(0)}$ is given by

$$\left\| \begin{bmatrix} (\mathbf{z}^{(j)} - \mathbf{z}^{(0)}) \\ (\mathbf{x}_{t,q}^{(j)} - \mathbf{x}_{t,q}^{(0)}) \\ (\alpha^{(j)} - \alpha^{(0)}) \\ (\theta^{(j)} - \theta^{(0)}) \end{bmatrix}^{(j)} \right\| = \left\| \begin{bmatrix} \delta\mathbf{z}^{(j)} \\ \delta\mathbf{x}_{t,q}^{(j)} \\ \delta\alpha^{(j)} \\ \theta^{(j)} \end{bmatrix} - \left(\begin{bmatrix} \mathbf{z}^{(0)} \\ \mathbf{x}_{t,q}^{(0)} \\ \alpha^{(0)} \\ \theta^{(0)} \end{bmatrix} - \begin{bmatrix} \mathbf{z}^{(j-1)} \\ \mathbf{x}_{t,q}^{(j-1)} \\ \alpha^{(j-1)} \\ \theta^{(j-1)} \end{bmatrix} \right) \right\|$$

which is of the form

$$\left\| \delta\mathbf{y} - (\mathbf{y}^{(0)} - \mathbf{y}^{(j-1)}) \right\| = \left\| \delta\mathbf{y}^{(j)} - \mathbf{p}^{(j)} \right\|, \tag{3.57}$$

and the minimisation problem is given by

Minimise

$$\left\| \delta\mathbf{y}^{(j)} - \mathbf{p}^{(j)} \right\| \tag{3.58}$$

subject to

$$\begin{bmatrix} H_{\mathbf{z}}, & H_{\mathbf{x}_{t,q}}, & H_{\beta}, & H_{\phi} \end{bmatrix}^{(j)} \delta\mathbf{y}^{(j)} = \mathbf{r}^{(j)}.$$

This problem can be solved by QR decomposition at each iteration [36], where $\delta\mathbf{y}^{(j)}$, $C^{(j)}$, $\mathbf{p}^{(j)}$ and $\mathbf{r}^{(j)}$ are updated between iterations.

If the iterative procedure converges and the residual is minimised, then the vector of perturbations $z^*_{1,i}$ for $i = 0, \ldots, m$ and $z^*_{2,i}$ for $i = 0, \ldots, n$, and parameters β^* and ϕ^* can be recovered from the vector $\mathbf{y}^{(j)}$. The updated polynomials are therefore given by

$$\dot{f}(\omega) = \sum_{i=0}^{m} \dot{a}_i \left(\phi^*\right)^i \binom{m}{i} (1 - \phi^*\omega)^{m-i}\omega^i \quad \text{where} \quad \tilde{a}_i = \left(\bar{a}_i + z^*_{1,i}\right) \tag{3.59}$$

$$\beta^*\dot{g}(\omega) = \beta \sum_{i=0}^{n} \dot{b}_i \left(\phi^*\right)^i \binom{n}{i} (1 - \phi^*\omega)^{n-i}\omega^i \quad \text{where} \quad \dot{b}_i = \bar{b}_i + z^*_{2,i}, \tag{3.60}$$

which have a GCD $\dot{d}(\omega)$ given by

$$\dot{d}(\omega) = \sum_{i=0}^{t} \tilde{d}_i \left(\phi^*\right)^i \binom{t}{i} (1 - \phi^*\omega)^{t-i}\, \omega^i. \tag{3.61}$$

These updated polynomials satisfy the equations

$$\dot{u}_t(\omega)\dot{d}_t(\omega) = \dot{f}(\omega) \qquad \text{and} \qquad \dot{v}_t(\omega)\dot{d}_t(\omega) = \beta\dot{g}(\omega),$$

where $\dot{u}_t(\omega)$ and $\dot{v}_t(\omega)$ are obtained by the least squares solution $\mathbf{x}_{t,q}$ of (3.48) with $\beta = \beta^*$ and $\mathbf{z} = \mathbf{z}^*$. Having obtained $\dot{u}_t(\omega)$ and $\dot{v}_t(\omega)$, the coefficients of the polynomial $\dot{d}_t(\omega)$ are obtained from the matrix equation

$$\begin{bmatrix} D_m^{-1} & \\ & D_n^{-1} \end{bmatrix} \begin{bmatrix} T_t\left(\dot{u}_t(\omega)\right) \\ T_t\left(\dot{v}_t(\omega)\right) \end{bmatrix} Q_t\dot{\mathbf{d}}_t = \begin{bmatrix} \dot{\mathbf{f}} \\ \beta^*\dot{\mathbf{g}} \end{bmatrix},$$

where the matrices $D_m^{-1} \in \mathbb{R}^{(m+1)\times(m+1)}$ and $D_n^{-1} \in \mathbb{R}^{(n+1)\times(n+1)}$ are matrices of binomial coefficients with the same structure as (2.10) and are given by

$$D_m^{-1} = \operatorname{diag}\left[\ \tfrac{1}{\binom{m}{0}},\ \ \tfrac{1}{\binom{m}{1}},\ \ \ldots,\ \ \tfrac{1}{\binom{m}{m}}\ \right]$$
$$D_n^{-1} = \operatorname{diag}\left[\ \tfrac{1}{\binom{n}{n}},\ \ \tfrac{1}{\binom{n}{n}},\ \ \ldots,\ \ \tfrac{1}{\binom{n}{n}}\ \right].$$

The matrices $T_t\left(\dot{u}_t(\omega)\right) \in \mathbb{R}^{(m+1)\times(t+1)}$ and $T_t\left(\dot{v}_t(\omega)\right) \in \mathbb{R}^{(n+1)\times(n+1)}$ are tth order univariate Toeplitz matrices of the polynomials $\dot{u}_t(\omega)$ and $\dot{v}_t(\omega)$. These matrices have the same structure as the Toeplitz matrix in (2.11). The diagonal matrix $Q_t \in \mathbb{R}^{(t+1)\times(t+1)}$ has the same structure as Q_n in (2.12) and consists of binomial coefficients corresponding to the polynomial $\dot{d}(\omega)$

$$Q_t = \operatorname{diag}\left[\ \tbinom{t}{0},\ \ \tbinom{t}{1},\ \ \ldots,\ \ \tbinom{t}{t}\ \right].$$

The vector $\dot{\mathbf{d}}$ is a vector of the coefficients of the polynomial $\dot{d}(\omega)$ and the vectors $\dot{\mathbf{f}}$ and $\dot{\mathbf{g}}$ are vectors of the coefficients of the corrected polynomials $\dot{f}(\omega)$ and $\dot{g}(\omega)$

$$\dot{\mathbf{f}} = \left[\ \dot{a}_0,\ \ \dot{a}_1\phi^*,\ \ \ldots,\ \ \dot{a}_m(\phi^*)^m\ \right]^T \in \mathbb{R}^{m+1},$$
$$\dot{\mathbf{g}} = \left[\ \dot{b}_0,\ \ \dot{b}_1\phi^*,\ \ \ldots,\ \ \dot{b}_n(\phi^*)^n\ \right]^T \in \mathbb{R}^{n+1},$$
$$\dot{\mathbf{d}} = \left[\ \dot{d}_0,\ \ \dot{d}_1\phi^*,\ \ \ldots,\ \ \dot{d}_t(\phi^*)^t\ \right]^T \in \mathbb{R}^{t+1},$$

where the coefficients $\dot{a}_i\,(\phi^*)^i$, $\dot{b}_i\,(\phi^*)^i$ and $\dot{d}_i\,(\phi^*)^i$ are defined in (3.59), (3.60) and (3.61).

A set of examples now shows that the determination of the low rank approximation of the subresultant matrix, by addition of minimal perturbations, yields improved results in the approximation of coefficients of cofactor polynomials $\hat{u}_t(x)$ and $\hat{v}_t(x)$.

Example 3.5.2. Consider the Bernstein form of the exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$, whose factorisations are given by

$$\hat{f}(x) = (x-1.46)^2(x-1.37)^3(x-0.82)^3(x-0.75)^3(x-0.56)^8(x-0.1)^3(x+0.27)^3$$
$$\hat{g}(x) = (x-2.12)(x-1.37)^3(x-1.2)^3(x-0.99)^4(x-0.75)^3(x-0.56)^8(x-0.1)^2,$$

and whose GCD is given by

$$\hat{d}(x) = (x-1.37)^3(x-0.75)^3(x-0.56)^8(x-0.1)^2.$$

Noise is added to the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$, and coefficients of the inexact polynomials $f(x)$ and $g(x)$ are given by

$$a_i = \hat{a}_i + r_{f,i}\hat{a}_i\epsilon_{f,i} \qquad \text{and} \qquad b_j = \hat{b}_j + r_{g,j}\hat{b}_j\epsilon_{g,j}, \tag{3.62}$$

where $\{r_{f,i}\}$ and $\{r_{g,j}\}$ are uniformly distributed random variables in the interval $[-1,1]$, and $\{\epsilon_{f,i}\}$ and $\{\epsilon_{g,j}\}$ are uniformly distributed random variables in the interval $\left[10^{-10}, 10^{-6}\right]$.

Figure 3.20 shows the normalised singular values of the Sylvester matrix of four pairs of polynomials (i) $f(x)$ and $g(x)$, (ii) $\tilde{f}(\omega)$ and $\alpha\tilde{g}(\omega)$, (iii) $\dot{f}(\omega)$ and $\alpha\dot{g}(\omega)$, and (iv) $\dot{f}(x)$ and $\dot{g}(x)$. The numerical rank of the Sylvester matrix of unprocessed polynomials $S(f(x), g(x))$ cannot be determined from its singular values as there is no distinct change between any singular value σ_i and the proceeding σ_{i+1}. That is, there is no $\delta\dot{\sigma}_k$ which is significantly larger than all other $\{\delta\sigma_i\}$. However, the numerical rank of the Sylvester matrix of preprocessed polynomials $\tilde{f}_t(\omega)$ and $\alpha_t\tilde{g}_t(\omega)$ is clearly defined and is equal to 33. The separation between non-zero and numerically zero singular values is more clearly defined for perturbed polynomials $S(\dot{f}(\omega), \beta\dot{g}(\omega))$, and similarly for $S(\dot{f}(x), \dot{g}(x))$.

□

The following examples consider three methods for the approximations of the GCD triple $\hat{u}(x)$, $\hat{v}(x)$ and $\hat{d}(x)$ given the inexact polynomials $f(x)$ and $g(x)$, whose coefficients are given by

$$a_i = \hat{a}_i + r_{f,i}\hat{a}_i\epsilon_{f,i} \qquad \text{and} \qquad b_j = \hat{b}_j + r_{g,j}\hat{b}_j\epsilon_{g,j}, \tag{3.63}$$

where $\{\epsilon_{f,i}\}$ and $\{\epsilon_{g,j}\}$ are uniformly distributed random variables in the interval $[\epsilon_L, \epsilon_U]$, and $\{r_{f,i}\}$ and $\{r_{g,j}\}$ are uniformly distributed random variables in the interval $[-1,1]$.

The following three methods call all be used to compute the approximations of $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{d}_t(x)$:

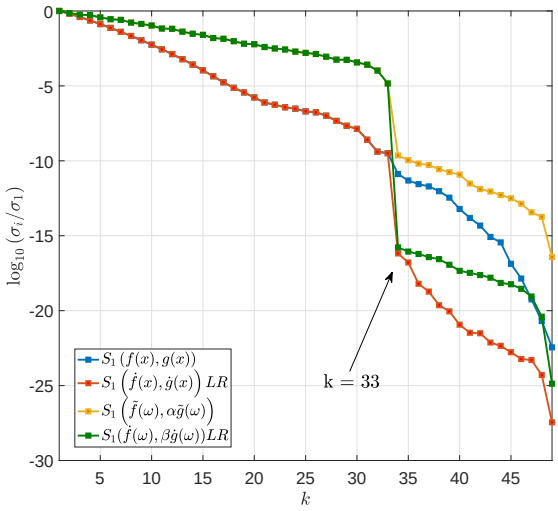


Figure 3.20: The normalised singular values $\{\sigma_i/\sigma_1\}$ of the Sylvester matrices $S(f(x),g(x))$ (before preprocessing) (■) , $S(\dot{f}(x),\dot{g}(x))$ (with structured perturbations) (■) , $S(\tilde{f}(x),\tilde{g}(x))$ (with preprocessing) (■) and $S(\dot{f}(\omega),\beta\dot{g}(\omega))$ (with preprocessing and structured perturbations) (■) in Example 3.5.2

1. The least squares solution $\mathbf{x}_{t,q}$ of $A_{t,q}(f(x),g(x))\mathbf{x}_{t,q} \approx \mathbf{c}_{t,q}$ contains the coefficients of the approximations $u_t(x)$ and $v_t(x)$, from which, the coefficients of the approximation of $d_t(x)$ are easily determined. The distances between these polynomials and their exact counterparts $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{d}_t(x)$ are given by

$$\epsilon\left(u_t(x)\right) = \frac{\|\hat{\mathbf{u}}_t - \mathbf{u}_t\|}{\|\hat{\mathbf{u}}_t\|}, \quad \epsilon\left(v_t(x)\right) = \frac{\|\hat{\mathbf{v}}_t - \mathbf{v}_t\|}{\|\hat{\mathbf{v}}_t\|} \quad \text{and} \quad \epsilon\left(d_t(x)\right) = \frac{\left\|\hat{\mathbf{d}}_t - \mathbf{d}_t\right\|}{\left\|\hat{\mathbf{d}}_t\right\|}. \tag{3.64}$$

2. The least squares solution $\mathbf{x}_{t,q}$ of $A_{t,q}(\tilde{f}_t(\omega),\alpha_t\tilde{g}_t(\omega))\mathbf{x}_{t,q} \approx \mathbf{c}_{t,q}$ contains the coefficients of $\tilde{u}_t(\omega)$ and $\tilde{v}_t(\omega)$, and by a final simple least squares problem, the coefficients of the approximation $\tilde{d}_t(\omega)$ are computed.

   By the substitution $\omega = x/\theta_t$, polynomials $\tilde{u}_t(x)$, $\tilde{v}_t(x)$ and $\tilde{d}_t(x)$ are obtained and the distances between these and the exact polynomials $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{d}_t(x)$ are given by

$$\tilde{\epsilon}\left(u_t(x)\right) = \frac{\|\hat{\mathbf{u}}_t - \tilde{\mathbf{u}}_t\|}{\|\hat{\mathbf{u}}_t\|}, \quad \tilde{\epsilon}\left(v_t(x)\right) = \frac{\|\hat{\mathbf{v}}_t - \tilde{\mathbf{v}}_t\|}{\|\hat{\mathbf{v}}_t\|} \quad \text{and} \quad \dot{\epsilon}\left(d_t(x)\right) = \frac{\left\|\hat{\mathbf{d}}_t - \tilde{\mathbf{d}}_t\right\|}{\left\|\hat{\mathbf{d}}_t\right\|}. \tag{3.65}$$

3. The solution $\mathbf{x}_{t,q}$ of $A_{t,q}(\tilde{f}_t(\omega) + \delta\tilde{f}_t(\omega), (\alpha_t + \delta\alpha_t)\left(\tilde{g}_t(\omega) + \delta\tilde{g}_t(\omega)\right))\mathbf{x}_{t,q} = \mathbf{c}_{t,q} + \mathbf{h}_{t,q}$ contains the coefficients of $\dot{u}_t(\omega)$ and $\dot{v}_t(\omega)$, from which $\dot{d}_t(\omega)$ can be computed. By the substitution $\omega = x/\theta$, the polynomials $\dot{u}_t(x)$, $\dot{v}_t(x)$ and $\dot{d}_t(x)$ are obtained, and

the distances between these and the exact polynomials are given by

$$\dot{\epsilon}\left(u_t(x)\right) = \frac{\|\hat{\mathbf{u}}_t - \dot{\mathbf{u}}_t\|}{\|\hat{\mathbf{u}}_t\|}, \quad \dot{\epsilon}\left(v_t(x)\right) = \frac{\|\hat{\mathbf{v}}_t - \dot{\mathbf{v}}_t\|}{\|\hat{\mathbf{v}}_t\|} \quad \text{and} \quad \dot{\epsilon}\left(d_t(x)\right) = \frac{\left\|\hat{\mathbf{d}}_t - \dot{\mathbf{d}}_t\right\|}{\left\|\hat{\mathbf{d}}_t\right\|}. \tag{3.66}$$

The condition number of each of the matrices (i) $A_{t,q}(f(x), g(x))$, (ii) $A_{t,q}(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$ and (iii) $A_{t,q}(\dot{f}_t(\omega), \beta_t\dot{g}_t(\omega))$ are also recorded.

Example 3.5.3. Consider the Bernstein form of the exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$, whose factorisations are given by

$$\begin{aligned}\hat{f}(x) &= (x-3.4)^3(x-2.5)^3(x-0.8)^2(x-0.7)^3(x-0.5)^2(x-0.3)^2(x-0.1)^4\\ \hat{g}(x) &= (x-1.1)^3(x-0.9)^4(x-0.85)^4(x-0.8)^2(x-0.1)^3,\end{aligned}$$

and whose GCD of degree $t = 5$ is given by

$$\hat{d}(x) = (x-0.8)^2(x-0.1)^3.$$

Noise is added to the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$, and the inexact polynomials $f(x)$ and $g(x)$ have the coefficients $\{a_i\}$ and $\{b_j\}$ given by (3.62), where $\{\epsilon_{f,i}\}$ and $\{\epsilon_{g,j}\}$ are uniformly distributed random variables in the interval $\left[10^{-12}, 10^{-10}\right]$ and $\{r_{f,i}\}$ and $\{r_{g,j}\}$ are uniformly distributed random variables in the interval $[-1, 1]$.

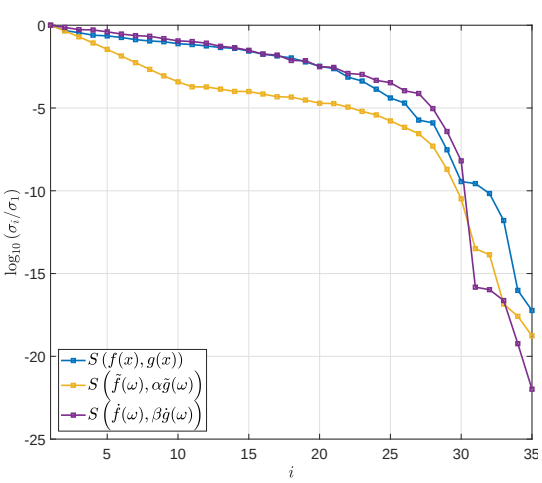
Figure 3.21i shows the singular values of three Sylvester matrices (i) $S_1(f(x), g(x))$, (ii) $S_1(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$ and (iii) $S_1(\dot{f}_t(\omega), \beta_t\dot{g}_t(\omega))$. The computation of the degree of the GCD of two polynomials can be reduced to the determination of the number of numerically zero singular values, and a significant change in the ordered singular values is indicative of moving from the non-zero set to the numerically zero set.

There exists a significant change in the size of the singular values of the unprocessed subresultant matrix $S_1(f(x), g(x))$, where $\{\, \sigma_{1,i} \mid i = 1, \ldots, 33 \,\}$ are large and the remaining values σ_{34} and σ_{35} are numerically zero. The numerical rank is incorrectly determined as 33, hence the degree of the GCD is incorrectly given as two.

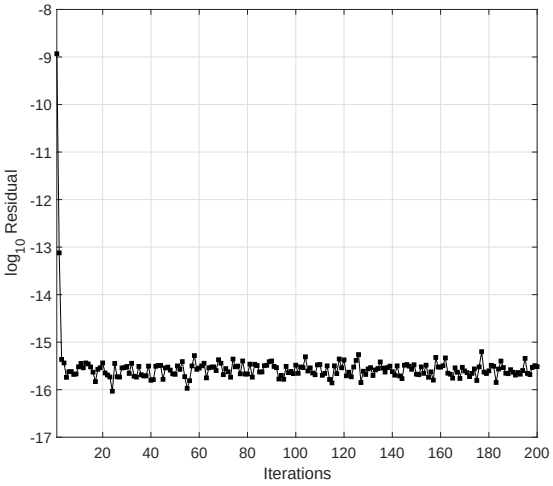
By the same metric, the numerical rank of the Sylvester matrix of the preprocessed polynomials $\tilde{f}_t(\omega)$ and $\alpha_t\tilde{g}_t(\omega)$, given by $S_1(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$, is equal to 30, and the degree of the GCD is correctly given as $t = 5$.

Similar results are obtained by analysis of the singular values of the Sylvester matrix of the perturbed polynomials $\dot{f}_t(\omega)$ and $\beta_t\dot{g}_t(\omega)$, but here the separation of the numerically zero and non-zero singular values is several orders of magnitude larger than the the separation of singular values of $S_1(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$.

In Table 3.2 the error between the exact GCD triple ($\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{d}_t(x)$) and the polynomials obtained by three methods of approximation are given. Also included in this table is the condition number of the matrix denoted A_t, from which the approximations were computed. Note that the first column of Table 3.2 is left blank, since the degree of the GCD was not correctly computed and the approximations of $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{d}_t(x)$ were not computed.



(i) The normalised singular values $\{\sigma_i/\sigma_1\}$ of $S_1(f(x), g(x))$ (■) , $S_1(\tilde{f}_t(\omega), \alpha_t\tilde{g}_t(\omega))$ (■) and $S_1(\dot{f}_t(\omega), \beta_t\dot{g}_t(\omega))$ (■) in Example 3.5.3



(ii) Residual at each iteration of the LSE problem

Figure 3.21: Low rank approximation of the Sylvester matrix in Example 3.5.3

The computation of the low rank approximation of the tth subresultant matrix yields the improved approximations $\dot{u}_t(x)$, $\dot{v}_t(x)$ and $\dot{d}_t(x)$ of the GCD triple $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{d}_t(x)$, which appear to be one order of magnitude better than the approximations obtained by preprocessing alone.

	Unprocessed $u_t(x)$, $v_t(x)$, $w_t(x)$	Preprocessed $\tilde{u}_t(x)$, $\tilde{v}_t(x)$, $\tilde{w}_t(x)$	Perturbed $\dot{u}_t(x)$, $\dot{v}_t(x)$, $\dot{w}_t(x)$
$\kappa(A_t)$	-	$1.1183e+11$	$1.5510e+09$
$\epsilon(u_t(x))$	-	$6.605633e-07$	$5.524534e-08$
$\epsilon(v_t(x))$	-	$1.092723e-07$	$8.380987e-09$
$\epsilon(d_t(x))$	-	$1.225898e-06$	$9.373335e-08$
Average ϵ	-	$6.6524e-07$	$5.2453e-08$

Table 3.2: Error in the approximations of $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{d}_t(x)$ in Example 3.5.3

□

Example 3.5.4. Consider the Bernstein form of the exact polynomials $\hat{f}(x, y)$ and $\hat{g}(x, y)$, whose factorisations are given by

$$\begin{aligned}\hat{f}(x) &= (x-9.7515678)(x-5.56)^2(x-1.46)(x-1.37)(x-1.2435487954)\times\\ &\quad (x-0.82)(x-0.10122344)(x+2.27564657)^2\\ \hat{g}(x) &= (x-9.7515678)(x-5.56)^2(x-2.12)(x-1.37)(x-1.2435487954)\times\\ &\quad (x-1.2222222)(x-0.99102445)\end{aligned}$$

and whose GCD $\hat{d}_t(x)$ of degree $t=5$ is given by

$$\hat{d}(x) = (x-9.7515678)(x-5.56)^2(x-1.37)(x-1.2435487954).$$

Noise is added to coefficients of $\hat{f}(x)$ and $\hat{g}(x)$ as described in (3.63), where $\epsilon_L = 1e-6$, $\epsilon_U = 1e-5$. The three sets of approximations of the GCD triple given by $(u_t(x), v_t(x), d_t(x))$, $(\tilde{u}_t(x), \tilde{v}_t(x), \tilde{d}_t(x))$ and $(\dot{u}_t(x), \dot{v}_t(x), \dot{d}_t(x))$ are computed and their respective errors are given in Table 3.3.

	Unprocessed $u_t(x)$, $v_t(x)$, $d_t(x)$	Preprocessed $\tilde{u}_t(x)$, $\tilde{v}_t(x)$, $\tilde{d}_t(x)$	Perturbed $\dot{u}_t(x)$, $\dot{v}_t(x)$, $\dot{d}_t(x)$
$\kappa(A_t)$	$1.9970e+04$	$3.6460e+03$	$2.4476e+03$
$\epsilon(u_t(x))$	$5.952614e-04$	$1.731751e-04$	$7.321304e-05$
$\epsilon(v_t(x))$	$1.957345e-04$	$7.402057e-05$	$2.468431e-05$
$\epsilon(d_t(x))$	$1.863888e-04$	$6.582286e-05$	$2.311860e-05$
Average ϵ	$3.2579e-04$	$1.0434e-04$	$4.0339e-05$

Table 3.3: Error in the approximations of $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{d}_t(x)$ in Example 3.5.4

Again, the approximations obtained from the structured low rank approximation of the tth subresultant matrix are an order of magnitude better than the approximations obtained by least squares of the preprocessed tth subresultant matrix. □

Example 3.5.5. Consider the Bernstein form of the exact polynomials $\hat{f}(x, y)$ and $\hat{g}(x, y)$, whose factorisations are given by

$$\begin{aligned}\hat{f}(x) &= (x-1.600548798)(x-1.2435487954)(x-0.76549843)^4(x-0.7165792)\times\\ &\quad (x-0.5465444984)(x-0.37987984)(x+2.27564657)^2(x+5.103579)^2\\ \hat{g}(x) &= (x-1.2435487954)(x-1.229876852)(x-0.9916546598)\\ &\quad (x-0.76549843)^4(x-0.54987)(x-0.37987984)(x+5.103579)^2,\end{aligned}$$

and whose GCD of degree $t = 8$ is given by

$$\hat{d}(x) = (x-1.2435487954)(x-0.76549843)^4(x-0.37987984)(x+5.103579)^2.$$

Noise is added to the coefficients of $\hat{f}(x)$ and $\hat{g}(x)$ as in (3.63) and the sets of values $\{\epsilon_{f,i}\}$ and $\{\epsilon_{g,j}\}$ are uniformly distributed random variables in the interval $[10^{-10}, 10^{-8}]$. Polynomials $(u_t(x), v_t(x), d_t(x))$, $(\tilde{u}_t(x), \tilde{v}_t(x), \tilde{d}_t(x))$ and $(\dot{u}_t(x), \dot{v}_t(x), \dot{d}_t(x))$ are computed by the methods described above and errors in these approximations are given in Table 3.4. In this example the approximations are improved by two orders of magnitude by considering the low rank approximation of the tth subresultant matrix.

□

This section has considered the computation of a low rank approximation of the tth subresultant matrix to obtain improved approximations for the coefficients of the cofactor polynomials $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{d}_t(x)$. The original problem of computing the AGCD of two inexact polynomials $f(x)$ and $g(x)$ was reduced to the computation of the rank of the subresultant matrices $S_k(\tilde{f}_k(\omega), \alpha_k\tilde{g}_k(\omega))$. By the computation of a low rank approximation of the tth subresultant matrix, the GCD of the corrected polynomials $\dot{f}_t(\omega)$ and $\beta_t\dot{g}_t(\omega)$ was

	Unprocessed $u_t(x)$, $v_t(x)$, $d_t(x)$	Preprocessed $\tilde{u}_t(x)$, $\tilde{v}_t(x)$, $\tilde{d}_t(x)$	With Perturbations $\dot{u}_t(x)$, $\dot{v}_t(x)$, $\dot{d}_t(x)$
$\kappa(A_t)$	$4.5106e+05$	$5.4676e+04$	$3.3417e+04$
$\epsilon(u_t(x))$	$6.900070e-07$	$9.694460e-07$	$1.025432e-08$
$\epsilon(v_t(x))$	$4.896091e-07$	$6.915281e-07$	$8.980188e-10$
$\epsilon(d_t(x))$	$2.053205e-07$	$2.962297e-07$	$5.693964e-10$
Average ϵ	$4.616455e-07$	$6.524013e-07$	$3.907244e-09$

Table 3.4: Error in the approximations of $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{d}_t(x)$ in Example 3.5.5

computed and shown to be an improved approximation over the standard least squares based method.

3.6 Conclusion

This chapter has considered the computation of the GCD of two univariate polynomials in Bernstein form and the main points of this chapter are now summarised:

The Sylvester Matrix : The Sylvester matrix and subresultant matrices for two univariate polynomials in Bernstein form were defined and several variants containing different combinations of binomial coefficients were discussed. The optimal variant of the kth subresultant matrix was given by $D^{-1}_{m+n-k}T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k$, which had both optimal scaling amongst its entries and the best separation between the numerically zero and non-zero singular values in its SVD. It was shown in Example 3.3.3 that this variant gave the best separation between the numerically zero and non-zero singular values in the SVD of the set of subresultant matrices.

GCD Degree Computation : Several methods for the computation of the degree of the GCD were considered, and SVD based methods, DC1 and DC2, were compared with QR decomposition based methods, DC3 and DC4.

Example 3.4.5 has shown that in some instances, the set of minimum singular values $\{\dot{\sigma}_k\}$ is not sufficient to determine the degree of the GCD, whereas observation of the complete set of singular values $\{\sigma_{k,i}\}$ reveals a clear separation between numerically zero and non-zero singular values and the degree of the GCD can correctly be identified. In the same example in Figure 3.17i it can be seen that DC1 is not the optimal method for determining the degree of the GCD.

A method which uses the SVD of the Bézoutian matrix was also included for comparison purposes, however this method failed to reliably determine the degree of the GCD in most cases where polynomials were inexactly defined.

Preprocessing : The two polynomials were preprocessed to minimise the ratio of the entry of maximum magnitude to the entry of minimum magnitude for each subresultant matrix. Examples have shown that the degree of the GCD of two polynomials is more reliably determined when the subresultant matrices contain preprocessed polynomials. Some efficient methods for the preprocessing operations were developed.

Coefficient Computation : Two methods were considered for computing approximations of the cofactor polynomials $\hat{u}_t(x)$ and $\hat{v}_t(x)$ and the GCD $\hat{d}_t(x)$. The first method was a simple least squares approach and the second method computed a structured low rank approximation of the tth subresultant matrix, from which coefficients of the cofactor polynomials were computed.

The low rank approximation based method was shown to give better results than the least squares method and the errors in the approximations are typically within the range of the initial input error.

Developing a robust method for the computation of the GCD of two univariate polynomials in Bernstein form is the first part of developing a method for polynomial square-free factorisation. The next chapter discusses modifications to this general purpose GCD finding method for use specifically in the square-free factorisation algorithm. The next chapter also considers methods for polynomial deconvolution and examples of polynomial root finding will be presented.

Chapter 4

The Univariate Polynomial Factorisation Algorithm

Chapter 3 discussed the UGCD method for the computation of the degree and coefficients of the GCD of two exact polynomials and the AGCD of two inexact polynomials in Bernstein form. Variants of the subresultant matrices were considered and various methods (DC1 - DC5) were considered for the computation of the degree of the GCD (or AGCD). It was also shown that a low rank approximation of the tth subresultant matrix typically yields improved results in the computation of the coefficients of the GCD.

GCD computation is only one part of the square-free factorisation algorithm (Algorithm 1), and the general purpose UGCD method developed in Chapter 3 can be specialised further for the problem found in square-free factorisation. In this chapter, the MUGCD method is developed, which is a refined version of UGCD and is used specifically as a part of the square-free factorisation algorithm.

Section 4.1 In this section the UGCD finding algorithm is modified for use in the square-free factorisation algorithm (Algorithm 1). This method, called MUGCD makes use of prior knowledge obtained from previous iterations of the square-free factorisation algorithm in order to quickly and reliably determine the degree of the ith GCD.

Section 4.2 This section considers the second part of the square-free factorisation algorithm. Several matrix based methods are considered for the deconvolution of the set of polynomials $\{\hat{f}_i(x)\}$. These deconvolution methods exploit the structure of the set of polynomials $\hat{f}_i(x)$ to give improved approximations of the set of polynomials $\{\hat{h}_i(x)\}$.

Section 4.3 In the final section of the chapter, the MUGCD method and the appropriate deconvolution method will be used to form a complete square-free factorisation algorithm, square-free factorisation (SQFF), and this method will be compared with other appropriate root finding methods. Some examples of univariate polynomial root finding will be given, and it will be shown that this method returns good results even in the presence of significant noise.

4.1 Modifications to the GCD Computation

The GCD computations in the square-free factorisation algorithm are structured such that polynomial $\hat{g}(x)$ is the derivative of $\hat{f}(x)$, so the general purpose GCD finder UGCD can be modified such that the degree of the GCD is more reliable computed, and in a computationally more efficient way.

4.1.1 Bounds on the Degree of the GCD of a Polynomial and its Derivative

This section describes how an upper and lower limit, K_{UB} and K_{LB} respectively, can be determined such that the degree t of the GCD of a polynomial and its derivative is known to be contained within the interval $[K_{LB}, K_{UB}]$ prior to its computation. The method for determining the upper and lower bound is now described.

Let $\hat{f}_0(x)$ be a univariate polynomial of degree M_0, with a set of d_0 distinct roots given by $\{ r_i \mid i = 1, \ldots, d_0 \}$, where each root r_i is of multiplicity m_i. The polynomial $\hat{f}_0(x)$ is written in terms of its factors as

$$\hat{f}_0(x) = (x - r_1)^{m_1}(x - r_2)^{m_2} \ldots (x - r_{d_0})^{m_{d_0}}$$

and its degree M_0 is given by

$$M_0 = m_1 + m_2 + \cdots + m_{d_0}.$$

The derivative of $\hat{f}_0(x)$ is given by

$$\frac{d\hat{f}_0(x)}{dx} = \hat{f}_0'(x) = (x - r_1)^{m_1 - 1}(x - r_2)^{m_2 - 1} \ldots (x - r_{d_0})^{m_{d_0} - 1} \times k_0(x),$$

where $k_0(x)$ is a polynomial of degree $(d_0 - 1)$.

As seen in Algorithm 1, the polynomial $\hat{f}_1(x)$ is defined as the GCD of $\hat{f}_0(x)$ and its derivative $\hat{f}_0'(x)$, and is given by

$$\hat{f}_1(x) = \text{GCD}\left(\hat{f}_0(x), \hat{f}_0'(x)\right) = (x - r_1)^{m_1 - 1}(x - r_2)^{m_2 - 1} \ldots (x - r_{d_0})^{m_{d_0} - 1},$$

which has degree $M_1 = M_0 - d_0$.

Suppose now that $\hat{f}_0$ is given in terms of its coefficients rather than its factorisation. The number of distinct factors d_0 is therefore unknown but can be determined by

$$d_0 = M_0 - M_1,$$

where M_0 and M_1 are the degrees of $\hat{f}_0(x)$ and $\hat{f}_1(x)$ respectively.

More generally, the number of distinct roots d_i of any of the polynomials in the set $\{\hat{f}_i(x) \mid i = 1, \ldots, m_{d_i}\}$ is a function of its degree M_i and the degree of $\hat{f}_{i+1}(x) = GCD(\hat{f}_i(x), \hat{f}_i'(x))$, and is given by

$$d_i = M_i - M_{i+1}.$$

The number of distinct roots of $\hat{f}_i(x)$ is only known after the computation of $\hat{f}_{i+1}(x)$. This can be used to determine a lower bound for the degree of the GCD$(\hat{f}_i(x), \hat{f}_i'(x))$ for any $i = 2, \ldots, m_{d_0}$.

Since the number of distinct roots of $\hat{f}_{i+1}(x)$ is less than or equal to the number of distinct roots of $\hat{f}_i(x)$, that is, $d_{i+1} \leq d_i$, then the lower bound of the degree of $\hat{f}_{i+1}(x)$ is given by

$$M_{i+1} \geq k_{LB} = M_i - d_{i-1} \quad \text{for } i = 1, \ldots, m_{d_0}. \tag{4.1}$$

This lower bound is only defined in the computation of the degree of $\hat{f}_2(x), \ldots, f_{m_{d_0}}(x)$.

The degree of $\hat{f}_0(x)$ given by M_0 is already known and the degree M_1 of $\hat{f}_1(x)$ is computed with knowledge of the upper bound only, that is $M_1 \leq M_0 - 1$. The degree computation for all subsequent $\hat{f}_i(x)$ is bounded by k_{LB} and k_{UB} as defined in (4.1).

In the case that $d_{i-1} = 1$, then $\hat{f}_{i-1}(x)$ has one distinct root, and the degree M_i of $\hat{f}_i(x) = \text{GCD}\left(\hat{f}_{i-1}(x), \hat{f}_{i-1}'(x)\right)$ is $M_{i-1} - 1$.

There are two aims of determining an upper and lower bound:

1. The first aim is to reduce the number of subresultant matrices which must be constructed, preprocessed, and have their numeric rank analysed.

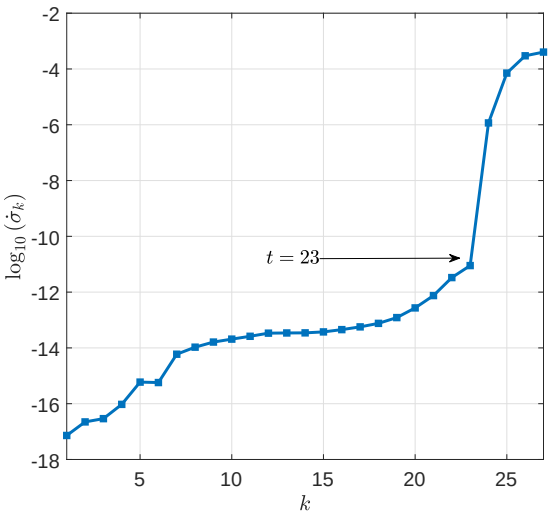
   When computing the degree of the GCD of two arbitrary polynomials, $\hat{f}(x)$ and $\hat{g}(x)$, the algorithm described in Chapter 3 requires that $\min(m, n)$ subresultant matrices be constructed and their numeric rank evaluated. In the univariate square-free factorisation algorithm a set of GCD computations are required, and the construction of the complete set of subresultant matrices for the computation of the GCD of each $\hat{f}_i(x)$ and its derivative $\hat{f}_i(x)'$ is a time consuming process.

   Rather than constructing and analysing the rank of each subresultant matrix $S_k(\hat{f}(x), \hat{f}'(x))$ for $k = 0, \ldots, \min(m, m-1)$, a lower bound k_{LB} is determined such that it is only necessary to consider a subset of the sequence of subresultant matrices, the set $\{S_k(\hat{f}(x), \hat{f}'(x)) \mid k = k_{LB}, \ldots, \min(m, m-1)\}$.

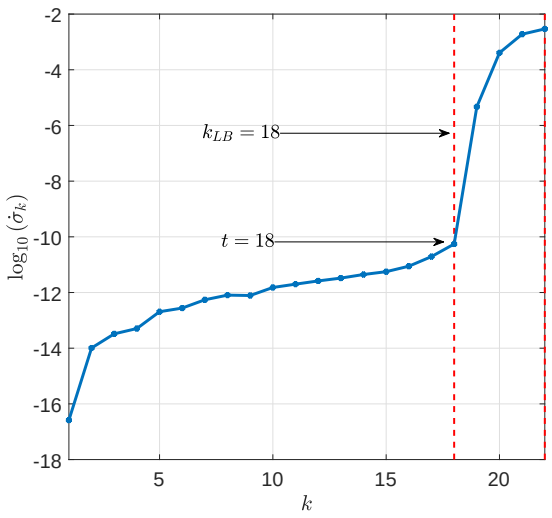
2. Alternatively, k_{LB} can be used to define a threshold in the computation of the degree of the GCD. Since it is known that all subresultant matrices $S_k(\hat{f}(x), \hat{g}(x))$ for $k = 0, \ldots, k_{LB}-1$ are numerically singular, the set of values $\{\rho_k \mid k = 1, \ldots, k_{LB}-1\}$ can be used to determine whether ρ_k for $k = k_{LB}, \ldots, \min(m, n)$ is indicative of $S_k(\hat{f}(x), \hat{g}(x))$ being a numerically singular or nonsingular subresultant matrix.

Example 4.1.1. Consider the Bernstein form of the exact polynomial $\hat{f}(x)$, whose factorisation is given by

$$\hat{f}(x) = (x - 1.5)^7(x - 0.17523547)^5(x - 0.1)^3(x + 0.75)^{10}(x + 1.2354)^3.$$



(i) The minimum singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{0,k}(\omega), \alpha_{0,k}\tilde{g}_{0,k}(\omega))\}$ for $k = 1, \ldots, 27$



(ii) The minimum singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{1,k}(\omega), \alpha_{1,k}\tilde{g}'_{1,k}(\omega))\}$ for $k = 1, \ldots, 22$

Figure 4.1: Computing the degree of the first and second GCD in the square-free factorisation of $\hat{f}(x)$ in Example 4.1.1

The exact polynomials $\{\, \hat{f}_i(x) \mid i = 0, \ldots, 10 \,\}$ are given by

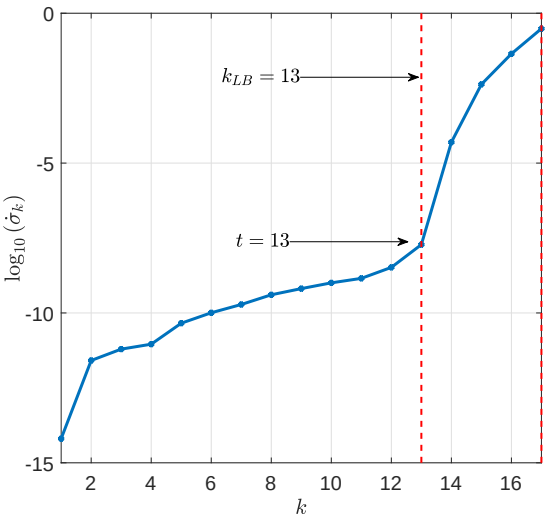
$$\hat{f}_i(x) = \begin{cases} (x-1.5)^{7-i}(x-0.17523547)^{5-i}(x-0.1)^{3-i}(x+0.75)^{10-i}(x+1.2354)^{3-i} & i = 0, 1, 2, \\ (x-1.5)^{7-i}(x-0.17523547)^{5-i}(x+0.75)^{10-i} & i = 3, 4, \\ (x-1.5)^{7-i}(x+0.75)^{10-i} & i = 5, 6, \\ (x+0.75)^{10-i} & i = 7, 8, 9, \\ 1 & i = 10, \end{cases}$$

where each $\hat{f}_{i+1}(x)$ is the GCD of $\hat{f}_i(x)$ and $\hat{f}'_i(x)$. The degrees of the set of polynomials $\{\hat{f}_i(x)\}$ are given by $\{m_i\} = \{28, 23, 18, 13, 10, 7, 5, 3, 2, 1\}$. Random noise is added to the coefficients of $\hat{f}(x)$ such that the perturbed polynomial $f(x)$ has coefficients $a_i = \hat{a}_i + \epsilon_i \hat{a}_i r_i$ for $i = 0, \ldots, m$, where $\{\epsilon_i\}$ are uniformly distributed random variables in the interval $\left[10^{-12}, 10^{-10}\right]$ and $\{r_i\}$ are uniformly distributed random variables in the interval $[-1, 1]$.

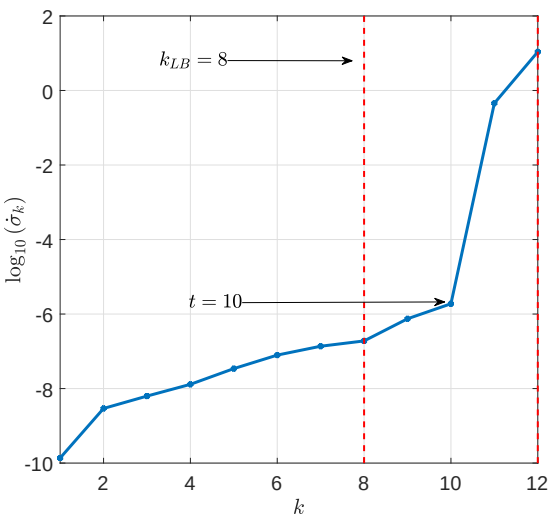
In the ith GCD computation, polynomials $f_i(x)$ and $f'_i(x)$ are preprocessed as described in Section 3.4 and the degree of the GCD $f_{i+1}(x)$ is computed by analysis of the set of minimum singular values of $\{S_k(\tilde{f}_{i,k}(\omega), \alpha_{i,k}\tilde{g}_{i,k}(\omega)) \mid k = 1, \ldots, m_i - 1\}$.

The notation $\tilde{f}_{i,k}(\omega)$ denotes the ith polynomial in the set $\{f_i(x)\}$ (computed by the square-free factorisation algorithm) having been preprocessed in the kth subresultant matrix. Also, the preprocessed form of $f'_i(x)$ is denoted $\alpha_{i,k}g_{i,k}(\omega)$ since the preprocessed polynomial is not necessarily the derivative of $\tilde{f}_i(\omega)$.

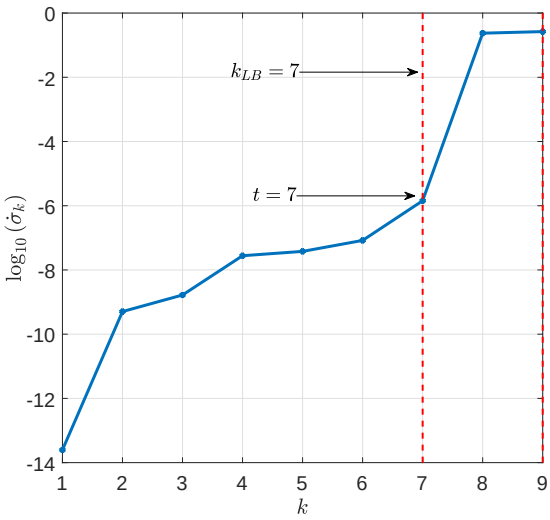
Figures 4.1 to 4.3 show the minimum singular values of each subresultant matrix $S_k(\tilde{f}_{i+1,k}(\omega), \alpha_{i+1,k}\tilde{f}'_{i+1,k}(\omega))$ in the computation of each GCD. The vertical dashed red lines in each of the figures indicate the computed upper and lower bound, k_{UB} and k_{LB} respectively, for the degree of the GCD.



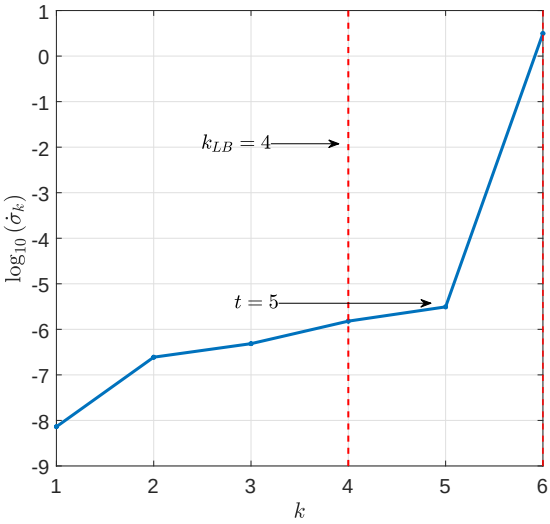
(i) The minimal singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{2,k}(\omega), \alpha_{2,k}\tilde{g}_{2,k}(\omega))\}$ for $k = 1, \ldots, 17$



(ii) The minimal singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{3,k}(\omega), \alpha_{3,k}\tilde{g}_{3,k}(\omega))\}$ for $k = 1, \ldots, 12$



(iii) The minimum singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{4,k}(\omega), \alpha_{4,k}\tilde{g}_{4,k}(\omega))\}$ for $k = 1, \ldots, 9$



(iv) The minimum singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{5,k}(\omega), \alpha_{5,k}\tilde{g}_{5,k}(\omega))\}$ for $k = 1, \ldots, 6$

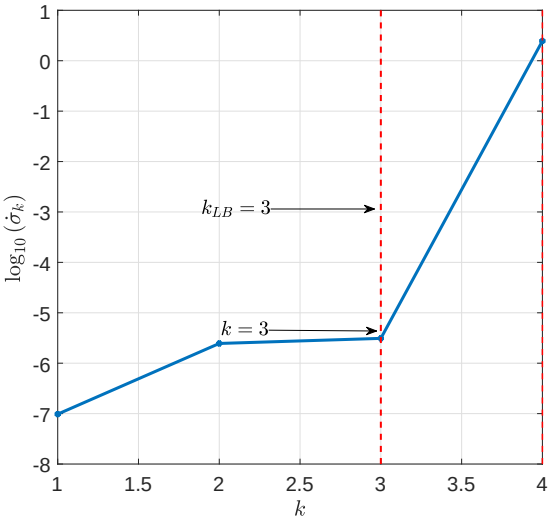
Figure 4.2: Computing the degree of the third, fourth, fifth and sixth GCD in the square-free factorisation of $\hat{f}(x)$ in Example 4.1.1

□

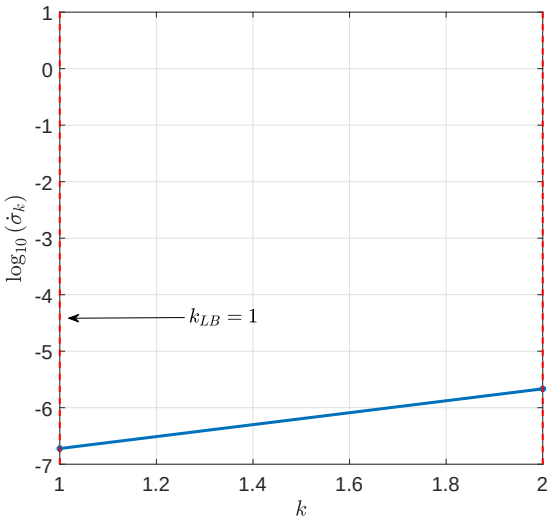
In this section a lower bound K_{LB} was derived for the computation of the degree of the ith GCD. This lower bound can be included in a fast version of the GCD finding algorithm used specifically in square-free factorisation, which can be significantly faster than UGCD. This depends on the multiplicity structure of the polynomial for which the square-free factorisation is computed. This fast method is referred to as MUGCD.

4.1.2 Bounds for Numerical Rank Determination

In the computation of the degree of the GCD of two arbitrary polynomials $\hat{f}(x)$ and $\hat{g}(x)$, a metric ρ_k of $S_k(\hat{f}(x), \hat{g}(x))$ is derived by one of several methods described in Section 3.2.



(i) The minimum singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{6,k}(\omega), \alpha_{6,k}\tilde{g}_{6,k}(\omega))\}$ for $k = 1, \ldots, 4$



(ii) The minimum singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{7,k}(\omega), \alpha_{7,k}\tilde{g}_{7,k}(\omega))\}$ for $k = 1, 2$

Figure 4.3: Computing the degree of the seventh and eighth GCD in the square-free factorisation of $\hat{f}(x)$ in Example 4.1.1

The value of ρ_k gives an indication as to whether the kth subresultant matrix is singular or nonsingular, and when the complete set $\{\, \rho_k \mid k = 1, \ldots, \min(m, n) \,\}$ is considered, the degree of the GCD can be computed from a maximum change in the set of values.

In the square-free factorisation algorithm, suppose t_i is the degree of $\hat{f}_i(x) = GCD(\hat{f}_{i-1}(x), \hat{f}'_{i-1}(x))$, then ρ_{t_i} is indicative of a numerically singular matrix while ρ_{t_i+1} is indicative of a nonsingular matrix. These two values can be used as threshold values, $\phi_{i,LB}$ and $\phi_{i,UP}$ respectively, in the computation of the degree of $\hat{f}_{i+1} = GCD(\hat{f}_i(x), \hat{f}'_i(x))$. It is typically observed that $\phi_{i,LB}$ and $\phi_{i,UP}$ closely approximate $\rho_{t_{i+1}}$ and $\rho_{t_{i+1}+1}$ in the subsequent GCD finding problem as shown in the following example.

Example 4.1.2. Consider the Bernstein form of the exact polynomial $\hat{f}(x)$ of degree $m = 11$, whose factorisation is given by

$$\hat{f}(x) = (x - 0.5)^4 (x + 0.75)^7.$$

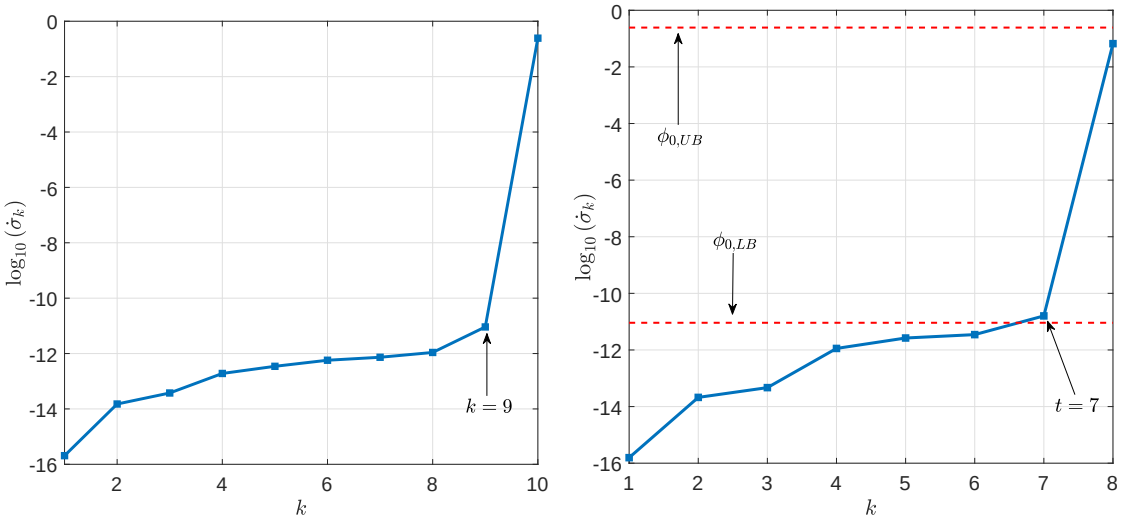
The polynomials $\hat{f}_0(x), \ldots, \hat{f}_7(x)$ are given by

$$\hat{f}_i(x) = \begin{cases} (x - 0.5)^{4-i}(x + 0.75)^{7-i} & i = 0, 1, 2, 3, \\ (x + 0.75)^{7-i} & i = 4, 5, 6, \\ 1 & i = 7, \end{cases}$$

where each $\hat{f}_{i+1}(x)$ is the GCD of $\hat{f}_i(x)$ and its derivative $\hat{f}'_i(x)$, the degrees of which are given by $\{M_i\} = \{11, 9, 7, 5, 3, 2, 1, 0\}$.

The degree of $\hat{f}_1(x) = GCD(\hat{f}_0(x), \hat{f}'_0(x))$ is determined by analysis of the minimum singular values $\{\dot{\sigma}_k\}$ of $S_k(\tilde{f}_{0,k}(\omega), \alpha_{0,k}\tilde{g}_{0,k}(\omega))$, and the set of values $\{\dot{\rho}_k = \log_{10}(\dot{\sigma}_k)\}$ are plotted in Figure 4.4i, from which the degree of $\hat{f}_0(x)$ is given by $t_1 = 9$.

Since $t_1 = 9$, the ninth subresultant matrix $S_9(\tilde{f}_{0,9}(\omega), \alpha_{0,9}\tilde{g}_{0,9}(\omega))$ is singular while



(i) The minimum singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{0,k}(\omega), \alpha_{0,k}\tilde{g}_{0,k}(\omega))\}$

(ii) The minimum singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{1,k}(\omega), \alpha_{1,k}\tilde{g}_{1,k}(\omega))\}$

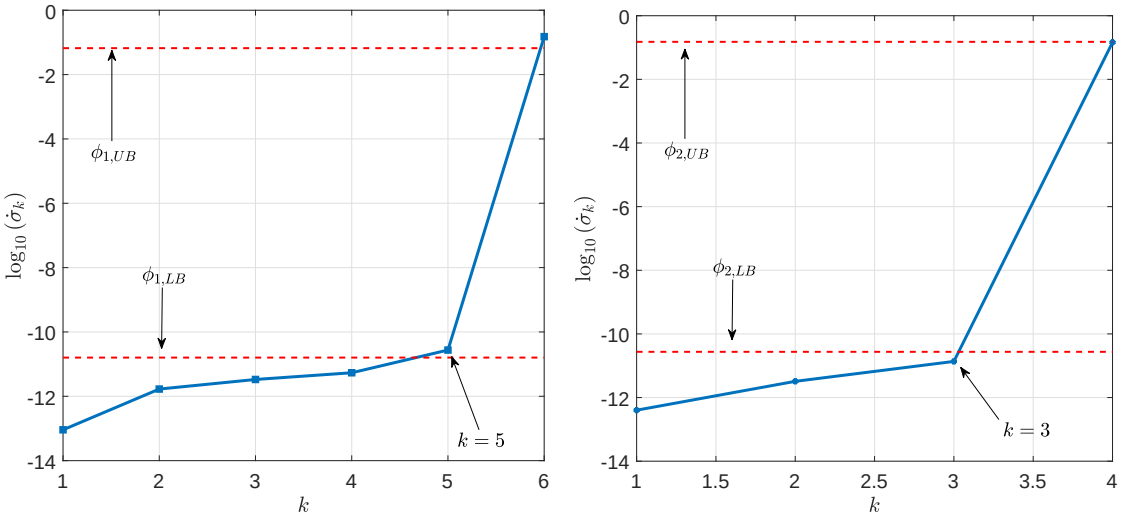
Figure 4.4: The minimum singular values $\{\dot{\sigma}_k\}$ in the first and second GCD computation of the square-free factorisation problem in Example 4.1.2

$S_{10}(\tilde{f}_{0,10}(\omega), \alpha_{0,10}\tilde{g}_{0,10}(\omega))$ is nonsingular, so the values $\dot{\rho}_9 = \log_{10}(\dot{\sigma}_9) \approx -11$ and $\dot{\rho}_{10} = \log_{10}(\dot{\sigma}_{10}) \approx -1$ are used as threshold values in the subsequent GCD computation. That is, $\phi_{0,LB} = \dot{\rho}_9$ and $\phi_{0,UB} = \rho_{10}$. These are used in computing the degree of $f_2(x)$, where $f_2(x) = \text{GCD}(f_1(x), f_1'(x))$. The minimum singular values of subresultant matrices $S_k(\tilde{f}_{1,k}(\omega), \alpha_{1,k}\tilde{g}_{1,k}(\omega))$ are plotted in Figure 4.4ii and horizontal red dotted lines indicate the values of $\phi_{0,UB}$ and $\phi_{0,LB}$. These values closely approximate the values $\dot{\rho}_7 = \log_{10}(\dot{\sigma}_7)$ and $\dot{\rho}_8 = \log_{10}(\dot{\sigma}_8)$ respectively, and it is determined that the subresultant matrix $S_7(\tilde{f}_{1,7}(\omega), \alpha_{1,7}\tilde{g}_{1,7}(\omega))$ is numerically rank deficient while the matrix $S_8(\hat{f}_{1,8}(\omega), \alpha_{1,8}\hat{g}_{1,8}(\omega))$ is of full rank. The values $\phi_{1,LB}$ and $\phi_{1,UB}$ are therefore given by ρ_7 and ρ_8 respectively.

Values $\phi_{1,LB}$, $\phi_{1,UB}$, $\phi_{2,LB}$ and $\phi_{2,UB}$ are similarly computed and used in the computations of the degree of $f_3(x)$ and $f_4(x)$ in Figure 4.5i and Figure 4.5ii respectively. The entries of the set $\{\phi_{0,LB}, \ldots, \phi_{3,LB}\}$ are approximately equal, as are the entries of the set $\{\phi_{0,UB}, \ldots, \phi_{3,UB}\}$, and this is typical for many examples. This approximation of a numerically zero singular value is particularly useful when in the jth GCD computation $\delta\dot{\rho}_i$ is insignificant for all values $i = 1, \ldots, \min(m_j, m_j - 1)$ and it must be determined whether all subresultant matrices are numerically rank deficient or of full rank.

In Figure 4.6, given the horizontal red dotted lines, the minimum singular values $\dot{\sigma}_1$ and $\dot{\sigma}_2$ of the subresultant matrices $S_1(\tilde{f}_{4,1}(\omega), \alpha_{4,1}\tilde{f}_{4,1}'(\omega))$ and $S_2(\alpha_{4,2}\tilde{f}_{4,2}(\omega), \tilde{f}_{4,2}'(\omega))$ are both deemed to be numerically zero and therefore the degree of $f_5(x)$ is given by $t_5 = 2$. □

This section has considered methods to adapt the UGCD method developed in Chapter 3, such that the GCD of two polynomials can be computed in the specific case where $\hat{g}(x)$ is the derivative of $\hat{f}(x)$ and forms a part of the square-free factorisation problem. Refinements to the GCD finding method were discussed, specifically for the GCD problems which arise in the square-free factorisation algorithm (Algorithm 1). These modifications



(i) The minimum singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{2,k}(\omega), \alpha_{2,k}\tilde{g}_{2,k}(\omega))\}$

(ii) The minimum singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{3,k}(\omega), \alpha_{3,k}\tilde{g}_{3,k}(\omega))\}$

Figure 4.5: The minimum singular values $\{\dot{\sigma}_k\}$ in the third and fourth GCD computation of the square-free factorisation problem in Example 4.1.2

give a significantly more efficient and reliable algorithm.

In the MUGCD method, only a subset of subresultant matrices, $S_{LB}(f,g), \ldots, S_{\min(m,n)}(f,g)$, are required, since it is known that the degree of the GCD lies in the interval $[k_{LB}, \min(m,n)]$. In the square-free factorisation of a polynomial with few roots of high multiplicity, this can be significantly faster than considering all subresultant matrices $S_1(f,g), \ldots, S_{\min(m,n)}(f,g)$ for each iteration of the algorithm.

The MUGCD method is also more reliable. It is known that the subresultant matrices $S_1(f,g), \ldots, S_{LB}(f,g)$ are singular, and therefore the ranges of numerically zero and non-zero singular values can be approximated for the singular values of $S_{LB+1}(f,g), \ldots, S_{\min(m,n)}(f,g)$.

The second stage of the square-free factorisation algorithm, deconvolution, must now be considered.

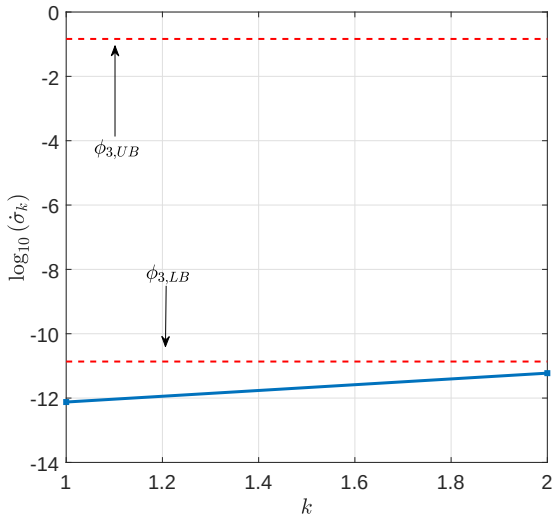


Figure 4.6: The minimum singular values $\{\dot{\sigma}_k\}$ of the subresultant matrices $\{S_k(\tilde{f}_{4,k}(\omega), \alpha_{4,k}\tilde{g}_{4,k}(\omega))\}$ in the fifth GCD computation of the square-free factorisation problem in Example 4.1.2

4.2 Deconvolution in the Factorisation Algorithm

The previous section discussed modifications to the general purpose GCD finding method for use in the square-free factorisation algorithm (Algorithm 1). The second stage of this algorithm is the computation of the set of polynomials $\{\hat{h}_i(x)\}$ by a set of divisions over the polynomials $\hat{f}_i(x)$, where each $\hat{h}_i$ is given by $\hat{f}_{i-1}/\hat{f}_i$. Rather than the exact polynomials $\hat{f}_i(x)$, the inexact polynomials $\{f_i(x)\}$ must be deconvolved. It is assumed that these polynomials are inexact since they are outputs of a sequence of non-exact GCD computations performed in a floating-point environment. The GCD computations are iterative, so errors are likely to propagate.

This section develops a robust matrix based method for the computation of approximations of the set of polynomials $\{\hat{h}_i(x)\}$, which utilises knowledge of the number of distinct roots of the set of polynomials $\{f_i(x)\}$. Five methods of polynomial deconvolution by matrix methods are considered and are given as follows.:

1. **Separate Deconvolution (SD) :** In Section 4.2.1 each of the deconvolutions $\frac{\hat{f}_{i-1}(x)}{\hat{f}_i(x)}$ for $i = 1, \ldots, m$ are performed independently and the approximations of $\{\hat{h}_i(x)\}$ are obtained by simple least squares.

2. **Batch Deconvolution (BD) :** In Section 4.2.2 the polynomial deconvolutions are performed simultaneously using a structured matrix method. Each polynomial in the set $\{\hat{h}_i(x)\}$ is given by $\hat{h}_i(x) = \frac{\hat{f}_{i-1}(x)}{\hat{f}_i(x)}$, so each $\hat{f}_i(x)$ is the divisor in the ith division and the numerator in the $(i+1)$th. The method described as batch deconvolution takes advantage of this structure.

3. **Batch Deconvolution with STLN (BDSTLN) :** It is likely that the polynomials $f_i(x)$ in the set $\{f_i(x)\}$ are inexact, and in Section 4.2.3 the method of deconvolution is extended to include the computation of minimal perturbations $\delta f_i(x)$, such that $f_i(x) + \delta f_i(x) \approx \hat{f}_i(x)$ and the polynomials $\{\hat{h}_i(x)\}$ are approximated with minimal error.

4. **Constrained Batch Deconvolution (CBD) :** In Section 4.2.4 further structure is added to the set of deconvolutions. By the degree structure of the set of polynomials $\{f_i(x)\}$ and the number of distinct roots in each $f_i(x)$, certain subsets of the polynomials $\{\hat{h}_i(x)\}$ are equal and the matrix deconvolution method is altered accordingly.

5. **Constrained Batch Deconvolution with STLN (CBDSTLN) :** It is likely that the polynomials in the set $\{f_i(x)\}$ are inexact, and in Section 4.2.5 polynomials $\delta f_i(x)$ are computed such that the polynomials $\hat{h}_i(x)$ are approximated by the deconvolution of the set $\{f_i(x) + \delta f_i(x)\}$ using the batch constrained method.

4.2.1 Separate Deconvolution (SD)

Each polynomial $\hat{h}_i(x,y)$ can be computed by an independent separate deconvolution. Let the ith polynomial in the set of polynomials $\hat{f}_i(x,y)$ be given by

$$\hat{f}_i(x) = \sum_{j=0}^{m_i} \hat{a}_{i,j} \binom{m_i}{j} (1-x)^{m_i-j} x^j, \tag{4.2}$$

where m_i is the degree of the polynomial $\hat{f}_i(x)$. The polynomial $\hat{h}_i(x)$ of degree $n_i = m_{i-1} - m_i$ is given by

$$\hat{h}_i(x) = \frac{\hat{f}_{i-1}(x)}{\hat{f}_i(x)} \qquad \text{for} \qquad i = 1, \ldots, \mu.$$

The polynomial division can be written in matrix form as

$$C_{n_i}\left(\hat{f}_i(x)\right) \hat{\mathbf{h}}_i(x) = \hat{\mathbf{f}}_{i-1}, \tag{4.3}$$

where $C_{n_i}(\hat{f}_i(x)) \in \mathbb{R}^{(m_{i-1}+1)\times(n_i+1)}$ is the matrix given by

$$C_{n_i}\left(\hat{f}_i(x)\right) = D_{m_{i-1}}^{-1} T_{n_i}\left(\hat{f}_i(x)\right) Q_{n_i}.$$

The matrix $D_{m_{i-1}}^{-1} \in \mathbb{R}^{(m_{i-1}+1)\times(m_{i-1}+1)}$ is given by

$$D_{m_{i-1}}^{-1} = \operatorname{diag}\left[\begin{array}{cccc} \frac{1}{\binom{m_{i-1}}{0}}, & \frac{1}{\binom{m_{i-1}}{1}}, & \cdots, & \frac{1}{\binom{m_{i-1}}{m_{i-1}}} \end{array} \right], \tag{4.4}$$

the matrix $T_{n_i}(\hat{f}_i(x)) \in \mathbb{R}^{(m_{i-1}+1)\times(n_i+1)}$ is given by

$$T_{n_i}\left(\hat{f}_i(x)\right) = \begin{bmatrix} \hat{a}_{i,0}\binom{m_i}{0} & & & \\ \hat{a}_{i,1}\binom{m_i}{1} & \hat{a}_{i,0}\binom{m_i}{0} & & \\ \vdots & \hat{a}_{i,1}\binom{m_i}{1} & \ddots & \\ \hat{a}_{i,m_i-1}\binom{m_i}{m_i-1} & \vdots & \ddots & \hat{a}_{i,0}\binom{m_i}{0} \\ \hat{a}_{i,m_i}\binom{m_i}{m_i} & \hat{a}_{i,m_i-1}\binom{m_i}{m_i-1} & & \hat{a}_{i,1}\binom{m_i}{1} \\ & \hat{a}_{i,m_i}\binom{m_i}{m_i} & \ddots & \vdots \\ & & \ddots & \hat{a}_{i,m_i-1}\binom{m_i}{m_i-1} \\ & & & \hat{a}_{i,m_i}\binom{m_i}{m_i} \end{bmatrix} \tag{4.5}$$

and the matrix $Q_{n_i} \in \mathbb{R}^{(n_i+1)\times(n_i+1)}$ is given by

$$Q_{n_i} = \operatorname{diag}\left[\begin{array}{cccc} \binom{n_i}{0}, & \ldots, & \binom{n_i}{n_i} \end{array} \right]. \tag{4.6}$$

The vector $\hat{\mathbf{h}}_i \in \mathbb{R}^{n_i+1}$ in (4.3) is given by

$$\hat{\mathbf{h}}_i = \left[\begin{array}{cccc} \hat{h}_{i,0}, & \hat{h}_{i,1}, & \ldots, & \hat{h}_{i,n_i} \end{array} \right]^T, \tag{4.7}$$

which contains the coefficients of the polynomial $\hat{h}_i(x)$ and is computed by a simple least squares method. The vector $\hat{\mathbf{f}}_i$ consists of the coefficients of the polynomial $\hat{f}_i(x)$ and is

given by

$$\hat{\mathbf{f}}_i = \left[\begin{array}{cccc} \hat{a}_{i,0}, & \hat{a}_{i,1}, & \ldots, & \hat{a}_{i,m_i} \end{array} \right]^T. \tag{4.8}$$

4.2.2 Batch Deconvolution (BD)

In the previous section the deconvolution problem was considered as a set of independent problems in which the coefficients of each polynomial $\hat{f}_i(x)$ for $i = 1, \ldots, \mu - 1$ appear in a convolution matrix in the ith deconvolution and a vector in the $(i+1)$th deconvolution. The coefficients of the polynomial $\hat{f}_0(x)$ only appear as a vector in the first deconvolution, and the coefficients of $\hat{f}_\mu(x)$ are only contained in a coefficient matrix in the μ^{th} deconvolution.

The values M and N are defined as

$$\begin{aligned} M &= (m_0 + 1) + (m_1 + 1) + \cdots + (m_{\mu-1} + 1) \\ &= m_0 + m_1 + \cdots + m_{\mu-1} + \mu \\ N &= (n_1 + 1) + (n_2 + 1) + \cdots + (n_\mu + 1) \\ &= n_1 + n_2 + \cdots + n_\mu + (\mu - 1) \end{aligned}$$

and these assist in the description of the coefficient matrix for the set of deconvolutions.

The set of deconvolutions can be written in matrix form as

$$C\left(\hat{f}_1(x), \ldots, \hat{f}_\mu(x)\right) \hat{\mathbf{h}} = \hat{\mathbf{f}}. \tag{4.9}$$

The matrix $C(\hat{f}_1(x), \ldots, \hat{f}_\mu(x)) \in \mathbb{R}^{M \times N}$ is given by

$$\left[\begin{array}{cccc} D_{m_0}^{-1} T_{n_1}\left(\hat{f}_1(x)\right) Q_{n_1} & & & \\ & D_{m_1}^{-1} T_{n_2}\left(\hat{f}_2(x)\right) Q_{n_2} & & \\ & & \ddots & \\ & & & D_{m_{\mu-1}}^{-1} T_{n_\mu}\left(\hat{f}_\mu\right) Q_{n_\mu} \end{array} \right]$$

or alternatively the coefficient matrix is given by

$$C\left(\tilde{f}_1(x), \tilde{f}_2(x), \ldots, \tilde{f}_\mu(x)\right) = D^{-1} T\left(\tilde{f}_1(x), \tilde{f}_2(x), \ldots, \tilde{f}_\mu(x)\right) \hat{Q}. \tag{4.10}$$

The block diagonal matrix $D^{-1} \in \mathbb{R}^{M \times M}$ is given by

$$D^{-1} = \operatorname{diag}\left[\begin{array}{cccc} D_{m_0}^{-1}, & D_{m_1}^{-1}, & \ldots, & D_{m_{\mu-1}}^{-1} \end{array} \right], \tag{4.11}$$

where each partition on the diagonal $D_{m_i}^{-1} \in \mathbb{R}^{(m_i+1)\times(m_i+1)}$ has the same structure as the matrix defined in (4.4). The block diagonal matrix $T(\hat{f}_1(x), \hat{f}_2(x), \ldots, \hat{f}_\mu(x)) \in \mathbb{R}^{M \times N}$ is given by

$$\operatorname{diag}\left[\begin{array}{ccc} T_{n_1}\left(\hat{f}_1(x)\right), & T_{n_2}\left(\hat{f}_2(x)\right), & \ldots, \quad T_{n_\mu}\left(\hat{f}_\mu(x)\right) \end{array} \right], \tag{4.12}$$

where each $T_{n_i}(\hat{f}_i(x))$ is of the structure defined in (4.5). The block diagonal matrix

$\hat{Q} \in \mathbb{R}^{N\times N}$ is given by

$$\hat{Q} = \text{diag}\left[\ Q_{n_1},\quad Q_{n_2},\quad \ldots,\quad Q_{n_\mu}\ \right], \tag{4.13}$$

where each $Q_{n_i} \in \mathbb{R}^{(n_i+1)\times(n_i+1)}$ is defined in (4.6).

A partition $D_{m_{i-1}}^{-1} T_{n_i}(\hat{f}_i(x)) Q_{n_i}$ has the structure

$$\begin{bmatrix}
\frac{\hat{a}_{i,0}\binom{m_i}{0}\binom{n_i}{0}}{\binom{m_{i-1}}{0}} & & & \\
\frac{\hat{a}_{i,1}\binom{m_i}{1}\binom{n_i}{0}}{\binom{m_{i-1}}{1}} & \frac{\hat{a}_{i,0}\binom{m_i}{0}\binom{n_i}{1}}{\binom{m_{i-1}}{1}} & \ddots & \\
\frac{\hat{a}_{i,2}\binom{m_i}{2}\binom{n_i}{0}}{\binom{m_{i-1}}{2}} & \frac{\hat{a}_{i,1}\binom{m_i}{1}\binom{n_i}{1}}{\binom{m_{i-1}}{2}} & \ddots & \frac{\hat{a}_{i,0}\binom{m_i}{0}\binom{n_i}{n_i}}{\binom{m_{i-1}}{m_{i-1}-m_i}} \\
\vdots & \frac{\hat{a}_{i,2}\binom{m_i}{2}\binom{n_i}{1}}{\binom{m_{i-1}}{3}} & & \frac{\hat{a}_{i,1}\binom{m_i}{1}\binom{n_i}{n_i}}{\binom{m_{i-1}}{m_{i-1}-m_i+1}} \\
\frac{\hat{a}_{i,m_i}\binom{m_i}{m_i}\binom{n_i}{0}}{\binom{m_{i-1}}{m_i}} & \vdots & & \frac{\hat{a}_{i,2}\binom{m_i}{2}\binom{n_i}{n_i}}{\binom{m_{i-1}}{m_{i-1}-m_i+2}} \\
 & \frac{\hat{a}_{i,m_i}\binom{m_i}{m_i}\binom{n_i}{2}}{\binom{m_{i-1}}{m_i+1}} & \ddots & \vdots \\
 & & & \frac{\hat{a}_{i,m_i}\binom{m_i}{m_i}\binom{n_i}{n_i}}{\binom{m_{i-1}}{m_{i-1}}}
\end{bmatrix}.$$

The vector $\hat{\mathbf{f}} \in \mathbb{R}^{M\times 1}$ in (4.9) is given by

$$\hat{\mathbf{f}} = \left[\ \hat{\mathbf{f}}_0,\quad \hat{\mathbf{f}}_1,\quad \ldots,\quad \hat{\mathbf{f}}_{\mu-1}\ \right]^T,$$

where the vectors $\hat{\mathbf{f}}_i \in \mathbb{R}^{(m_{i+1})\times 1}$ are defined in (4.8). Similarly, the vector $\hat{\mathbf{h}} \in \mathbb{R}^{N\times 1}$ in (4.9) is given by

$$\left[\ \hat{\mathbf{h}}_1,\quad \hat{\mathbf{h}}_2,\quad \ldots,\quad \hat{\mathbf{h}}_{\mu-1},\quad \hat{\mathbf{h}}_\mu\ \right]^T,$$

where the vectors $\hat{\mathbf{h}}_i$ are of the form (4.7).

As with the first method described in Section 4.2.1, this problem is also solved by simple least squares.

Computations on polynomials whose coefficients vary widely in magnitude may be unreliable, and preprocessing the set of polynomials $\{f_i(x)\}$ which are defined in (4.2) is necessary. The substitution $x = \theta\omega$ is considered, and the optimal value of θ is determined by a linear programming problem such that the ratio of maximum entry of the preprocessed $D^{-1}T(\tilde{f}_1(\omega),\ldots,\tilde{f}_\mu(\omega))\hat{Q}$ to the minimum entry of $D^{-1}T(\tilde{f}_1(\omega),\ldots,\tilde{f}_\mu(\omega))\hat{Q}$ is minimal. The polynomials in the set $\{\ddot{f}_k(\theta,\omega)\}$ are unoptimised in the independent variable ω and are defined by

$$\ddot{f}_k(\theta,\omega) = \sum_{i=0}^{m_k} \hat{a}_{k,i}\theta^j \binom{m_k}{i}\left[(1-\theta\omega)^{m_k-j}\omega^k\right] \quad \text{for } k = 0,\ldots,\mu. \tag{4.14}$$

A general expression for a non-zero element in the first $(n_1 + 1)$ columns of

$D^{-1}T(\ddot{f}_1(x),\ldots,\ddot{f}_\mu(x))Q$, that is, in the block $D_{m_0}^{-1}T_{n_1}(\ddot{f}_1(\theta,\omega))Q_{n_1}$, is

$$C_{n_1}\left(\ddot{f}_1(\theta,\omega)\right)_{(i+j+1,j+1)} = \begin{cases} \frac{\hat{a}_{1,i}\theta^i\binom{m_1}{i}\binom{n_1}{j}}{\binom{m_0}{i+j}} & i=0,\ldots,m_1; \quad j=0,\ldots,n_1, \\ 0 & \text{otherwise.} \end{cases}$$

More generally, the kth block $D_{m_{k-1}}^{-1}T_{n_k}(\ddot{f}_k(\theta,\omega))Q_{n_k}$ contains entries of the form

$$C_{n_k}\left(\ddot{f}_k(\theta,\omega)\right)_{(i+j+1,j+1)} = \begin{cases} \frac{\hat{a}_{k,i}\theta^i\binom{m_k}{i}\binom{n_k}{j}}{\binom{m_{k-1}}{i+j}}, & i=0,\ldots,m_i;\ j=0,\ldots,n_i, \\ 0 & \text{otherwise.} \end{cases}$$

It is convenient to define the sets $\mathscr{P}_k(\theta)$ for $k=1,\ldots,\mu$, where

$$\mathscr{P}_k(\theta) = \left\{ \frac{\left|\hat{a}_{k,i}\theta^i\binom{m_k}{i}\binom{n_k}{j}\right|}{\binom{m_{k-1}}{i+j}} \;\middle|\; i=0,\ldots,m_k;\ j=0,\ldots,n_k \right\},$$

such that the minimisation problem to determine the optimal value of θ is written as

$$\theta_0 = \arg\min_\theta \left\{ \frac{\max\{\,\mathscr{P}_k(\theta) \mid k=1,\ldots,\mu\,\}}{\min\{\,\mathscr{P}_k(\theta) \mid k=1,\ldots,\mu\,\}} \right\}.$$

This problem can be written as

$$\begin{aligned} &\text{Minimise} \quad \frac{t}{s} \\ &\text{Subject to} \\ &t \geq \frac{\left|\hat{a}_{k,i}\theta^i\binom{m_k}{i}\binom{n_k}{j}\right|}{\binom{m_{k-1}}{i+j}} \qquad k=1,\ldots,\mu; \quad i=0,\ldots,m_k; \quad j=0,\ldots,n_k, \\ &s \leq \frac{\left|\hat{a}_{k,i}\theta^i\binom{m_k}{i}\binom{n_k}{j}\right|}{\binom{m_{k-1}}{i+j}} \qquad k=1,\ldots,\mu; \quad i=0,\ldots,m_k; \quad j=0,\ldots,n_k, \\ &s>0, \\ &\theta>0. \end{aligned}$$

By the transformations

$$T=\log_{10}(t), \quad S=\log_{10}(s), \quad \bar{\phi}=\log_{10}(\theta) \quad \text{and} \quad \bar{\alpha}_{k,i,j}=\log_{10}\left(\frac{\left|\hat{a}_{k,i}\binom{m_k}{i}\binom{n_k}{j}\right|}{\binom{m_{k-1}}{i+j}}\right),$$

the constrained minimisation problem can be written as

$$\begin{array}{l}\text{Minimise} \qquad T - S \\ \text{subject to} \\ \begin{array}{llcll} T & -i\bar{\phi} & \geq & \bar{\alpha}_{1,i,j} & i = 0,\ldots,m_1;\quad j = 0,\ldots,n_1, \\ T & -i\bar{\phi} & \geq & \bar{\alpha}_{2,i,j} & i = 0,\ldots,m_2;\quad j = 0,\ldots,n_2, \\ \vdots & \vdots & & \vdots & \vdots \\ T & -i\bar{\phi} & \geq & \bar{\alpha}_{\mu,i,j} & i = 0,\ldots,m_\mu;\quad j = 0,\ldots,n_\mu, \\ -S & +j\bar{\phi} & \geq & -\bar{\alpha}_{1,i,j} & i = 0,\ldots,m_1;\quad j = 0,\ldots,n_1, \\ -S & +j\bar{\phi} & \geq & -\bar{\alpha}_{2,i,j} & i = 0,\ldots,m_2;\quad j = 0,\ldots,n_2, \\ \vdots & \vdots & & \vdots & \vdots \\ -S & +j\bar{\phi} & \geq & -\bar{\alpha}_{\mu,i,j} & i = 0,\ldots,m_\mu;\quad j = 0,\ldots,n_\mu. \end{array} \end{array} \tag{4.15}$$

Since the counter j only appears on the left hand side of the inequalities, let $\bar{\lambda}_{k,i}$ and $\bar{\psi}_{k,i}$ be defined as

$$\begin{array}{lcll} \bar{m}_{k,i} & = & \max\{\bar{\alpha}_{k,i,j} \mid j = 0,\ldots,0,\ldots,n_k\} & k = 0,\ldots,\mu;\quad i = 0,\ldots,m_k \\ \bar{m}_{k,i} & = & \min\{\bar{\alpha}_{k,i,j} \mid j = 0,\ldots,n_k\} & k = 0,\ldots,\mu;\quad i = 0,\ldots m_k, \end{array} \tag{4.16}$$

then the above minimisation problem (4.15) can be written as

$$\begin{array}{l}\text{Minimise} \quad T - S \\ \text{subject to} \\ \begin{array}{llcll} T & -i\bar{\phi} & \geq & \bar{m}_{1,i} & i = 0,\ldots,m_1 \\ T & -i\bar{\phi} & \geq & \bar{m}_{2,i} & i = 0,\ldots,m_2 \\ \vdots & \vdots & & \vdots & \vdots \\ T & -i\bar{\phi} & \geq & \bar{m}_{\mu,i}, & i = 0,\ldots,m_\mu \\ -S & +i\bar{\phi} & \geq & -\bar{m}_{1,i} & i = 0,\ldots,m_1 \\ -S & +i\bar{\phi} & \geq & -\bar{m}_{2,i} & i = 0,\ldots,m_2 \\ \vdots & \vdots & & \vdots & \vdots \\ -S & +i\bar{\phi} & \geq & -\bar{m}_{1,i} & i = 0,\ldots,m_\mu \end{array} \end{array}.$$

The minimisation problem can be written in matrix form as

$$\text{Minimise} \qquad \begin{bmatrix} 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} T \\ S \\ \bar{\phi} \end{bmatrix} \quad \text{Subject to} \quad A \begin{bmatrix} T \\ S \\ \bar{\phi} \end{bmatrix} \geq \mathbf{b}. \tag{4.17}$$

The matrix $A \in \mathbb{R}^{(m_1+m_2+\cdots+m_\mu)\times 3}$ in (4.17) is given by

$$A = \begin{bmatrix} \mathcal{A}_1, & \mathcal{A}_2, & \ldots, & \mathcal{A}_\mu, & a_1, & a_2, & \ldots, & a_\mu \end{bmatrix}^T,$$

where the matrices $\mathcal{A}_k$, $a_k \in \mathbb{R}^{(m_k+1)\times 3}$ are given by

$$\mathcal{A}_k = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & -1 \\ \vdots & \vdots & \vdots \\ 1 & 0 & -(m_k-1) \\ 1 & 0 & -m_k \end{bmatrix}, \quad a_k = \begin{bmatrix} 0 & -1 & 0 \\ 0 & -1 & 1 \\ \vdots & \vdots & \vdots \\ 0 & -1 & (m_k-1) \\ 0 & -1 & m_k \end{bmatrix}.$$

The vector $\mathbf{b}$ in (4.17) is given by

$$\mathbf{b} = \begin{bmatrix} \bar{m}_1, & \bar{m}_2, & \ldots, & \bar{m}_\mu, & -\bar{m}_1, & -\bar{m}_2, & \ldots, & -\bar{m}_\mu \end{bmatrix}^T,$$

where the vectors $\bar{m}_k, \bar{m}_k \in \mathbb{R}^{m_k+1}$ are given by

$$\bar{m}_k = \begin{bmatrix} \bar{m}_{k,0}, & \bar{m}_{k,1}, & \ldots, & \bar{m}_{k,m_k} \end{bmatrix}^T, \quad \bar{m}_k = \begin{bmatrix} \bar{m}_{k,0}, & \bar{m}_{k,1}, & \ldots, & \bar{m}_{k,m_k} \end{bmatrix}^T.$$

The linear programming problem returns the value $\theta_{(0)}$ and it follows that the set of preprocessed polynomials are given by

$$\tilde{f}_i(\omega) = \sum_{j=0}^{m_i} \hat{a}_{i,j}\theta_0^j \binom{m_i}{j} \left[(1-\theta_0\omega)^{m_i-j}\omega^j\right], \quad i = 0, \ldots, \mu.$$

Assuming now that $\tilde{f}_i(\omega)$ are preprocessed forms of the inexact polynomials $f_i(x)$, the μ deconvolutions of the preprocessed polynomials $\{\tilde{f}_i(\omega)\}$ can be written in matrix form as

$$D^{-1}T\left(\tilde{f}_1(\omega), \tilde{f}_2(\omega), \ldots \tilde{f}_\mu(\omega)\right)\hat{Q}\tilde{\mathbf{h}} \approx \tilde{\mathbf{f}}. \tag{4.18}$$

The block diagonal matrix D^{-1} is already defined in (4.11). Similarly, the block diagonal matrix $\hat{Q}$ is defined in (4.13), and the matrix $T(\tilde{f}_1(\omega), \tilde{f}_2(\omega), \ldots \tilde{f}_\mu(\omega))$ is given by

$$T\left(\tilde{f}_1(\omega), \tilde{f}_2(\omega), \ldots \tilde{f}_\mu(\omega)\right) = \begin{bmatrix} T_{n_1}\left(\tilde{f}_1(\omega)\right) & & & \\ & T_{n_2}\left(\tilde{f}_2(\omega)\right) & & \\ & & \ddots & \\ & & & T_{n_\mu}\left(\tilde{f}_\mu(\omega)\right) \end{bmatrix}.$$

The vectors $\tilde{\mathbf{h}}$ and $\tilde{\mathbf{f}}$ are given by

$$\tilde{\mathbf{h}} = \begin{bmatrix} \tilde{\mathbf{h}}_1, & \tilde{\mathbf{h}}_2, & \ldots, & \tilde{\mathbf{h}}_\mu \end{bmatrix}^T \quad \text{and} \quad \tilde{\mathbf{f}} = \begin{bmatrix} \tilde{\mathbf{f}}_0, & \tilde{\mathbf{f}}_1, & \ldots, & \tilde{\mathbf{f}}_{\mu-1} \end{bmatrix}^T,$$

where $\tilde{\mathbf{f}}_i \in \mathbb{R}^{m_i+1}$ is given by

$$\tilde{\mathbf{f}}_i = \begin{bmatrix} \bar{a}_{i,0}, & \bar{a}_{i,1}\theta_0, & \ldots, & \bar{a}_{i,m}\theta_0^{m_i} \end{bmatrix}^T$$

and $\tilde{\mathbf{h}}_i \in \mathbb{R}^{n_i+1}$ is given by

$$\tilde{\mathbf{h}}_i = \begin{bmatrix} \bar{h}_{i,0}, & \bar{h}_{i,1}\theta_0, & \ldots, & \bar{h}_{i,n_i}\theta_0^{n_i} \end{bmatrix}.$$

The vectors $\tilde{\mathbf{h}}_i$ containing the coefficients of the polynomials $\{\tilde{h}_i(\omega)\}$ are computed by least squares solution of (4.18).

4.2.3 Batch Deconvolution with Structured Total Least Norm (BD-STLN)

The previous section considered the computation of several linked deconvolutions using a structured matrix based method. The polynomials were preprocessed such that the entries in each of the partitions were scaled to minimise the ratio of the entry of maximum magnitude to entry of minimum magnitude. This section extends the previous work by computing structured perturbations of the polynomials $\tilde{f}_i(\omega)$, resulting in the improved approximation of the set of polynomials $\{\hat{h}_i(x)\}$. The described method is somewhat similar to that used in the computation of the low rank approximation of the subresultant matrix in Section 3.5.2.

The coefficient matrix in (4.9), given by $D^{-1}T(\tilde{f}_1(\omega), \ldots, \tilde{f}_\mu(\omega))\hat{Q}$, is of order $M \times N$. It is assumed that the coefficients of the polynomials are inexact, and thus (4.9) does not possess an exact solution. It is therefore necessary to add a structured matrix to the coefficient matrix and a structured vector to the right hand side vector of this equation.

Let $\tilde{q}_i(\omega)$ be the polynomial added to $\tilde{f}_i(\omega)$, and let $\tilde{\mathbf{q}}_i \in \mathbb{R}^{m_i+1}$ be the vector of perturbations added to $\tilde{\mathbf{f}}_i$. Let the vector $\tilde{\mathbf{q}}$ contain all perturbations of each vector $\tilde{\mathbf{f}}_i$

$$\tilde{\mathbf{q}} = \begin{bmatrix} \tilde{\mathbf{q}}_0, & \tilde{\mathbf{q}}_1, & \ldots, & \tilde{\mathbf{q}}_\mu \end{bmatrix}^T,$$

where

$$\begin{aligned}
\tilde{\mathbf{q}}_0 &= \begin{bmatrix} z_0, & z_1\theta, & \ldots, & z_{m_0}\theta^{m_0} \end{bmatrix}^T \in \mathbb{R}^{m_0+1}, \\
\tilde{\mathbf{q}}_1 &= \begin{bmatrix} z_{m_0+1}, & z_{m_0+2}\theta, & \ldots, & z_{m_0+m_1+1}\theta^{m_1} \end{bmatrix}^T \in \mathbb{R}^{m_1+1}, \\
&\vdots \\
\tilde{\mathbf{q}}_\mu &= \begin{bmatrix} z_M, & z_{M+1}\theta, & \ldots, & z_{M_1-1}\theta^{m_\mu} \end{bmatrix}^T \in \mathbb{R}^{m_\mu+1}.
\end{aligned}$$

A matrix of structured perturbations is added to each of the matrices $T_i(\tilde{f}_i(x))$ for $i = 1, \ldots, \mu$, and thus the coefficient matrix in (4.9) is replaced by

$$D^{-1}\left[T\left(\tilde{f}_1(\omega), \ldots, \tilde{f}_\mu(\omega)\right) + T\left(\tilde{q}_1(\omega), \ldots, \tilde{q}_\mu(\omega)\right)\right]\hat{Q},$$

where the matrix $T\left(\tilde{q}_1(\omega), \tilde{q}_2(\omega), \ldots, \tilde{q}_\mu(\omega)\right) \in \mathbb{R}^{M\times N}$ is a block diagonal matrix whose diagonal partitions are of the form $T_{n_i}\left(\tilde{q}_i(\omega)\right) \in \mathbb{R}^{(m_{i-1}+1)\times(n_i+1)}, i = 1, \ldots, \mu$.

Consider now the vector on the right hand side of (4.9), the perturbed form of which is

$$\begin{bmatrix} \tilde{\mathbf{f}}_0 + \tilde{\mathbf{q}}_0 \\ \tilde{\mathbf{f}}_1 + \tilde{\mathbf{q}}_1 \\ \vdots \\ \tilde{\mathbf{f}}_{\mu-2} + \tilde{\mathbf{q}}_{\mu-2} \\ \tilde{\mathbf{f}}_{\mu-1} + \tilde{\mathbf{q}}_{\mu-1} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{f}}_0 \\ \tilde{\mathbf{f}}_1 \\ \vdots \\ \tilde{\mathbf{f}}_{\mu-2} \\ \tilde{\mathbf{f}}_{\mu-1} \end{bmatrix} + \left[\begin{array}{c|c} I_M & 0_{M\times m_\mu} \end{array}\right] \begin{bmatrix} \tilde{\mathbf{q}}_0 \\ \tilde{\mathbf{q}}_1 \\ \vdots \\ \tilde{\mathbf{q}}_{\mu-2} \\ \tilde{\mathbf{q}}_{\mu-1} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{f}}_0 \\ \tilde{\mathbf{f}}_1 \\ \vdots \\ \tilde{\mathbf{f}}_{\mu-2} \\ \tilde{\mathbf{f}}_{\mu-1} \end{bmatrix} + P\tilde{\mathbf{q}},$$

where

$$P = \left[\begin{array}{c|c} I_M & 0_{M\times m_\mu} \end{array}\right] \in \mathbb{R}^{M\times M_1}.$$

It follows that the corrected form of (4.9) is

$$D^{-1}\left(T\left(\tilde{f}_1,\ldots,\tilde{f}_\mu\right)+T\left(\tilde{q}_1,\ldots,\tilde{q}_\mu\right)\right)\hat{Q}\tilde{\mathbf{h}} = \tilde{\mathbf{f}} + P\tilde{\mathbf{q}}, \tag{4.19}$$

where

$$\tilde{\mathbf{h}} = \left[\begin{array}{cccc} \tilde{\mathbf{h}}_1, & \ldots, & \tilde{\mathbf{h}}_{\mu-1}, & \tilde{\mathbf{h}}_\mu \end{array}\right]^T \in \mathbb{R}^N \quad \text{and} \quad \tilde{\mathbf{f}} = \left[\begin{array}{cccc} \tilde{\mathbf{f}}_0, & \ldots, & \tilde{\mathbf{f}}_{\mu-2}, & \tilde{\mathbf{f}}_{\mu-1} \end{array}\right]^T \in \mathbb{R}^M.$$

The residual due to an approximate solution of (4.19) is

$$r = r(\tilde{\mathbf{q}}) = \left(\tilde{\mathbf{f}}+P\tilde{\mathbf{q}}\right) - D^{-1}\left(T\left(\tilde{f}_1(\omega),\ldots,\tilde{f}_\mu(\omega)\right)+T\left(\tilde{q}_1(\omega),\ldots,\tilde{q}_\mu(\omega)\right)\right)\hat{Q}\tilde{\mathbf{h}}, \tag{4.20}$$

and the first order Taylor expansion of $r(\tilde{\mathbf{q}})$ is given by

$$\begin{aligned}\tilde{r}\left(\tilde{\mathbf{q}}+\delta\tilde{\mathbf{q}}\right) &= \left(\mathbf{f}+P\left(\tilde{\mathbf{q}}+\delta\tilde{\mathbf{q}}\right)\right)\\ &\quad - D^{-1}\left(T\left(\tilde{f}_1,\ldots,\tilde{f}_\mu\right)+T\left(\tilde{q}_1+\delta\tilde{q}_1,\ldots,\tilde{q}_\mu+\delta\tilde{q}_\mu\right)\right)\hat{Q}\left(\tilde{\mathbf{h}}+\delta\tilde{\mathbf{h}}\right)\\ &= r\left(\tilde{\mathbf{q}}\right)+P\delta\tilde{\mathbf{q}} - D^{-1}\left(T\left(\tilde{f}_1,\ldots,\tilde{f}_\mu\right)+T\left(\tilde{q}_1,\ldots,\tilde{q}_\mu\right)\right)\hat{Q}\left(\delta\tilde{\mathbf{h}}\right)\\ &\quad - D^{-1}\delta T\left(\tilde{q}_1,\ldots,\tilde{q}_\mu\right)\hat{Q}\tilde{\mathbf{h}}.\end{aligned} \tag{4.21}$$

There exist matrices $Y_i(\tilde{h}_i(\omega)) \in \mathbb{R}^{(m_{i-1}+1)\times(m_i+1)}$ for $i = 1,\ldots,\mu$, such that

$$D_{m_{i-1}}^{-1}T_{n_i}\left(\tilde{q}_i(\omega)\right)Q_{n_i}\tilde{\mathbf{h}}_i = D_{m_{i-1}}^{-1}Y_{m_i}\left(\tilde{h}_i(\omega)\right)Q_{m_i}\tilde{\mathbf{q}}_i$$

and thus

$$D_{m_{i-1}}^{-1}\delta T_{n_i}\left(\tilde{q}_i(\omega)\right)Q_{n_i}\tilde{\mathbf{h}}_i = D_{i-1}^{-1}Y_{m_i}\left(\tilde{h}_i(\omega)\right)Q_{m_i}\delta\tilde{\mathbf{q}}_i,$$

from which it follows that

$$\begin{aligned}&D^{-1}\delta T\left(\tilde{q}_1,\ldots,\tilde{q}_\mu\right)\hat{Q}\tilde{\mathbf{h}} =\\ &\left[\begin{array}{cccc} D_{m_0}^{-1}Y_{m_1}(\tilde{h}_1(\omega))Q_{m_1} & & & \\ & \ddots & & \\ & & D_{m_{\mu-2}}^{-1}Y_{m_{\mu-1}}(\tilde{h}_{\mu-1}(\omega))Q_{m_{\mu-1}} & \\ & & & D_{m_{\mu-1}}^{-1}Y_{m_\mu}(\tilde{h}_\mu(\omega))Q_{m_\mu} \end{array}\right]\left[\begin{array}{c} \delta\tilde{\mathbf{q}}_1\\ \delta\tilde{\mathbf{q}}_2\\ \vdots\\ \delta\tilde{\mathbf{q}}_{\mu-1}\\ \delta\tilde{\mathbf{q}}_\mu \end{array}\right]\\ &= \left[\begin{array}{c|ccc} 0 & D_{m_0}^{-1}Y_{m_1}(\tilde{h}_1(\omega))Q_{m_1} & & \\ 0 & & & \\ \vdots & & \ddots & \\ 0 & & & D_{m_{\mu-2}}^{-1}Y_{m_{\mu-1}}(\tilde{h}_{\mu-1}(\omega))Q_{m_{\mu-1}} \\ 0 & & & \end{array}\right.\\ &\qquad\left.\begin{array}{c} \\ \\ \\ \\ D_{m_{\mu-1}}^{-1}Y_{m_\mu}(\tilde{h}_\mu(\omega))Q_{m_\mu} \end{array}\right]\delta\tilde{\mathbf{q}}\\ &= D^{-1}Y\left(\tilde{h}_1,\ldots,\tilde{h}_\mu\right)\hat{Q}_z\delta\tilde{\mathbf{q}},\end{aligned} \tag{4.22}$$

and an element of a partition of this matrix, $D_{m_{k-1}}^{-1} Y_{m_k}(\tilde{h}_k(\omega)) Q_{m_k}$, has the form

$$\left(D_{m_{k-1}}^{-1} Y_{m_k}\left(\tilde{h}_k(\omega)\right) Q_{m_k}\right)_{(i+j+1,j+1)} = \begin{cases} \tilde{h}_{k,i}\theta_0^i \frac{\binom{n_i}{i}\binom{m_i}{j}}{\binom{m_i-1}{i+j}} & i = 0,\ldots,n_i;\ j = 0,\ldots,m_i; \\ 0 & \text{otherwise.} \end{cases}$$

The substitution of (4.22) into (4.21) yields

$$r\left(\tilde{\mathbf{q}} + \delta\tilde{\mathbf{q}}\right) = r\left(\tilde{\mathbf{q}}\right) - D^{-1}\left(T\left(\tilde{f}_1,\ldots,\tilde{f}_\mu\right) + T\left(\tilde{q}_1,\ldots,\tilde{q}_\mu\right)\right)\hat{Q}\delta\tilde{\mathbf{h}} - \left(D^{-1}Y\left(\tilde{h}_1,\ldots,\tilde{h}_\mu\right)\hat{Q}_z - P\right)\delta\tilde{\mathbf{q}},$$

and thus the Newton-Raphson method requires the iterative solution of

$$\left[\ D^{-1}\left(T(\tilde{f}_1,\ldots,\tilde{f}_\mu) + T(\tilde{q}_1,\ldots,\tilde{q}_\mu)\right)\hat{Q} \quad \left(D^{-1}Y(\tilde{h}_1,\ldots,\tilde{h}_\mu)\hat{Q}_z - P\right)\ \right]\begin{bmatrix} \delta\tilde{\mathbf{h}} \\ \delta\tilde{\mathbf{q}} \end{bmatrix} = r(\tilde{\mathbf{q}}),$$

which is an under-determined equation, and the coefficient matrix is given by

$$\left[\ D^{-1}\left(T(\tilde{f}_1,\ldots,\tilde{f}_\mu) + T(\tilde{q}_1,\ldots,\tilde{q}_\mu)\right)\hat{Q} \quad \left(D^{-1}Y(\tilde{h}_1,\ldots,\tilde{h}_\mu)\hat{Q}_z - P\right)\ \right] \in \mathbb{R}^{M\times(N+M_1)}.$$

If $\tilde{\mathbf{h}}^{(0)}$ and $\tilde{\mathbf{q}}^{(0)} = 0$ are the initial values of $\tilde{\mathbf{h}}$ and $\tilde{\mathbf{q}}$ respectively in the Newton-Raphson method, then the $(j+1)$th iteration requires the minimisation of

$$\left\| \begin{matrix} \tilde{\mathbf{h}}^{(j+1)} - \tilde{\mathbf{h}}^{(0)} \\ \tilde{\mathbf{q}}^{(j+1)} \end{matrix} \right\| = \left\| \begin{matrix} \tilde{\mathbf{h}}^{(j)} + \delta\tilde{\mathbf{h}}^{(j)} - \tilde{\mathbf{h}}^{(0)} \\ \tilde{\mathbf{q}}^{(j)} + \delta\tilde{\mathbf{q}}^{(j)} \end{matrix} \right\| = \left\| \begin{bmatrix} \delta\tilde{\mathbf{h}}^{(j)} \\ \delta\tilde{\mathbf{q}}^{(j)} \end{bmatrix} - \begin{bmatrix} -(\tilde{\mathbf{h}}^{(j)} - \tilde{\mathbf{h}}^{(0)}) \\ -\tilde{\mathbf{q}}^{(j)} \end{bmatrix} \right\|$$

subject to

$$\left[\ D^{-1}\left(T\left(\tilde{f}_1,\ldots,\tilde{f}_\mu\right) + T\left(\tilde{q}_1,\ldots,\tilde{q}_\mu\right)\right)\hat{Q} \quad D^{-1}Y\left(\tilde{h}_1,\ldots,\tilde{h}_\mu\right)\hat{Q}_z - P\ \right]^{(j)}\begin{bmatrix} \delta\tilde{\mathbf{h}} \\ \delta\tilde{\mathbf{q}} \end{bmatrix}^{(j)} = r^{(j)},$$

where the initial value of $\tilde{\mathbf{h}}$ is calculated from (4.9)

$$\tilde{\mathbf{h}}^{(0)} = \left(D^{-1}T(\tilde{f}_1(\omega), \tilde{f}_2(\omega),\ldots,\tilde{f}_\mu(\omega))\hat{Q}\right)^\dagger \tilde{\mathbf{f}}.$$

This gives rise to an LSE problem

$$\min_y \|Fy - s\| \quad \text{subject to} \quad Gy = t,$$

where

$$F = I_{N+M_1}$$

$$G = \left[\ D^{-1}\left(T\left(\tilde{f}_1,\ldots,\tilde{f}_\mu\right) + T\left(\tilde{q}_1,\ldots,\tilde{q}_\mu\right)\right)\hat{Q} \quad D^{-1}Y\left(\tilde{h}_1,\ldots,\tilde{h}_\mu\right)\hat{Q}_z - P\ \right]^j \in \mathbb{R}^{M\times(N+M_1)}$$

$$y = \begin{bmatrix} \delta\tilde{\mathbf{h}}^{(j)} \\ \delta\tilde{\mathbf{q}}^{(j)} \end{bmatrix} \in \mathbb{R}^{N+M_1}$$

$$s = \begin{bmatrix} \mathbf{h}^{(0)} - \mathbf{h}^{(j)} \\ -\tilde{\mathbf{q}}^{(j)} \end{bmatrix} = \begin{bmatrix} \mathbf{h}_0 \\ \mathbf{q}_0 \end{bmatrix} - \begin{bmatrix} \tilde{\mathbf{h}} \\ \tilde{\mathbf{q}} \end{bmatrix}^{(j)} \in \mathbb{R}^{N+M_1}$$

and $t = r^{(j)} \in \mathbb{R}^M$.

4.2.4 Constrained Batch Deconvolution (CBD)

The previous two sections have considered the batch deconvolution (BD) method and the extension to the batch deconvolution with STLN (BDSTLN) method. In the batch deconvolution with STLN (BDSTLN) method, structured perturbations are computed such that the residual associated with (4.19) is minimised. In this section further constraints are applied to the problem, and these are derived from the multiplicity structure of the factors of $\hat{f}_0(x)$. Given the multiplicity structure, certain subsets of the polynomials $\{\hat{h}_i(x) = \hat{f}_{i+1}(x)/\hat{f}_i(x)\}$ are equal, and this constraint can be added to the structure of the deconvolution problem.

For the development of the theory in this section, it is assumed that the polynomials to be deconvolved, $\hat{f}_i(x)$, are in the exact form. The factorisation of the exact polynomial $\hat{f}(x)$ is given by

$$\hat{f}_0(x) = w_1(x)w_2(x)^2 \dots w_\mu(x)^\mu,$$

where $\{w_i(x)\}$ are square-free polynomials of multiplicity i. Let $\{w_{k_i}(x)\}$ be the subset of $\{w_i(x)\}$, such that each w_{k_i} in a non-constant polynomial. The set is given by

$$\{w_{k_i}\} = \{\, w_i(x) \mid w_i(x) \neq 1 \,\},$$

where $k_1, \dots, k_t$ are integers of increasing size. The polynomial $\hat{f}(x)$ has at least one factor of multiplicity k_i for $i = 1, \dots, t$, where t denotes the size of the set of w_{k_i}.

Example 4.2.1. Consider the polynomial $\hat{f}(x)$ given by

$$\hat{f}(x) = (x - 0.2)^2(x - 0.3)^5,$$

then

$$w_1 = 1, \quad w_2 = (x - 0.2), \quad w_3 = 1, \quad w_4 = 1 \quad \text{and} \quad w_5 = (x - 0.3)$$

so $k_1 = 2$ and $k_2 = 5$. □

The set of polynomials $\{h_i(x)\}$ can therefore be split into the subsets $\{h_0, h_1, \dots, h_{k_1}\}$, $\{h_{k_1+1}, \dots, h_{k_2}\}$, $\dots$, $\{h_{k_{t-1}+1,\dots,h_{k_t}}\}$, where the polynomials contained within a subset are all identical. Let the set of polynomials $\{\, \hat{p}_j \mid j = 1, \dots, t \,\}$ be given by

$$\hat{p}_j(x) = \hat{h}_{k_{j-1}}(x) = \hat{h}_{k_{j-1}+1}(x) = \dots = \hat{h}_{k_j}(x),$$

where the degree of $p_j(x)$ is given by o_j. The polynomials $\{\, \hat{p}_i(x) \mid i = 1, \dots, t \,\}$ form the solution vector of the system

$$C\left(\hat{f}_1(x), \dots, \hat{f}_{k_t}(x)\right)\hat{\mathbf{p}} = \hat{\mathbf{f}}, \tag{4.23}$$

where the matrix $C(\hat{f}_1(x), \ldots, \hat{f}_{k_t}(x))$ is given by

$$\begin{bmatrix}
D_{m_0}^{-1} T_{o_1}\left(\hat{f}_1(x)\right) Q_{o_1} & & & & & \\
\vdots & & & & & \\
D_{m_{k_1-1}}^{-1} T_{o_1}\left(\hat{f}_{k_1}(x)\right) Q_{o_1} & & & & & \\
& D_{m_{k_1}}^{-1} T_{o_2}\left(\hat{f}_{k_1+1}(x)\right) Q_{o_2} & & & & \\
& \vdots & & & & \\
& D_{m_{k_2-1}}^{-1} T_{o_2}\left(\hat{f}_{k_2}(x)\right) Q_{o_2} & & & & \\
& & \ddots & & & \\
& & \ddots & & & \\
& & \ddots & & & \\
& & & D_{m_{k_{t-1}-1}}^{-1} T_{o_t}\left(\hat{f}_{k_{t-1}+1}(x)\right) Q_{o_t} \\
& & & \vdots \\
& & & D_{m_{k_t-1}}^{-1} T_{o_t}\left(\hat{f}_{k_t}(x)\right) Q_{o_t}
\end{bmatrix}.$$

The vector $\hat{\mathbf{p}}$ is given by

$$\hat{\mathbf{p}} = \begin{bmatrix} \hat{\mathbf{p}}_1, & \hat{\mathbf{p}}_2, & \ldots, & \hat{\mathbf{p}}_t \end{bmatrix}^T$$

and the vector $\hat{\mathbf{f}}$ is given by

$$\hat{\mathbf{f}} = \begin{bmatrix} \hat{\mathbf{f}}_0, & \ldots, & \hat{\mathbf{f}}_{k_1-1}, \Big| \hat{\mathbf{f}}_{k_1}, & \ldots, & \hat{\mathbf{f}}_{k_2-1}, \Big| \hat{\mathbf{f}}_{k_2}, & \ldots, & \hat{\mathbf{f}}_{k_3-1}, \Big| \ldots, & \hat{\mathbf{f}}_{k_{t-2}}, \Big| \ldots, & \hat{\mathbf{f}}_{k_{t-1}} \end{bmatrix}^T.$$

Assuming now that polynomials $\{f_i(x)\}$ are defined inexactly, the approximations $p_i(x)$ are computed from the least squares solution of

$$C(f_1(x), f_2(x), \ldots, f_{k_t}(x))\mathbf{p} \approx \mathbf{f}. \tag{4.24}$$

Example 4.2.2. Consider the Bernstein form of the exact polynomial $\hat{f}(x)$, whose factorisation is given by

$$\hat{f}(x) = (x - 0.3)^7 (x - 0.2)^3.$$

The polynomials $\{\, \hat{f}_i(x) \mid i = 0, \ldots, 7 \,\}$ are given in factorised form by

$$\hat{f}_i(x) = \begin{cases} (x - 0.3)^{7-i}(x - 0.2)^{3-i} & i = 0, \ldots, 3, \\ (x - 0.3)^{7-i} & i = 4 \ldots 7 \end{cases}$$

and the polynomials $\{\, \hat{h}_i(x) \mid i = 1, \ldots, 7 \,\}$ are given by

$$\hat{h}_i(x) = \frac{\hat{f}_{i-1}(x)}{\hat{f}_i(x)},$$

where the polynomials $\{\hat{h}_1(x), \ldots, \hat{h}_3(x)\}$ are equal, as are the polynomials $\{\hat{h}_4(x), \ldots, \hat{h}_7(x)\}$. Therefore, the set of polynomials $\{\, \hat{h}_i(x) \mid i = 1, \ldots, 7 \,\}$ can be

computed by defining $\hat{p}_1(x)$ and $\hat{p}_2(x)$ as

$$\hat{p}_1(x) = \hat{h}_1(x) = \cdots = \hat{h}_3(x) \quad \text{and} \quad \hat{p}_2(x) = \hat{h}_4(x) = \cdots = \hat{h}_7(x)$$

and constructing the matrix-vector product

$$\begin{bmatrix} C(\hat{f}_1(x)) & & \\ C(\hat{f}_2(x)) & & \\ C(\hat{f}_3(x)) & & \\ & C(\hat{f}_4(x)) & \\ & C(\hat{f}_5(x)) & \\ & C(\hat{f}_6(x)) & \\ & C(\hat{f}_7(x)) & \end{bmatrix} \begin{bmatrix} \hat{\mathbf{p}}_1 \\ \hat{\mathbf{p}}_2 \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{f}}_0 \\ \hat{\mathbf{f}}_1 \\ \hat{\mathbf{f}}_2 \\ \hat{\mathbf{f}}_3 \\ \hat{\mathbf{f}}_4 \\ \hat{\mathbf{f}}_5 \\ \hat{\mathbf{f}}_6 \end{bmatrix},$$

then solving to find vectors $\hat{\mathbf{p}}_1$ and $\hat{\mathbf{p}}_2$. □

As with the batch deconvolution method, preprocessing the polynomials can be considered, and typically best results are obtained by the inclusion of preprocessing. Since the matrix partitions in the batch constrained problem are identical to the partitions found in the unconstrained problem, the linear programming problem is unaltered.

4.2.5 Constrained Batch Deconvolution with STLN (CBDSTLN)

In the constrained batch deconvolution with STLN (CBDSTLN) method, the low rank approximation of the matrix in (4.24) is computed. Minimal perturbations are added to the polynomials $\{f_i\}$ much in the same way as in BDSTLN but with the added constraints introduced by the CBD method. Given the inexactly defined set of polynomials $\{f_i(x)\}$, perturbations $\{q_i(x)\}$ are computed such that

$$C(f_1, f_2, \ldots, f_\mu) + B(q_1, q_2, \ldots, q_\mu)\mathbf{p} = \mathbf{f} + \mathbf{q} \tag{4.25}$$

and the vector $\mathbf{p}$ contains the coefficients of the set of polynomials $\{p_i(x)\}$.

4.2.6 Results

Three distinct approaches to the deconvolution problem have been considered: (i) separate deconvolution (SD), (ii) batch deconvolution (BD) and (iii) constrained batch deconvolution (CBD). Both the batch deconvolution (BD) and constrained batch deconvolution (CBD) methods may include the computation of structured perturbations. The following examples will consider the computation of approximations of the polynomials $\{\hat{h}_i(x, y)\}$ given the inexact polynomials $\{\hat{f}_i(x, y)\}$ for each of the described methods.

In each of the following examples a polynomial $\hat{f}(x)$ is defined in terms of its factors and each of the subsequent polynomials $\hat{f}_i(x)$ for $i = 1, \ldots, \mu$ are given such that each $\hat{f}_{i+1}(x)$ is the GCD of $\hat{f}_i(x)$ and its derivative $\hat{f}_i'(x)$. Noise is added to the coefficients of

each $\hat{f}_k(x)$ and the perturbed polynomials $f_k(x)$ have coefficients

$$f_k(x) = \sum_{i=0}^{m_k} a_{k,i} \binom{m}{i} (1-x)^{m_k - i} x^i \qquad \text{where} \quad a_{k,i} = \hat{a}_{k,i} + \delta \hat{a}_{k,i}.$$

The noise introduced in these examples does not necessarily give an accurate representation of the noise in the polynomials $\{ f_i(x) \mid i = 0, \ldots, \mu \}$ computed by the square-free factorisation algorithm (Algorithm 1). In fact, it would reasonably be anticipated that each polynomial $f_i(x)$ would have a larger error than $f_{i-1}(x)$. That is, $f_j(x)$ has more error in its coefficients than $f_i(x)$ for $j > i$. This is due to the iterative nature of the square-free factorisation algorithm. However, for simplicity, these examples assume noise is random and defined within a given interval.

Approximations of the polynomials $\{ \hat{h}_i(x) = \frac{\hat{f}_{i-1}(x)}{\hat{f}_i(x)} \mid i = 1, \ldots, \mu \}$ are computed by the five methods (i) separate deconvolution (SD), (ii) batch deconvolution (BD), (iii) batch deconvolution with STLN (BDSTLN), (iv) constrained batch deconvolution (CBD) and (v) constrained batch deconvolution with STLN (CBDSTLN), which were described in Section 4.2.

For each method, two variations are considered :

1. The inexact polynomials $\{f_i(x)\}$ are deconvolved to obtain the set of approximations $\{h_i(x)\}$. The error between the approximated polynomials $\{h_i(x)\}$ and the exact polynomials $\{\hat{h}_i(x)\}$ is given by

$$\epsilon_i\left(h_i\right) = \frac{\left\| \hat{\mathbf{h}}_i - \mathbf{h}_i \right\|}{\left\| \hat{\mathbf{h}}_i \right\|} \quad \text{for} \quad i = 1, \ldots, \mu, \tag{4.26}$$

where $\hat{\mathbf{h}}_i \in \mathbb{R}^{n_i+1}$ is the vector of coefficients of the exact polynomial $\hat{h}_i(x)$ and $\mathbf{h}_i \in \mathbb{R}^{n_i+1}$ is the vector of coefficients of the computed polynomial $h_i(x)$.

2. Preprocessing the polynomials $\{f_i(x)\}$ gives the set of preprocessed polynomials $\{ \tilde{f}_i(\omega) \mid i = 0, \ldots, \mu \}$ which are deconvolved to obtain approximations $\{ \tilde{h}_i(\omega) \mid i = 1, \ldots, \mu \}$. The errors between the set of exact polynomials $\{\hat{h}_i(x)\}$ and the set of corresponding approximations $\{ \tilde{h}_i(x) \}$ are given by

$$\tilde{\epsilon}_i(\tilde{h}_i) = \frac{\left\| \hat{\mathbf{h}}_i - \tilde{\mathbf{h}}_i \right\|}{\left\| \hat{\mathbf{h}}_i \right\|} \quad \text{for} \quad i = 1, \ldots, \mu, \tag{4.27}$$

where $\tilde{\mathbf{h}}_i$ is a vector of the coefficients of $\tilde{h}_i(x)$ and the polynomials $\{ \tilde{h}_i(x) \}$ are given by a change in the independent variable where $\omega = \frac{x}{\theta}$. That is, the polynomials $\{\tilde{h}_i(x)\}$ are given by

$$\begin{aligned} \tilde{h}_i(x) &= \sum_{j=0}^{n_i} h_{i,j} \theta^j \binom{n_i}{j} (1 - \theta\omega)^{n_i - j} \omega^j \times \left(\frac{x^j}{\theta^j} \right) \\ &= \sum_{j=0}^{n_i} h_{i,j} \binom{n_i}{j} (1-x)^{n_i - j} x^j \quad \text{for} \quad i = 1, \ldots, \mu. \end{aligned}$$

The values ϵ_{avg} and $\tilde{\epsilon}_{avg}$ are defined as the averages of the sets $\{\,\epsilon_i \mid i = 1, \ldots, \mu\,\}$ and $\{\,\tilde{\epsilon}_i \mid i = 1, \ldots, \mu\,\}$, where ϵ_i and $\tilde{\epsilon}_i$ are defined in (4.26) and (4.27) respectively. By the examples included in this section, it will be shown that $\tilde{\epsilon}_{avg}$ is typically smaller than ϵ_{avg}.

Example 4.2.3. Consider the Bernstein form of the exact polynomial $\hat{f}_0(x)$, whose factorisation is given by

$$\hat{f}_0(x) = (x - 0.56897)^3(x + 0.56921)^9(x + 1.24672)^6.$$

The polynomials $\{\hat{f}_1(x), \ldots, \hat{f}_9(x)\}$ are computed by a sequence of GCD computations such that $\hat{f}_{i+1}(x) = \text{GCD}(\hat{f}_i(x), \hat{f}_i'(x))$, and each $\hat{f}_i(x)$ is given by

$$\hat{f}_i(x) = \begin{cases} (x - 0.56897)^{(3-i)}(x + 0.56921)^{(9-i)}(x + 1.24672)^{(6-i)} & i = 0, 1, 2, \\ (x + 0.56921)^{(9-i)}(x + 1.24672)^{(6-i)} & i = 3, 4, 5, \\ (x + 0.56921)^{(9-i)} & i = 6, 7, 8, \\ 1 & i = 9. \end{cases}$$

Random noise is added to the coefficients of each $\hat{f}_i(x)$, such that the coefficients of the inexact polynomials $\{\,f_i(x) \mid i = 0, \ldots, 9\,\}$ are given by

$$a_{i,j} = \hat{a}_{i,j} + r_{i,j}\hat{a}_{i,j}\epsilon_{i,j}, \quad \text{for} \quad i = 1, \ldots, \mu; \quad j = 0, \ldots, m_i, \tag{4.28}$$

where r_j are uniformly distributed random variables in the interval $[-1, 1]$ and ϵ_j are uniformly distributed random variables in the interval $\left[10^{-10}, 10^{-8}\right]$.

The five deconvolution methods are considered for (i) the set of unprocessed inexact polynomials $\{f_i(x)\}$ and (ii) preprocessed inexact polynomials $\{\tilde{f}_i(x)\}$, and the errors $\{\epsilon_i\}$ and $\{\tilde{\epsilon}_i\}$ for $i = 1, \ldots, 9$ are plotted in Figures 4.7i and 4.7ii respectively.

The average errors ϵ_{avg} and $\tilde{\epsilon}_{avg}$ for each of the five methods are given in Table 4.1.

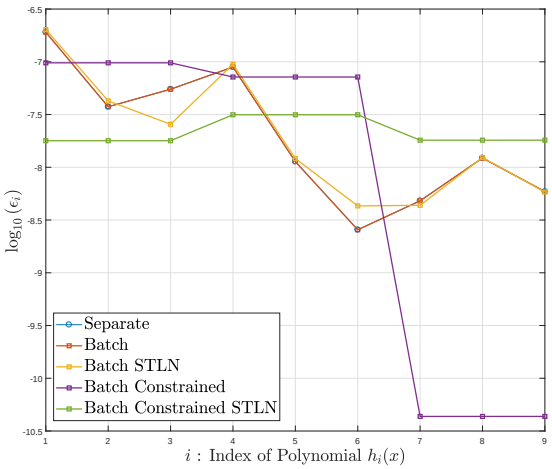
Method	Unprocessed ϵ_{avg}	Preprocessed $\tilde{\epsilon}_{avg}$
Separate	$4.545356e-08$	$4.545356e-08$
Batch	$4.545355e-08$	$8.423505e-09$
Batch with STLN	$4.490187e-08$	$6.783958e-09$
Batch Constrained	$5.651231e-08$	$6.708721e-09$
Batch Constrained with STLN	$2.246971e-08$	$3.199323e-09$

Table 4.1: Error in the approximations of the polynomials $\{\hat{h}_i(x)\}$ in Example 4.2.3

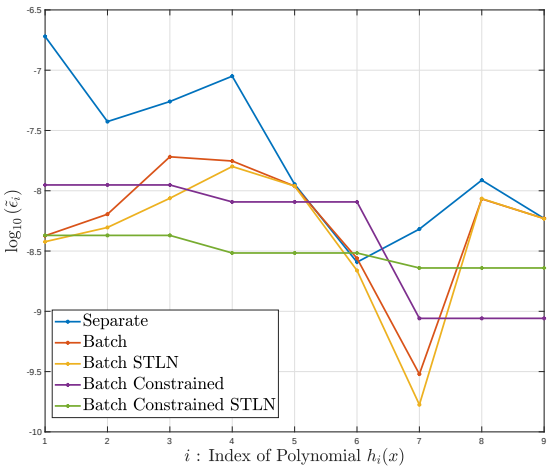
□

Example 4.2.4. Consider the Bernstein form of the exact polynomial $\hat{f}(x)$, whose factorised form is given by

$$\hat{f}_0(x) = (x - 6.5432)^7(x - 2.1234565487)(x - 1.589212457)^4(x - 0.7213)^{10}(x + 0.72)^{20}.$$



(i) Errors $\{\epsilon_i\}$ in the approximations of $\{\hat{h}_i(x)\}$ by five deconvolution methods



(ii) Errors $\{\tilde{\epsilon}_i\}$ in the approximations of $\{\hat{h}_i(x)\}$ by five deconvolution methods

Figure 4.7: Error in the approximations of $\{\hat{h}_i(x)\}$ by five deconvolution methods (i) excluding and (ii) including preprocessing in Example 4.2.3

The polynomials $\{\,\hat{f}_i(x) \mid i = 1, \ldots, 20\}$ are given by

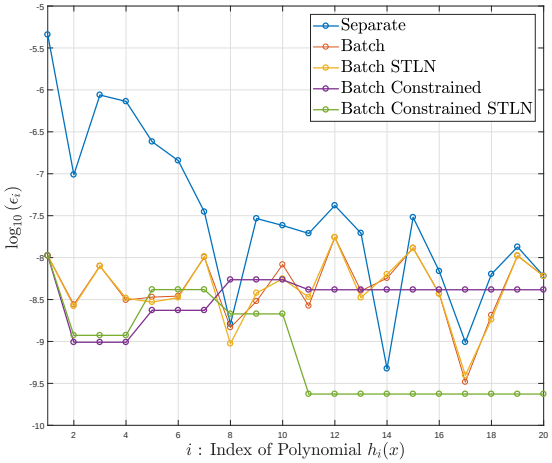
$$\hat{f}_i(x) = \begin{cases} (x-6.5432)^{7-i}(x-1.589212457)^{4-i}(x-0.7213)^{10-i}(x+0.72)^{20-i} & i = 1,2,3, \\ (x-6.5432)^{7-i}(x-0.7213)^{10-i}(x+0.72)^{20-i} & i = 4,5,6, \\ (x-0.7213)^{10-i}(x+0.72)^{20-i} & i = 7,8,9, \\ (x+0.72)^{20-i} & i = 10,\ldots,19, \\ 1 & i = 20. \end{cases}$$

Let the coefficients of each $\hat{f}_i(x)$ be denoted $\{\,\hat{a}_{i,j} \mid j = 0, \ldots, m_{20}\}$. Noise is added to the coefficients of the polynomials $\{\hat{f}_i(x)\}$ such that the coefficients of the inexact polynomials $\{f_i(x)\}$ are given by (4.28), where $\{r_{i,j}\}$ are uniformly distributed random variables in the interval $[-1, 1]$ and $\{\epsilon_{i,j}\} = 10^{-8}$.

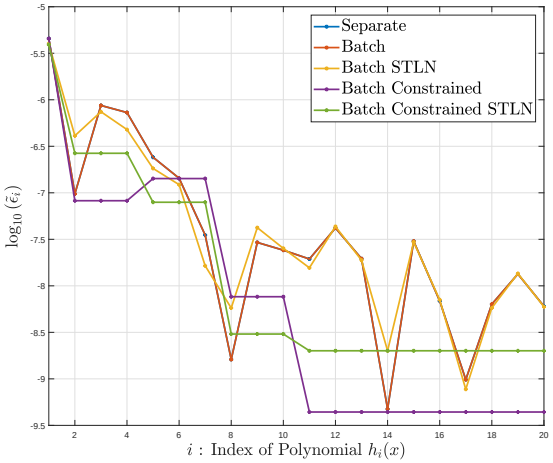
The sets of polynomials $\{\,h_i(x) \mid i = 1, \ldots, 20\}$ and $\{\,\tilde{h}_i(x) \mid i = 1, \ldots, 20\,\}$ are approximations of $\{\hat{h}_i(x)\}$ and the distances between the exact polynomials $\{\hat{h}_i(x)\}$ and the approximations are given by $\{\epsilon_i\}$ and $\{\tilde{\epsilon}_i\}$ respectively, which are plotted in Figure 4.8i and Figure 4.8ii. The average errors denoted ϵ and $\tilde{\epsilon}$ of the sets $\{\epsilon_i\}$ and $\{\tilde{\epsilon}_i\}$ for each deconvolution method are given in Table 4.2.

Method	Unprocessed ϵ_{avg}	Preprocessed $\tilde{\epsilon}_{avg}$
Separate	$3.431876e-07$	$3.431876e-07$
Batch	$3.431876e-07$	$5.974866e-09$
Batch with STLN	$3.090918e-07$	$5.856129e-09$
Batch Constrained	$2.624582e-07$	$3.912796e-09$
Batch Constrained with STLN	$2.490096e-07$	$1.767510e-09$

Table 4.2: Error in the approximations of the polynomials $\{\hat{h}_i(x)\}$ in Example 4.2.4



(i) Error $\{\epsilon_i\}$ in the approximations of $\{\hat{h}_i(x)\}$ by five deconvolution methods



(ii) Error $\{\tilde{\epsilon}_i\}$ in the approximations of $\{\hat{h}_i(x)\}$ by five deconvolution methods

Figure 4.8: Error in the approximations of $\{\hat{h}_i(x)\}$ computed using five deconvolution methods (i) excluding and (ii) including preprocessing in Example 4.2.4

□

Example 4.2.5. Consider the Bernstein form of the exact polynomial $\hat{f}(x)$, whose factorisation is given by

$$\hat{f}(x) = (x - 2.16547697898)^2(x - 1.589)^{12}(x + 0.2789)^{30}.$$

By the square-free factorisation algorithm (Algorithm 1), the exact polynomials $\{\, \hat{f}_i(x) \mid i = 0, \ldots, 30 \,\}$ are given by

$$\hat{f}_i(x) = \begin{cases} (x - 2.16547697898)^{2-i}(x - 1.589)^{12-i}(x + 0.2789)^{30-i} & i = 0, 1, \\ (x - 1.589)^{12-i}(x + 0.2789)^{30-i} & i = 2, \ldots, 11, \\ (x + 0.2789)^{30-i} & i = 13, \ldots, 29, \\ 1 & i = 30 \end{cases}$$

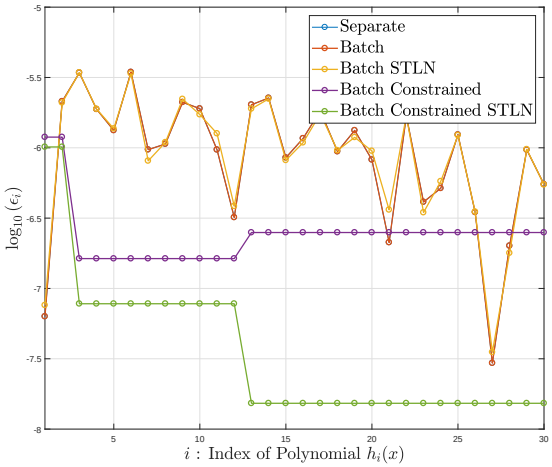
and the exact polynomials $\hat{h}_i(x)$ for $i = 1, \ldots, 30$ are given by

$$\hat{h}_i(x) = \begin{cases} (x - 2.16547697898)(x - 1.589)(x + 0.2789) & i = 1, 2, \\ (x - 1.589)(x + 0.2789) & i = 3, \ldots, 12, \\ (x + 0.2789) & i = 13, \ldots, 30. \end{cases}$$

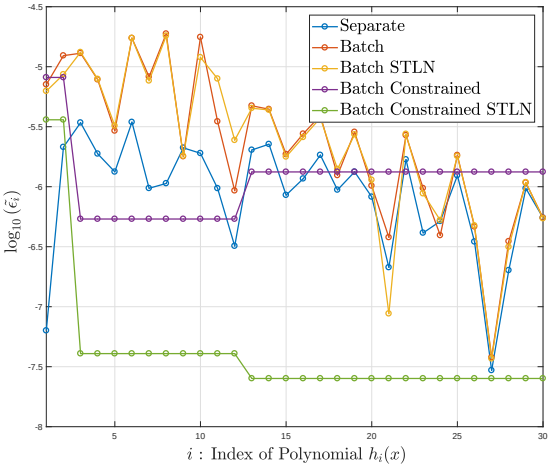
This example proceeds in the same way as Example 4.2.4, and the coefficients of each inexact polynomial $f_i(x)$ are given by (4.28), where $\epsilon_{i,j}$ in this case is set equal to 10^{-6}. The distance between the two sets of approximations $\{h_i(x)\}$ and $\{\tilde{h}_i(x)\}$ and the exact polynomials $\{\hat{h}_i(x)\}$ is measured and plotted in Figure 4.9. The average errors in these two sets of approximations are denoted ϵ_{avg} and $\tilde{\epsilon}_{avg}$ respectively, and are given for each method in Table 4.3.

Method	Unprocessed ϵ_{avg}	Preprocessed $\tilde{\epsilon}_{avg}$
Separate	$1.233120e-06$	$9.979658e-07$
Batch	$1.233120e-06$	$4.675257e-06$
Batch with STLN	$1.228831e-06$	$4.469532e-06$
Batch Constrained	$2.836729e-07$	$1.311885e-06$
Batch Constrained with STLN	$1.027127e-07$	$2.867082e-07$

Table 4.3: Error in the approximations of the polynomials $\{\hat{h}_i(x)\}$ in Example 4.2.5



(i) Error $\{\epsilon_i\}$ in the approximations of $\{\hat{h}_i(x)\}$ computed using five deconvolution methods



(ii) Error $\{\tilde{\epsilon}_i\}$ in the approximations of $\{\hat{h}_i(x)\}$ computed using five deconvolution methods

Figure 4.9: Error in the approximations of $\{\hat{h}_i(x)\}$ computed using five deconvolution methods (i) excluding and (ii) including preprocessing in Example 4.2.5

□

Example 4.2.6. Consider the Bernstein form of the exact polynomial $\hat{f}_0(x)$, whose factorised form is given by

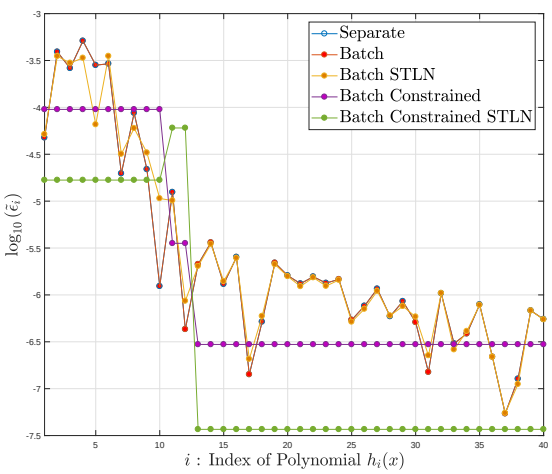
$$\hat{f}_0(x) = (x - 3.654132475632154)^{10}(x - 1.589)^{12}(x + 0.278912456789)^{40}.$$

The polynomials $\{\, \hat{f}_i(x) \mid i = 1, \ldots, 40 \,\}$ are given by

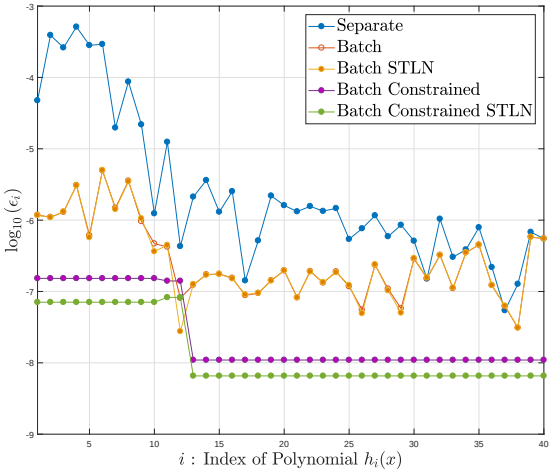
$$\hat{f}_i(x) = \begin{cases} (x - 3.654132475632154)^{10-i}(x - 1.589)^{12-i}(x + 0.278912456789)^{40-i} & i = 1, \ldots, 9, \\ (x - 1.589)^{12-i}(x + 0.278912456789)^{40-i} & i = 10, 11, \\ (x + 0.278912456789)^{40-i} & i = 12, \ldots, 39, \\ 1 & i = 40. \end{cases}$$

Noise is added to the coefficients of each of the polynomials in the set $\{\, \hat{f}_i \mid i = 1, \ldots, 40 \,\}$, where the coefficients of the inexact polynomials are given by (4.28), where $\{r_{i,j}\}$ are uniformly distributed random variables in the interval $[-1, 1]$ and $\{\epsilon_{i,j}\} = 10^{-6}$.

The approximations $\{h_i(x)\}$ are computed from the unprocessed inexact polynomials $\{f_i(x)\}$ using five deconvolution methods. The error in the approximation of each polyno-



(i) Errors $\{\epsilon_i\}$ in the approximations of $\{\hat{h}_i(x)\}$ computed using five deconvolution methods



(ii) Errors $\{\tilde{\epsilon}_i\}$ in approximations of $\{\hat{h}_i(x)\}$ computed using five deconvolution methods

Figure 4.10: Error in the approximations of $\{\hat{h}_i(x)\}$ computed using five deconvolution methods (i) excluding and (ii) including preprocessing in Example 4.2.6

mial $\hat{h}_i$, given by ϵ_i, is plotted in Figure 4.10i. The polynomials $\{\tilde{h}_i(x)\}$ are computed by the deconvolution of the preprocessed inexact polynomials $\{\tilde{f}_i(\omega)\}$ and the errors in these approximations, given by $\tilde{\epsilon}_i$, are plotted in Figure 4.10ii. The average errors, denoted ϵ and $\tilde{\epsilon}$, are given in Table 4.4.

Method	Unprocessed ϵ_{avg}	Preprocessed $\tilde{\epsilon}_{avg}$
Separate	$1.071075e-04$	$4.889416e-05$
Batch	$1.071075e-04$	$6.134607e-07$
Batch with STLN	$8.844553e-05$	$6.082674e-07$
Batch Constrained	$5.383393e-05$	$5.283715e-08$
Batch Constrained with STLN	$1.116176e-05$	$2.641052e-08$

Table 4.4: Error in the approximations of the polynomials $\{\hat{h}_i(x)\}$ in Example 4.2.6

□

This section has described several methods of polynomial deconvolution using structured matrix methods, given the synthetically inexact polynomials $\{f_i(x)\}$. The examples in this section have considered five methods (i) separate deconvolution (SD), (ii) batch deconvolution (BD), (iii) batch deconvolution with STLN (BDSTLN), (iv) constrained batch deconvolution (CBD) and (v) constrained batch deconvolution with STLN (CBD-STLN) for division of the polynomials $\{f_i(x) \mid i = 0, \ldots, \mu\}$. It has been shown that first preprocessing the polynomials, such that the block partitions of the coefficient matrix are relatively scaled, yields improved results in the approximation of $\{\hat{h}_i(x)\}$.

The methods batch deconvolution (BD) and constrained batch deconvolution (CBD) can give significantly smaller errors in the approximations of $\hat{f}_i(x)$ when compared with separate deconvolution (SD), particularly when the polynomial $\hat{f}_0(x)$ is of high degree and has few roots of high multiplicity.

The next section combines the work so far and considers examples where the square-free factorisation of a univariate polynomial in Bernstein form is computed.

4.3 Univariate Root Finding Results

This section now considers results of the square-free factorisation algorithm, and these results are compared with the standard MATLAB `roots()` method, as well as `multroot()` by Zeng.

The first stage of the square-free factorisation algorithm (Algorithm 1) requires the computation of the polynomials $\{\hat{f}_i(x)\}$, where each $\hat{f}_i(x)$ is the GCD of $\hat{f}_{i-1}(x)$ and its derivative. In the following examples, given a perturbed $f_0(x)$, two sets of approximations of $\{\hat{f}_i(x)\}$ are computed by two different methods which are outlined below:

Method 1 : This method employs the standard least squares based method (described in Section 3.5) for the computation of the coefficients of the sequence of GCDs. This gives the set of approximations $\{\tilde{f}_i(x)\}$.

Method 2 : In this method each $f_i(x)$ in the set of polynomials $\{f_i(x)\}$, generated by a set of GCD computations, has its coefficients computed using a low rank approximation of the t_ith subresultant matrix, where t_i is the degree of $f_i(x)$.

The errors between the exact polynomials $\{\hat{f}_i(x)\}$ and the two sets of approximations $\{\tilde{f}(x)\}$ and $\{\dot{f}(x)\}$ are given by Equation (3.65) and Equation (3.66) respectively.

Example 4.3.1. This example considers the two methods of computing approximations of the polynomials $\{\hat{f}_i(x)\}$ and five methods of deconvolution. No other root finding methods are considered for this example.

Consider the Bernstein form of the exact polynomial $\hat{f}(x)$, whose factorised form is given by

$$\hat{f}(x) = (x - 0.101)^{20}(x - 0.1)^{20}(x + 0.5)^2.$$

This example considers the computation of the square-free factorisation, and roots of the inexact polynomial $f(x)$. Noise is added to the coefficients of the polynomial $\hat{f}(x)$ such that the coefficients of the inexact polynomial $f(x)$ are given by $a_i = \hat{a}_i + r_i\hat{a}_i\epsilon_i$, where $\{r_i\}$ are uniformly distributed random variables in the interval $[-1, 1]$ and $\{\epsilon_i\} = 10^{-8}$. The polynomials $\{\hat{f}_i(x)\}$ are given by

$$\hat{f}_i(x) = \begin{cases} (x - 0.101)^{20-i}(x - 0.1)^{20-i}(x + 0.5)^{2-i} & i = 0, 1, \\ (x - 0.101)^{20-i}(x - 0.1)^{20-i} & i = 2, \ldots, 19, \\ 1 & i = 20. \end{cases}$$

The first stage of the square-free factorisation algorithm computes approximations of these polynomials. Figure 4.11 shows the error between exact polynomials $\{\hat{f}_i(x)\}$ and the two sets of approximations $\{\tilde{f}_i(x)\}$ and $\{\dot{f}_i(x)\}$ obtained by **Method 1** and **Method 2** respectively.

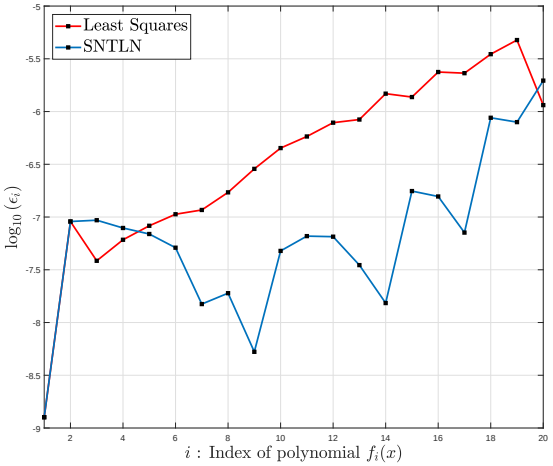


Figure 4.11: Error in approximation of $\{\,\hat{f}_i(x) \mid i = 1, \ldots, 20\,\}$ by (i) **Method 1** (■) and (ii) **Method 2** (■) in Example 4.3.1

Given the set of approximations $\dot{f}_i(x)$ obtained by **Method 2**, the second part of the square-free factorisation algorithm computes approximations of the polynomials $\{\hat{h}_i(x)\}$. The five deconvolution methods are used in this problem and are compared. In Figure 4.12 the error in each approximation $h_i(x)$ is plotted and it is clear to see that the best approximations are obtained by the constrained batch deconvolution with STLN (CBDSTLN) method.

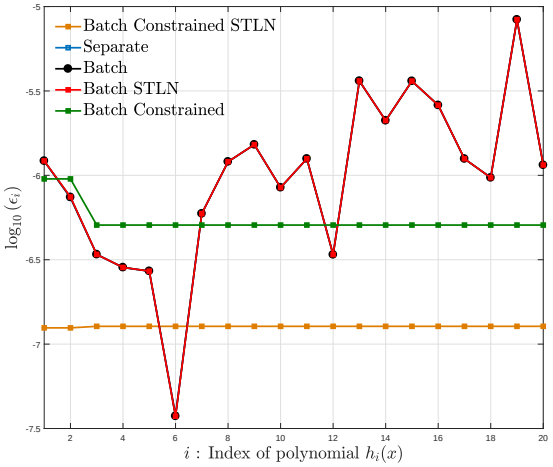


Figure 4.12: Error in the approximations of $\{\hat{h}_i(x)\}$ computed using five deconvolution methods in Example 4.3.1

□

In the following examples two methods are considered for the square-free factorisation problem :

Method 1 : The first method used in the following examples proceeds as follows:

1. The approximations $\{\tilde{f}_i(x)\}$ are computed by the GCD finding method which makes use of least squares approximation in Section 3.5.1.
2. The polynomials $\{\hat{h}_i(x)\}$ are approximated using the separate deconvolution (SD) method, that is, the approximations $\{\tilde{h}_i(x)\}$ are given by $\tilde{f}_{i-1}/\tilde{f}_i$ as described in Section 4.2.1.

3. The polynomials $\{\tilde{w}_i(x)\}$ are given by deconvolution of the set $\{\tilde{h}_i(x)\}$ where the batch deconvolution (BD) method is used.

Method 2 (SQFF) : The second method considered in the following examples is given the name square-free factorisation (SQFF) and proceeds as follows :

1. The approximations $\{\dot{f}_i(x)\}$ are computed by the GCD finding method which makes use of the low rank approximation of the tth subresultant matrix as seen in Section 3.5.2.
2. The polynomials in the set $\{\dot{f}_i(x)\}$ are deconvolved using the constrained batch deconvolution with STLN (CBDSTLN) method as described in Section 4.2.5 to obtain the approximations $\{\dot{h}_i(x)\}$.
3. The approximations $\{\dot{w}_i(x)\}$ are computed by batch deconvolution (BD) of the polynomials $\{\dot{h}_i(x)\}$ as described above.

Example 4.3.2. Consider the Bernstein form of the exact polynomial $\hat{f}(x)$ of degree $m = 45$, whose factorisation is given by

$$\hat{f}(x) = (x - 3.216789879)^{13}(x - 1.23456)^5(x - 0.75)^{15}(x - 0.4)^{10}(x + 0.12687)^2.$$

Noise is added to the coefficients of $\hat{f}(x)$, and the coefficients of the inexact polynomial $f(x)$ are given by $a_i = \hat{a}_i + \hat{a}_i r_i \epsilon_i$, where $\{r_i\}$ are uniformly distributed random variables in the interval $[-1, 1]$ and $\{\epsilon_i\}$ are uniformly distributed random variables in the interval $[10^{-10}, 10^{-8}]$.

Figure 4.13i plots the errors in the sets of approximations $\{\tilde{f}_i(x)\}$, $\{\tilde{h}_i(x)\}$ and $\{\tilde{w}_i(x)\}$ and Figure 4.13ii plots the errors in the approximations $\{\dot{f}_i(x)\}$, $\{\dot{h}_i(x)\}$ and $\{\dot{w}_i(x)\}$, where the two batches of approximations are obtained by **Method 1** and **Method 2 (SQFF)** respectively. The roots $\{r_i\}$ computed by **Method 1** and **Method 2 (SQFF)** are shown in Table 4.5 and Table 4.6 respectively.

These indicate that the errors in approximations obtained by **Method 2 (SQFF)** are smaller than those obtained by **Method 1**, while both methods correctly identify the root multiplicity structure.

Computed Root r_i	Error $\lvert\hat{r}_i - r_i\rvert$	Root Multiplicity
-0.126869586179687	$4.1382e-07$	2
1.234560435234715	$4.3523e-07$	5
0.399988340390411	$1.1660e-05$	10
3.216453021241865	$3.3686e-04$	13
0.750002359901100	$2.3599e-06$	15

Table 4.5: Roots and root multiplicities computed by **Method 1** in Example 4.3.2

Computed Root r_i	Error $\lvert\hat{r}_i - r_i\rvert$	Root Multiplicity
-0.126870364702462	$3.6470e-07$	2
1.234560309624994	$3.0962e-07$	5
0.400000368352912	$3.6835e-07$	10
3.216807497210620	$1.7618e-05$	13
0.750000508265343	$5.0827e-07$	15

Table 4.6: Roots and root multiplicities computed by **Method 2 (SQFF)** in Example 4.3.2

The average errors in the approximations of $\{\hat{f}_i(x)\}$, $\{\hat{h}_i(x)\}$ and $\{\hat{w}_i(x)\}$ for each of the methods are given in Table 4.7. The errors in the approximations $\{\tilde{f}_i(x)\}$ computed by the simple least squares based method are typically larger than the errors in the approximations $\{\dot{f}(x)\}$.

	Method 1	**Method 2 (SQFF)**
$\epsilon_{avg}\ \{\hat{f}_i(x)\}$	$2.560914e-06$	$7.997036e-07$
$\epsilon_{avg}\ \{\hat{h}_i(x)\}$	$1.068434e-05$	$9.474574e-07$
$\epsilon_{avg}\ \{\hat{w}_i(x)\}$	$9.773989e-06$	$6.944887e-07$

Table 4.7: Error in the approximations of the sets of polynomials $\{\hat{f}_i(x)\}$, $\{\hat{h}_i(x)\}$ and $\{\hat{w}_i(x)\}$ in Example 4.3.2

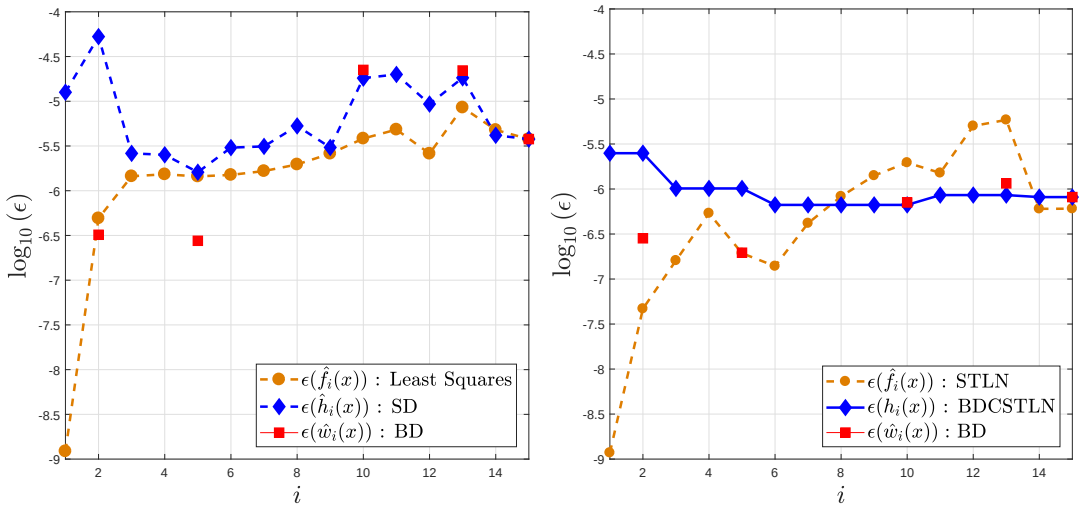
Results from **Method 2 (SQFF)** are compared with the MATLAB `roots()` method and Zeng's `multroot()`. Both MATLAB and Zeng's methods fail to retain the multiplicity structure of the roots of $\hat{f}(x)$ in the presence of noise. Consequently, clusters of roots are computed in complex conjugate pairs and these surround the exact roots (see Figure 4.14). The errors in the approximations of the roots by these two methods are significant, but their respective backward errors are small. However, **Method 2 (SQFF)** retains the multiplicity structure.

Example 4.3.3. Consider the Bernstein form of the exact polynomial $\hat{f}(x)$ of degree $m = 11$, whose factorised form is given by

$$\hat{f}(x) = (x - 0.5)^4(x + 0.75)^7.$$

Noise is added to the coefficients of $\hat{f}(x)$ such that the coefficients of the inexact polynomial $f(x)$ are given by $a_i = \hat{a}_i + \hat{a}_i \times r_i \times 10^{-8}$, where $\{r_i\}$ are uniformly distributed random variables in the interval $[-1, 1]$.

Approximations of the sets of polynomials $\{\hat{f}_i(x)\}$, $\{\hat{h}_i(x)\}$ and $\{\hat{w}_i(x)\}$ are computed by **Method 1** and **Method 2 (SQFF)** and the errors for both of these methods are plotted in Figure 4.15i and Figure 4.15ii respectively. The average errors are then given in Table 4.8. The sets of roots approximated by **Method 1** and **Method 2 (SQFF)** and their respective errors are given in Table 4.9 and Table 4.10 respectively.



(i) Method 1 **(ii) Method 2 (SQFF)**

Figure 4.13: Average error in the approximations of $\{\hat{f}_i(x)\}$, $\{\hat{h}_i(x)\}$ and $\{\hat{w}_i(x)\}$ approximated by (i) **Method 1** and (ii) **Method 2 (SQFF)** in Example 4.3.2

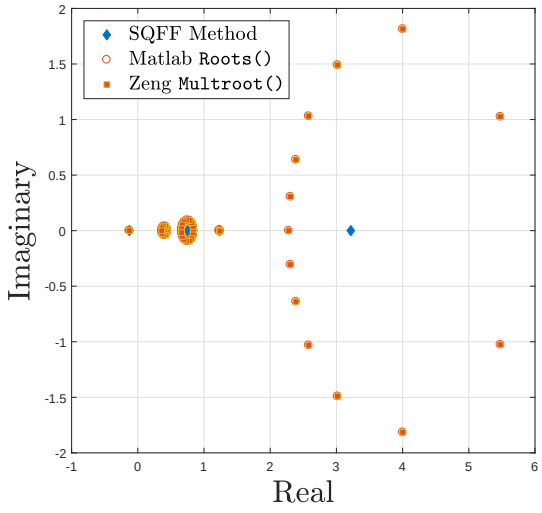


Figure 4.14: Roots of $\hat{f}(x)$ as approximated by (i) SQFF, (ii) MATLAB `roots()` and (iii) `multroot()` in Example 4.3.2

	Method 1	**Method 2 (SQFF)**
$\epsilon_{avg}\{\hat{f}_i(x)\}$	$3.140354e-09$	$4.731169e-09$
$\epsilon_{avg}\{\hat{h}_i(x)\}$	$1.182284e-08$	$1.584437e-09$
$\epsilon_{avg}\{\hat{w}_i(x)\}$	$9.779087e-09$	$8.945724e-10$

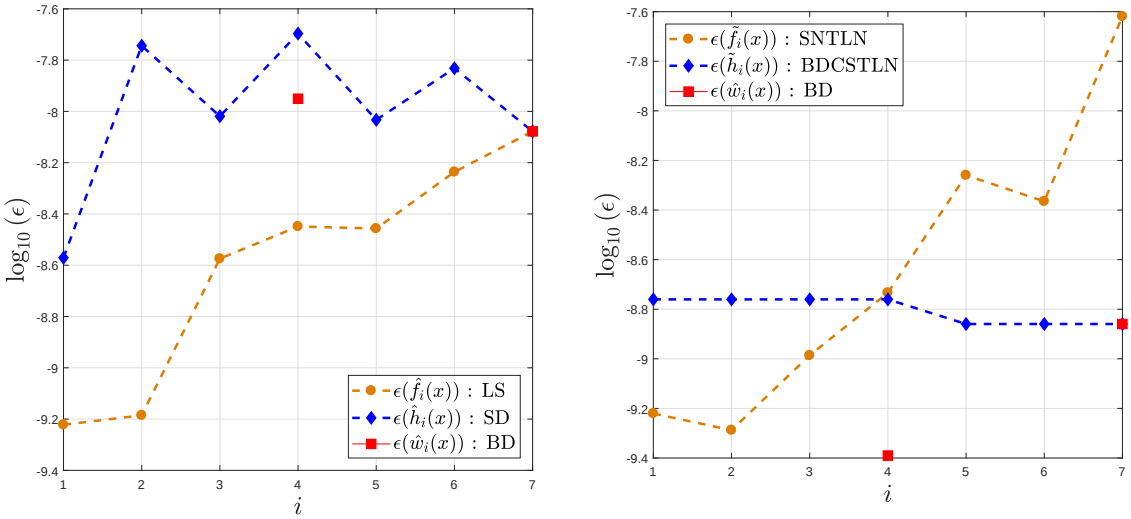
Table 4.8: Error in the approximations of $\{\hat{f}_i(x)\}$, $\{\hat{h}_i(x)\}$ and $\{\hat{w}_i(x)\}$ in Example 4.3.3

Computed Root r_i	Error $\lvert\hat{r}_i - r_i\rvert$	Root Multiplicity
0.499999994397712	$5.6023e-09$	4
-0.750000030281793	$3.0282e-08$	7

Table 4.9: Roots and root multiplicities approximated by **Method 1** in Example 4.3.3

Computed Root r_i	Error $\lvert\hat{r}_i - r_i\rvert$	Root Multiplicity
0.499999999796249	$-2.0375e-10$	4
-0.749999994991544	$-5.0085e-09$	7

Table 4.10: Roots and root multiplicities approximated by **Method 2 (SQFF)** in Example 4.3.3



(i) Method 1 **(ii) Method 2 (SQFF)**

Figure 4.15: Average error in the approximations of $\{\hat{f}_i(x)\}$, $\{\hat{h}_i(x)\}$ and $\{\hat{w}_i(x)\}$ approximated by (i) **Method 1** and (ii) **Method 2 (SQFF)** in Example 4.3.3

□

Many other combinations of preprocessing, low rank approximation methods and deconvolution methods can be considered, but generally best results are obtained by methods which :

1. Preprocess the polynomials in the GCD problem
2. Exploit structure preserving low rank approximation methods
3. Make use of constrained deconvolution methods

4.4 Conclusion

This section has considered the computation of the factorisation of a univariate polynomial in Bernstein form. Gauss' algorithm was used in the computation of the square-free

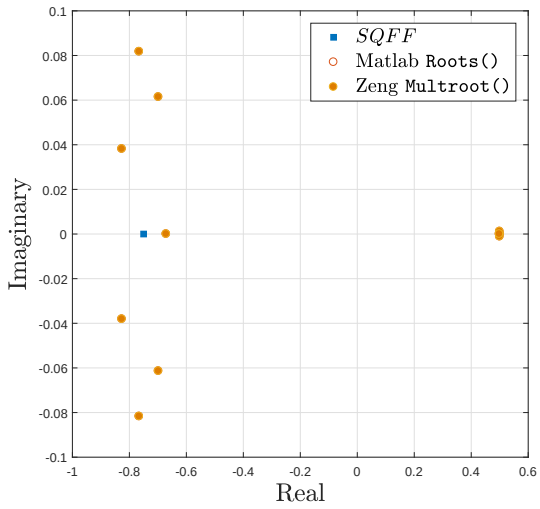


Figure 4.16: Roots of $\hat{f}(x)$ as approximated by (i) SQFF, (ii) MATLAB `roots()` and (iii) `multroot()` in Example 4.3.3

factorisation, which makes use of a sequence of GCD computations and two sets of polynomial deconvolutions. The simple roots of the square-free polynomials $\{\hat{w}_i(x)\}$ are the roots of $\hat{f}(x)$ of multiplicity i.

The MUGCD Method : The modified univariate GCD (MUGCD) method was developed in this chapter and modifies the standard univariate GCD (UGCD) method developed in the previous chapter. This method is significantly faster when dealing with the type of GCD problem found in the square-free factorisation algorithm (Algorithm 1). Using this method, fewer subresultant matrices must be constructed and preprocessed. This also reduces the number of required SVD operations.

The Deconvolution Problem : Several deconvolution methods were considered and the best approximations of the set of polynomials $\{\hat{h}_i(x)\}$ were shown to be obtained using the structured matrix methods of batch deconvolution (BD) and constrained batch deconvolution (CBD).

The Square-Free Factorisation Problem : Results from the square-free factorisation (SQFF) method developed in this chapter compare favourably with MATLAB `roots()` and Zeng's `multroots()` methods. The multiplicity structure is generally preserved in SQFF where the other methods fail. This can be seen in Example 4.3.2.

The extension of the square-free factorisation algorithm (Algorithm 1) to determine the factorisation of a bivariate polynomial will be described in Section 6.1. This algorithm requires the computation of the GCD of three bivariate polynomials. First, the UGCD method described in Chapter 3 is extended to compute the GCD of three univariate polynomials. Following this, a simple extensions to compute the GCD of three bivariate polynomials is derived.

Chapter 5

The Univariate Polynomial GCD - The Three Polynomial Problem

Chapter 3 described the UGCD method for the computation of the GCD of two exact polynomials $\hat{f}(x)$ and $\hat{g}(x)$ or the AGCD of two inexact polynomials $f(x)$ and $g(x)$, where the polynomials were defined in Bernstein form.

For the factorisation of a univariate polynomial, the square-free factorisation algorithm (Algorithm 1) generates the set $\{\hat{f}_i(x)\}$, where $\hat{f}_{i+1}(x) = \text{GCD}(\hat{f}_i(x), \hat{f}_i'(x))$ as discussed in Chapter 4. An extension of the above mentioned algorithm allows for the computation of the square-free factorisation of a bivariate polynomial $\hat{f}(x, y)$. In this algorithm the set of polynomials $\{\hat{f}_i(x, y)\}$ is generated, where each $\hat{f}_{i+1}(x, y)$ is given by

$$\hat{f}_{i+1}(x, y) = \text{GCD}\left(\hat{f}_i(x, y), \frac{\partial \hat{f}_i(x, y)}{\partial x}, \frac{\partial \hat{f}_i(x, y)}{\partial y}\right).$$

Therefore, it is necessary to develop a robust method for the computation of the GCD of three bivariate polynomials. This introduces two new problems. They are the computation of the GCD of three polynomials and the computation of the GCD of bivariate polynomials. This chapter first extends the univariate GCD (UGCD) method to compute the GCD of three polynomials, $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$, using similar structured matrix methods. Subsequent chapters will discuss the extension to compute the GCD of two or three bivariate polynomials.

One approach to the problem is to first compute the GCD $\hat{d}_a(x)$ of any two of the three polynomials, say $\hat{f}(x)$ and $\hat{g}(x)$, then compute the GCD of the result and the third polynomial $\hat{h}(x)$. That is,

$$\begin{aligned}\hat{d}(x) = \text{GCD}\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right) &= \text{GCD}\left(\text{GCD}\left(\hat{f}(x), \hat{g}(x)\right), \hat{h}(x)\right) \\ &= \text{GCD}\left(\text{GCD}\left(\hat{f}(x), \hat{h}(x)\right), \hat{g}(x)\right) \\ &= \text{GCD}\left(\text{GCD}\left(\hat{g}(x), \hat{h}(x)\right), \hat{f}(x)\right).\end{aligned}$$

Let $\hat{p}(x)$ be the GCD of $\hat{f}(x)$ and $\hat{g}(x)$, then it may be that the GCD of $\hat{p}(x)$ and $\hat{h}(x)$, that is $d_t(x)$, is also equal to $\hat{p}(x)$, and therefore all of the subresultant matrices in the set $\{S_k(\hat{p}(x), \hat{h}(x))\}$ are singular. However, it is this type of problem which forms one of the

exceptions to the univariate GCD (UGCD) finding algorithm discussed in Section 3.2.5.

The AGCD of any two of three polynomials may be less well defined than the AGCD of any other pair. That is to say, by the method described in Chapter 3, the AGCD is not recoverable by analysis of the numerical rank of the subresultant matrices. However, another pairing may avoid this problem. For example, $f(x)$ and $g(x)$ may have a poorly defined AGCD but $g(x)$ and $h(x)$ may have a well-defined AGCD such that

$$\text{AGCD}\Big(\text{AGCD}\left(f(x), g(x)\right), h(x)\Big) \neq \text{AGCD}\Big(f(x), \text{AGCD}\left(g(x), h(x)\right)\Big) = d(x).$$

When computing the AGCD of three inexact polynomials, $f(x)$, $g(x)$ and $h(x)$, the AGCD of $d_a(x)$ $\Big($the AGCD of $f(x)$ and $g(x)\Big)$ and $h(x)$ can become coprime, since the first AGCD computation moves the result $p(x)$ away from the point which would represent the AGCD of all three polynomials.

Improved results are obtained by utilising a method where the GCD of all three polynomials is considered simultaneously, and for this a new structured matrix based method is considered. The new set of three-polynomial subresultant matrices extends the definition of the two-polynomial subresultant matrices to a form consistent with the three-polynomial problem.

The bivariate or n-variate Sylvester matrix and sequence of subresultant matrices is covered in a small number of publications [32,39,53,79]. However, significantly less work is found on the Sylvester matrix of three polynomials in Bernstein form, and this is addressed in this chapter.

Section 5.1 The determination of the degree of the GCD of three polynomials reduces to the computation of the numerical rank of a set of subresultant matrices. This is similar to the method already seen for the computation of the degree of the GCD of two polynomials in Chapter 3. There are two variations of the subresultant matrices for the three-polynomial problem, namely $\{\hat{S}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\}$ and $\{\tilde{S}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\}$, which have (2×3) and (3×3) partitioned structures respectively, and the structure of these matrices is discussed.

This section also considers the optimal ordering of polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ for inclusion in the (2×3) partitioned subresultant matrix, where each ordering gives rise to a set of subresultant matrices of different dimensions. The entries of these matrices can be of significantly different magnitudes, and their numerical ranks are not consistent.

Section 5.2 The optimal variant of the two-polynomial subresultant matrix sequence was described in Section 3.1.4 and was given by the set of subresultant matrices of the form $\{D^{-1}_{m+n-k} T_k(\hat{f}(x), \hat{g}(x))\hat{Q}_k \mid k = 1, \ldots, \min(m, n)\}$. This section considers the optimal variant of the three-polynomial subresultant matrix sequence.

Section 5.3 Preprocessing the two-polynomial subresultant matrices was considered in Section 3.4, and it was shown that the degree t of the AGCD was more reliably computed from the subresultant matrices of preprocessed polynomials. The preprocessed tth subresultant matrix also gave improved approximations of the coefficients of the GCD. This section extends the preprocessing, such that the (2×3) and (3×3)

partitioned subresultant matrices of three polynomials are preprocessed. It will be shown that the preprocessing of these three-polynomial subresultant matrices has similar benefits in the computation of the degree of the three-polynomial GCD with similar results. It will also be shown how poor scaling can arise amongst the row-partitions, should the polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{g}(x)$ be of significantly different degree.

Section 5.4 Approximations of the cofactor polynomials $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{w}_t(x)$ and the GCD $\hat{d}_t(x)$ are computed by least squares and the method is described in this section.

Section 5.5 This section presents a set of examples in which the degree of the GCD of three polynomials is computed using the subresultant matrices of unprocessed and preprocessed polynomials.

Approximations of the coefficients of cofactor polynomials and the GCD are computed from the tth unprocessed and tth preprocessed subresultant matrix and it is shown that approximations from the preprocessed subresultant matrix are significantly better.

5.1 The Degree of the Three-Polynomial GCD

The three polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ of degrees m, n and o respectively, have a common divisor $\hat{d}_k(x)$ of degree k if and only if there exist cofactor polynomials $\hat{u}_k(x)$, $\hat{v}_k(x)$ and $\hat{w}_k(x)$ of degrees $(m-k)$, $(n-k)$ and $(o-k)$ respectively, such that

$$\frac{\hat{f}(x)}{\hat{u}_k(x)} = \hat{d}_k(x), \qquad \frac{\hat{g}(x)}{\hat{v}_k(x)} = \hat{d}_k(x) \qquad \text{and} \qquad \frac{\hat{h}(x)}{\hat{w}_k(x)} = \hat{d}_k(x).$$

A divisor $\hat{d}_k(x)$ is common to all three polynomials if all three of the equations

$$\hat{f}(x)\hat{v}_k(x) - \hat{g}(x)\hat{u}_k(x) = 0 \tag{5.1}$$

$$\hat{f}(x)\hat{w}_k(x) - \hat{h}(x)\hat{u}_k(x) = 0 \tag{5.2}$$

$$\hat{g}(x)\hat{w}_k(x) - \hat{h}(x)\hat{v}_k(x) = 0 \tag{5.3}$$

are simultaneously satisfied.

As with the two-polynomial GCD problem, these equations can be written in matrix form. There are two variations of the three-polynomial subresultant matrices with (3×3) and (2×3) partitioned structures respectively. The (2×3) partitioned subresultant matrix takes three different forms, dependent on which two of the three equations are considered. These variations will be defined later in this section, but first the (3×3) partitioned subresultant matrix is defined.

The three equations (5.1, 5.2, 5.3) can be written in matrix form as

$$\tilde{S}_k\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right)\mathbf{x}_k = \mathbf{0}, \tag{5.4}$$

which has non-trivial solutions for $k = 1, \ldots, t$ and the solution vector $\mathbf{x}_k$ is given by

$$\mathbf{x}_k = \left[\begin{array}{ccc} \hat{\mathbf{v}}_k, & \hat{\mathbf{w}}_k, & -\hat{\mathbf{u}}_k \end{array} \right]^T, \tag{5.5}$$

where vectors $\hat{\mathbf{v}}_k$, $\hat{\mathbf{w}}_k$ and $\hat{\mathbf{u}}_k$ contain the coefficients of the cofactor polynomials $\hat{v}_k(x)$, $\hat{w}_k(x)$ and $\hat{u}_k(x)$ respectively. The matrix $\tilde{S}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ in (5.4) is the (3×3) partitioned subresultant matrix and is given by

$$\tilde{S}_k\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right) = \tilde{D}_k^{-1} \tilde{T}_k\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right) \tilde{Q}_k. \tag{5.6}$$

The block diagonal matrix $\tilde{D}_k^{-1}$ of order $(2m + 2n + 2o - 3k + 3)$ is given by

$$\tilde{D}_k^{-1} = \operatorname{diag}\left[\begin{array}{ccc} D_{m+n-k}^{-1}, & D_{m+o-k}^{-1}, & D_{n+o-k}^{-1} \end{array} \right],$$

where D_{m+n-k}^{-1} is defined in (3.6) and the matrices D_{m+o-k}^{-1} and D_{n+o-k}^{-1} share the same structure. The matrix $\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ is given by

$$\tilde{T}_k\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right) = \left[\begin{array}{ccc} T_{n-k}\left(\hat{f}(x)\right) & & T_{m-k}\left(\hat{g}(x)\right) \\ & T_{o-k}\left(\hat{f}(x)\right) & T_{m-k}\left(\hat{h}(x)\right) \\ T_{n-k}\left(\hat{h}(x)\right) & -T_{o-k}\left(\hat{g}(x)\right) & \end{array} \right],$$

where each partition of the form $T_{n^*-k}(\hat{f}^*(x))$ is the $(n^* - k)$th order Toeplitz matrix as defined in (2.11). The block diagonal matrix $\tilde{Q}_k$ of order $(m + n + o - 3k + 3)$ is given by

$$\tilde{Q}_k = \operatorname{diag}\left[\begin{array}{ccc} Q_{n-k}, & Q_{o-k}, & Q_{m-k} \end{array} \right], \tag{5.7}$$

where Q_{n-k}, Q_{m-k} and Q_{o-k} have the same structure as the diagonal matrix $Q_n \in \mathbb{R}^{(n+1)\times(n+1)}$ defined in (2.12).

The kth subresultant matrix of three univariate polynomials is therefore given by

$$\tilde{S}_k\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right) = \left[\begin{array}{c} \mathcal{R}_{a,k} \\ \mathcal{R}_{b,k} \\ \mathcal{R}_{c,k} \end{array} \right] = \left[\begin{array}{ccc} C_{n-k}\left(\hat{f}(x)\right) & & C_{m-k}\left(\hat{g}(x)\right) \\ & C_{o-k}\left(\hat{f}(x)\right) & C_{m-k}\left(\hat{h}(x)\right) \\ C_{n-k}\left(\hat{h}(x)\right) & -C_{o-k}\left(\hat{g}(x)\right) & \end{array} \right], \tag{5.8}$$

where each partition $C_{n-k}(\hat{f}(x)) = D_{m+n-k}^{-1} T_{n-k}(\hat{f}(x)) Q_{n-k}$ is a univariate convolution matrix as defined in (2.9) and $\mathcal{R}_{a,k}$, $\mathcal{R}_{b,k}$ and $\mathcal{R}_{c,k}$ are the row-partitions given by

$$\mathcal{R}_{a,k} = \left[\begin{array}{ccc} C_{n-k}\left(\hat{f}(x)\right) & 0_{m+n-k+1,o-k+1} & C_{m-k}\left(\hat{g}(x)\right) \end{array} \right], \tag{5.9}$$

$$\mathcal{R}_{b,k} = \left[\begin{array}{ccc} 0_{m+o-k+1,n-k+1} & C_{o-k}\left(\hat{f}(x)\right) & C_{m-k}\left(\hat{h}(x)\right) \end{array} \right], \tag{5.10}$$

$$\mathcal{R}_{c,k} = \left[\begin{array}{ccc} C_{n-k}\left(h(x)\right) & -C_{o-k}\left(g(x)\right) & 0_{n+o-k+1,n-k+1} \end{array} \right]. \tag{5.11}$$

Having defined the (3×3) partitioned subresultant matrix, the (2×3) subresultant matrix is now defined. Two of the equations (5.1, 5.2, 5.3) are sufficient to describe the system of equations, and the third equation can be derived given the first two. This is

equivalent to solving two of the GCD problems simultaneously

$$\text{GCD}\left(\text{GCD}\left(\hat{f}^*(x),\hat{g}^*(x)\right),\text{GCD}\left(\hat{f}^*(x),\hat{h}^*(x)\right)\right).^{(i)} \tag{5.12}$$

The three variations of the (2×3) partitioned subresultant matrix are defined by the three ways of choosing two of the above equations. *Variation 1 :* Equations (5.1) and (5.2) can be written in matrix form as

$$\hat{S}_k\left(\hat{f}(x),\hat{g}(x),\hat{h}(x)\right)\mathbf{x}_{k,1}=\mathbf{0}, \tag{5.13}$$

which has a non-zero solution for $k=1,\ldots,t$, and the solution vector $\mathbf{x}_{k,1}$ in (5.13) is equal to the solution vector $\mathbf{x}_k$ defined in (5.5). The (2×3) partitioned subresultant matrix $\hat{S}_k(\hat{f}(x),\hat{g}(x),\hat{h}(x))$ in (5.13) is of dimension $(2m+n+o-2k+2)\times(m+n+o-3k+3)$ and is given by

$$\hat{S}_k(\hat{f}(x),\hat{g}(x),\hat{h}(x))=\begin{bmatrix}\mathcal{R}_{a,k}\\ \mathcal{R}_{b,k}\end{bmatrix}=\begin{bmatrix} C_{n-k}\left(\hat{f}(x)\right) & & C_{m-k}\left(\hat{g}(x)\right)\\ & C_{o-k}\left(\hat{f}(x)\right) & C_{m-k}\left(\hat{h}(x)\right)\end{bmatrix},$$

where $\mathcal{R}_{a,k}$ and $\mathcal{R}_{b,k}$ are row-partitions defined in (5.9) and (5.10) respectively. *Variation 2 :* Equations (5.1) and (5.3) are written in matrix form as

$$\hat{S}_k\left(\hat{g}(x),\hat{f}(x),\hat{h}(x)\right)\mathbf{x}_{k,2}=\mathbf{0}, \tag{5.14}$$

which has non-zero solutions for $k=1,\ldots,t$, and the vector $\mathbf{x}_{k,2}$ is a rearrangement of the component vectors of $\mathbf{x}_{k,1}$, given by

$$\mathbf{x}_{k,2}=\begin{bmatrix}\hat{\mathbf{u}}_k & \hat{\mathbf{w}}_k & -\hat{\mathbf{v}}_k\end{bmatrix}^T.$$

The subresultant matrix $\hat{S}_k(\hat{g}(x),\hat{f}(x),\hat{h}(x))$ is of dimension $(m+2n+o-2k+2)\times(m+n+o-3k+3)$ and is given by

$$\hat{S}_k(\hat{g}(x),\hat{f}(x),\hat{h}(x))=\begin{bmatrix}\tilde{\mathcal{R}}_{a,k}\\ \tilde{\mathcal{R}}_{c,k}\end{bmatrix}=\begin{bmatrix} C_{m-k}\left(\hat{g}(x)\right) & & C_{n-k}\left(\hat{f}(x)\right)\\ & C_{o-k}\left(\hat{g}(x)\right) & C_{n-k}\left(\hat{h}(x)\right)\end{bmatrix},$$

where $\tilde{\mathcal{R}}_{a,k}$ and $\tilde{\mathcal{R}}_{c,k}$ are rearranged row-partitions $\mathcal{R}_{a,k}$ and $\mathcal{R}_{c,k}$. *Variation 3 :* The equations (5.2) and (5.3) give rise to the system

$$\hat{S}_k\left(\hat{h}(x),\hat{g}(x),\hat{f}(x)\right)\mathbf{x}_{k,3}=\mathbf{0}, \tag{5.15}$$

which has non-zero solutions for $k=1,\ldots,t$, and the solution vector $\mathbf{x}_{k,3}$ is a rearranged

[(i)] Where * indicates that $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ can be considered in any order.

form of $\mathbf{x}_{k,1}$ which is given by

$$\mathbf{x}_{k,3} = \begin{bmatrix} \hat{\mathbf{v}}_k, & \hat{\mathbf{u}}_k, & -\hat{\mathbf{w}}_k \end{bmatrix}^T.$$

The subresultant matrix $S_k(\hat{h}(x), \hat{g}(x), \hat{f}(x))$ in (5.15) of dimension $(m+n+2o-2k+2) \times (m+n+o-3k+3)$ is the third variation of the (2×3) partitioned subresultant matrix. It can be written as

$$\begin{bmatrix} \tilde{\mathcal{R}}_{b,k} \\ \tilde{\mathcal{R}}_{c,k} \end{bmatrix} = \begin{bmatrix} C_{n-k}\left(\hat{h}(x)\right) & & C_{o-k}\left(\hat{g}(x)\right) \\ & C_{m-k}\left(\hat{h}(x)\right) & C_{o-k}\left(\hat{f}(x)\right) \end{bmatrix},$$

where $\tilde{\mathcal{R}}_{b,k}$ and $\tilde{\mathcal{R}}_{c,k}$ are rearranged forms of $\mathcal{R}_{b,k}$ and $\mathcal{R}_{c,k}$.

Since there are three variations of the (2×3) partitioned subresultant matrix, the notation $\hat{S}_k(\hat{f}^*(x), \hat{g}^*(x), \hat{h}^*(x))$ is used to denote that polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ can be considered in any order, where each ordering gives one of the three variations of subresultant matrices.

Computing the Degree of the Three-Polynomial GCD

The different systems of equations (5.4, 5.13, 5.14, 5.15) only have non-trivial solutions when k is the degree of a common divisor of polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$, therefore

$$\begin{aligned}
\text{rank}\left(\tilde{S}_k\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right)\right) &< 2m+2n+2o-3k+3 \quad \text{for} \quad k = 1, \ldots, t, \\
\text{rank}\left(\tilde{S}_k\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right)\right) &= 2m+2n+2o-3k+3 \quad \text{for} \quad k = t+1, \ldots, \min(m,n,o), \\
\text{rank}\left(\hat{S}_k\left(\hat{f}^*(x), \hat{g}^*(x), \hat{h}^*(x)\right)\right) &< 2m^*+n^*+o^*-3k+3 \quad \text{for} \quad k = 1, \ldots, t, \\
\text{rank}\left(\hat{S}_k\left(\hat{f}^*(x), \hat{g}^*(x), \hat{h}^*(x)\right)\right) &= 2m^*+n^*+o^*-3k+3 \quad \text{for} \quad k = t+1, \ldots, \min(m,n,o).
\end{aligned}$$

As with the two-polynomial GCD problem, poorly scaled three-polynomial subresultant matrices can lack the desired property of having clear separation between the zero and non-zero singular values in their SVD. One source of poor scaling in the two-polynomial problem came from a large difference in the entries in the two column-partitions $C_{n-k}(\hat{f}(x))$ and $C_{m-k}(\hat{g}(x))$. This remains true for three-polynomial subresultant matrices, but poor scaling amongst the row-partitions must also be considered. It will be shown that poor scaling of this type can lead to erroneous GCD degree computations.

The following examples consider the effect of poor scaling amongst the row-partitions. First, however, it is necessary to define the pairwise GCDs for each pair of the three polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$. Let the pairwise GCDs be defined as

$$\hat{d}_a(x) = \text{GCD}(\hat{f}(x), \hat{g}(x)), \tag{5.16}$$

$$\hat{d}_b(x) = \text{GCD}(\hat{f}(x), \hat{h}(x)), \tag{5.17}$$

$$\hat{d}_c(x) = \text{GCD}(\hat{g}(x), \hat{h}(x)), \tag{5.18}$$

where $\hat{d}_a(x)$, $\hat{d}_b(x)$ and $\hat{d}_c(x)$ are of degree t_a, t_b and t_c respectively.

The row-partition $\mathcal{R}_{a,k}$ in (5.9) is singular for $k = 1, \ldots, t_a$, $\mathcal{R}_{b,k}$ in (5.10) is singular

for $k = 1, \ldots, t_b$ and $\mathcal{R}_{c,k}$ in (5.11) is singular for $k = 1, \ldots, t_c$. Since the three-polynomial subresultant matrices consist of various arrangements of these row-partitions, the computation of the degree of the GCD is affected by poor scaling amongst them. The following theoretical example shows how poor scaling can cause the degree of the GCD to be determined incorrectly.

Example 5.1.1. Consider the three polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ of degrees m, n and o respectively. All three polynomials have a GCD $\hat{d}_t(x)$ of degree t and the pairwise GCDs of degrees t_a, t_b and t_c are defined in (5.16, 5.17, 5.18). For this example, assume that $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ are polynomials in the power basis since this avoids scaling in the subresultant matrices due to binomial terms. The polynomials $\hat{f}(x)$ and $\hat{g}(x)$ are both scaled by 10^5 and $\hat{h}(x)$ is scaled by 10^{-5}

$$f(x) = \hat{f}(x) \times 10^5, \qquad g(x) = \hat{g}(x) \times 10^5 \qquad \text{and} \quad h(x) = \hat{h}(x) \times 10^{-5}.$$

The system of equations

$$\hat{S}_k\left(f(x), g(x), h(x)\right) \begin{bmatrix} \mathbf{v}_k \\ \mathbf{w}_k \\ -\mathbf{u}_k \end{bmatrix} = \mathbf{0} \tag{5.19}$$

has non-zero solutions for $k = 1, \ldots, t$. However, the kth (2×3) subresultant matrix $\hat{S}_k(f(x), g(x), h(x))$ contains the partition $C_{m-k}(h(x))$, whose entries are approximately zero relative to the large entries of the partitions containing the coefficients of $f(x)$ and $g(x)$

$$\begin{bmatrix} C_{n-k}\left(f(x)\right) & & C_{m-k}\left(g(x)\right) \\ & C_{o-k}\left(f(x)\right) & C_{m-k}\left(h(x)\right) \end{bmatrix} \approx \begin{bmatrix} C_{n-k}\left(f(x)\right) & & C_{m-k}\left(g(x)\right) \\ & C_{o-k}\left(f(x)\right) & \mathbf{0} \end{bmatrix}.$$

Therefore, the second row-partition $\mathcal{R}_{b,k}$ is approximately full rank, so the rank loss of $\hat{S}_k(f(x), g(x), h(x))$ effectively reduces to the rank loss of

$$\left[\; C_{n-k}\left(f(x)\right) \;\middle|\; C_{m-k}\left(g(x)\right) \;\right].$$

Since

$$\left[\; C_{n-k}\left(f(x)\right) \;\middle|\; C_{m-k}\left(g(x)\right) \;\right] \begin{bmatrix} \mathbf{v}_k \\ -\mathbf{u}_k \end{bmatrix} = \mathbf{0}$$

has non-trivial solutions for $k = 1, \ldots, t_a$, the degree of the three-polynomial GCD, $\hat{d}_t(x)$, is erroneously computed to be equal to t_a.

□

Each of the three combinations of row-partitions in the three (2×3) partitioned subresultant matrices theoretically have the same rank

$$\operatorname{rank}\left(\hat{S}_k\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right)\right) = \operatorname{rank}\left(\hat{S}_k\left(\hat{g}(x), \hat{f}(x), \hat{h}(x)\right)\right) = \operatorname{rank}\left(\hat{S}_k\left(\hat{h}(x), \hat{g}(x), \hat{f}(x)\right)\right),$$

but in practice, as shown by the Example 5.1.1, polynomials whose coefficients span significantly different orders of magnitude or polynomials of significantly different degree (when in Bernstein form) can give rise to three subresultant matrices of different numerical rank and a numeric example is now given.

Example 5.1.2. This example shows the difficulties in the computation of the degree of the three-polynomial GCD where two of the three polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ have a GCD of degree t_a, which is greater than the degree t of the GCD of all three polynomials.

The polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ are considered in the power basis, to eliminate scaling due to binomial terms in the entries of the subresultant matrices. Also, no noise is added to their coefficients. The polynomials are instead multiplied by scalars such that their GCD is unaltered but their subresultant matrices are badly scaled.

Consider the exact polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ of degrees $m = 17$, $n = 13$ and $o = 14$, whose factorisations are given by

$$\begin{aligned}
\hat{f}(x) &= (x-4.65)^2(x-1.5)^2(x-1.26)^4(x-1.1)^3(x-1)^4(x+3)^2 \\
\hat{g}(x) &= (x-4.99)^3(x-4.65)^2(x-2)(x-1.26)^4(x-1.1)^3 \\
\hat{h}(x) &= (x-4.65)^2(x-3.2)^2(x-1.26)^4(x-0.71)^4(x+1.75)^2,
\end{aligned}$$

and whose GCD $\hat{d}_t(x)$ of degree $t = 6$ is given by

$$\hat{d}(x) = (x-1.26)^4(x-4.65)^2.$$

The GCD of $\hat{f}(x)$ and $\hat{g}(x)$ is denoted $\hat{d}_a(x)$ of degree $t_a = 9$ and is given by

$$\hat{d}_a(x) = (x-4.65)^2(x-1.26)^4(x-1.1)^3.$$

The GCD $\hat{d}_b(x)$ of $\hat{f}(x)$ and $\hat{h}(x)$ is equal to the GCD $\hat{d}_c(x)$ of $\hat{g}(x)$ and $\hat{h}(x)$ and

$$\text{GCD}\left(\hat{f}(x), \hat{h}(x)\right) = \text{GCD}\left(\hat{g}(x), \hat{h}(x)\right) = \text{GCD}\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right) = \hat{d}_t(x).$$

As in Example 5.1.1, the polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ are scaled by 10^5, 10^5 and 10^{-5} respectively such that $f(x) = 10^5 \times \hat{f}(x)$, $g(x) = 10^5 \times \hat{g}(x)$ and $h(x) = 10^{-5} \times \hat{h}(x)$. The SVDs of the sets of subresultant matrices (i) $\{\tilde{S}_k(f(x), g(x), h(x))\}$, (ii) $\{\hat{S}_k(f(x), g(x), h(x))\}$, (iii) $\{\hat{S}_k(g(x), f(x), h(x))\}$ and (iv) $\{\hat{S}_k(h(x), g(x), f(x))\}$ are computed, and the singular values are plotted in Figures 5.1i to 5.1iv. In Figure 5.1ii and Figure 5.1iii the large separation between the numerically zero and non-zero singular values suggests that the degree of the GCD of $f(x)$, $g(x)$ and $h(x)$ is given by $t = 9$, but this is an incorrect result, and is due to the magnitude of the entries in the row-partition $\mathcal{R}_{a,k}$ being significantly larger than those in the row-partition $\mathcal{R}_{b,k}$.
The system of equations

$$\left[\; C_{n-k}\left(f(x)\right), \quad 0_{(m+n-k+1),(o-k)}, \quad C_{m-k}\left(g(x)\right) \;\right] \mathbf{x}_{k,1} = \mathbf{0}$$

has a non-trivial solution for $k = 1, \ldots, 9$ and the row-partition $\mathcal{R}_{a,k}$ is rank deficient for $k = 1, \ldots, 9$. The entries of the rows contained in this row-partition are significantly larger

than the entries in the coefficient matrix of

$$\left[\ 0_{(m+o-k+1),(n-k+1)},\quad C_{o-k}\left(f(x)\right),\quad C_{m-k}\left(h(x)\right)\ \right]\mathbf{x}_{k,1}=\mathbf{0},$$

which only has non-zero solutions for $k=1,\ldots,6$.

Let $r_{a,k}$, $r_{b,k}$ and $r_{c,k}$ be the ratios of entry of maximum magnitude to entry of minimum magnitude in the row-partitions $\mathcal{R}_{a,k}$, $\mathcal{R}_{b,k}$ and $\mathcal{R}_{c,k}$ respectively. The ratios in the first subresultant matrix $\tilde{S}_k(f(x),g(x),h(x))$ are given by

$$r_{a,1}=\frac{\max\{\mathcal{R}_{a,1}\}}{\min\{\mathcal{R}_{a,1}\}}=\frac{\max\{C_{n-k}(f(x)),C_{m-k}(g(x))\}}{\min\{C_{n-k}(f(x)),C_{m-k}(g(x))\}}=1.180983e+06 \tag{5.20}$$

$$r_{b,1}=\frac{\max\{\mathcal{R}_{b,1}\}}{\min\{\mathcal{R}_{b,1}\}}=\frac{\max\{C_{o-k}(f(x)),C_{m-k}(h(x))\}}{\min\{C_{o-k}(f(x)),C_{m-k}(h(x))\}}=7.437698e+15 \tag{5.21}$$

$$r_{c,1}=\frac{\max\{\mathcal{R}_{c,1}\}}{\min\{\mathcal{R}_{c,1}\}}=\frac{\max\{C_{o-k}(g(x)),C_{n-k}(h(x))\}}{\min\{C_{o-k}(g(x)),C_{n-k}(h(x))\}}=7.437698e+15. \tag{5.22}$$

The subresultant matrix $\hat{S}_k(h(x),g(x),f(x))$ consists of the row-partitions $\tilde{\mathcal{R}}_{b,k}$ and $\tilde{\mathcal{R}}_{c,k}$. The ratios of entry of maximum magnitude to entry of minimum magnitude of $\mathcal{R}_{b,1}$ and $\mathcal{R}_{c,1}$ are given by $r_{b,1}=7.437698e+15$ and $r_{c,1}=7.437698e+15$ respectively, and $\frac{r_{b,1}}{r_{c,1}}=1$. This indicates that the row-partitions are ideally scaled for the computation of the degree of the GCD. However, the entries of the two non-zero partitions in each of $\mathcal{R}_{a,1}$ and $\mathcal{R}_{b,1}$ are unbalanced. Preprocessing the subresultant matrices yields improved results and will be considered in Section 5.3. This example is then repeated for preprocessed polynomials in Section 5.5 (Example 5.5.1), where it will be shown that preprocessing has eliminated the poor scaling and the degree of the GCD is correctly identified.

□

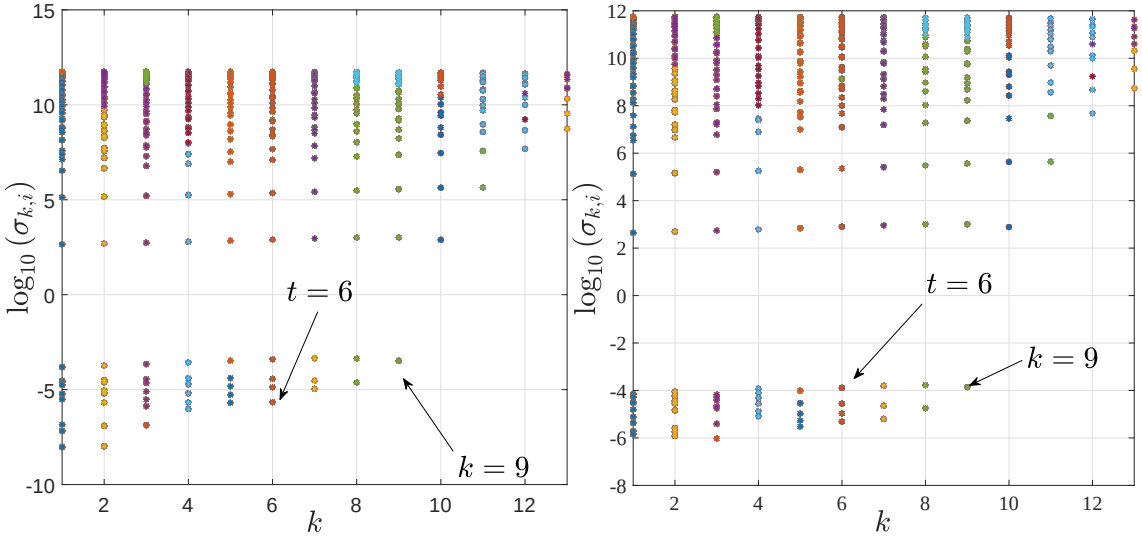
Example 5.1.3. Consider the Bernstein form of the exact polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ of degree $m=29$, $n=19$ and $o=18$ respectively, whose factorisations are given by

$$\begin{aligned}
\hat{f}(x)&=(x-9.2657984335)^2(x-1.2657984335)^4(x-0.41564897)^6(x-0.21657894)\times\\
&\quad(x-0.0654654561)^2(x+0.7879734)^9(x+1.654987654)^2(x+1.932654987)\times\\
&\quad(x+2.3549879)^2\\
\hat{g}(x)&=(x-9.2657984335)^2(x-1.75292)(x-1.2657984335)^4(x-0.99851354877)^3\times\\
&\quad(x-0.21657894)(x-0.0654654561)^2(x+0.1654988136)^4(x+1.654987654)^2\\
\hat{h}(x)&=(x-9.2657984335)^2(x-1.2657984335)^4(x-0.564987986958)^3(x-0.21657894)\times\\
&\quad(x-0.0654654561)^2(x+0.778912324654)^2(x+1.654987654)^2(x+1.75)^2
\end{aligned}$$

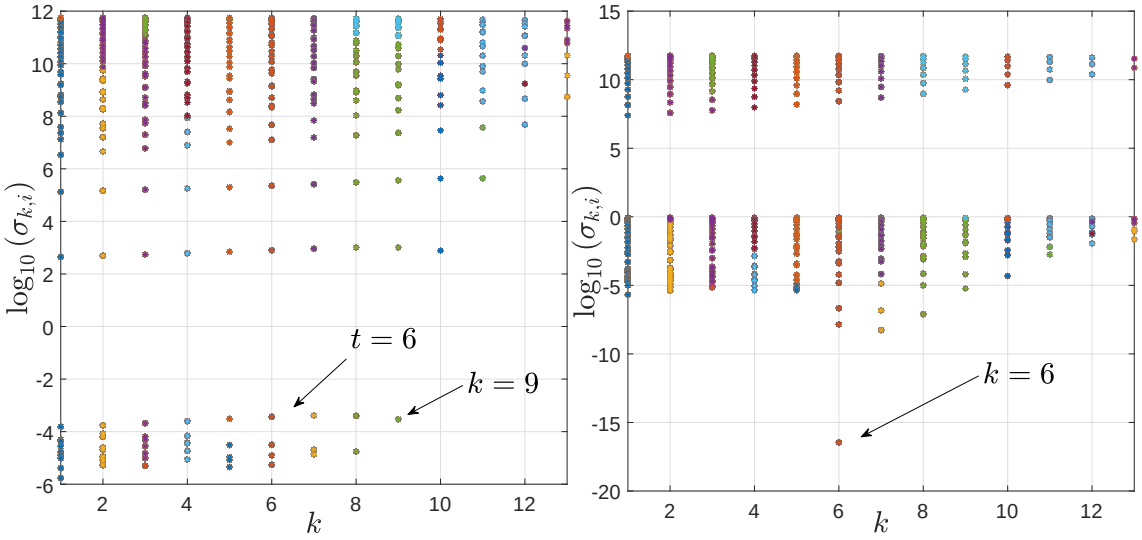
and whose GCD $\hat{d}_t(x)$ of degree $t=11$ is given in factorised form by

$$\begin{aligned}
\hat{d}_t(x)&=(x-9.2657984335)^2(x-1.2657984335)^4(x-0.21657894)\times\\
&\quad(x-0.0654654561)^2(x+1.654987654)^2.
\end{aligned}$$

Noise is added to the coefficients of $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ such that the coefficients of inexact



(i) $\{\tilde{S}_k(f(x), g(x), h(x))\}$ **(ii)** $\{\hat{S}_k(f(x), g(x), h(x))\}$



(iii) $\{\hat{S}_k(g(x), f(x), h(x))\}$ **(iv)** $\hat{S}_k(h(x), g(x), f(x))$

Figure 5.1: The singular values $\{\sigma_{k,i}\}$ of the unprocessed subresultant matrices (i) $\{\tilde{S}_k(f(x), g(x), h(x))\}$, (ii) $\{\hat{S}_k(f(x), g(x), h(x))\}$, (iii) $\{\hat{S}_k(g(x), f(x), h(x))\}$ and (iv) $\{\hat{S}_k(h(x), g(x), f(x))\}$ in Example 5.1.2

polynomials $f(x)$, $g(x)$ and $h(x)$ are given by

$$a_i = \hat{a}_i + \hat{a}_i \epsilon_{f,i} r_{f,i}, \quad b_j = \hat{b}_j + \hat{b}_j \epsilon_{g,j} r_{g,j} \quad \text{and} \quad c_p = \hat{c}_p + \hat{c}_p \epsilon_{h,p} r_{h,p}, \tag{5.23}$$

where $\{r_{f,i}\}$, $\{r_{g,j}\}$ and $\{r_{h,p}\}$ are uniformly distributed random variables in the interval $[-1, 1]$ and $\{\epsilon_{f,i}\} = \{\epsilon_{g,j}\} = \{\epsilon_{h,p}\} = 10^{-9}$ for $i = 0, \ldots, m$, $j = 0, \ldots, n$ and $p = 0, \ldots, o$. In Figure 5.2 the coefficients of $f(x)$, $g(x)$ and $h(x)$ are plotted. The degree of $f(x)$ is significantly higher than the degree of $g(x)$ and $h(x)$ and the span of the magnitude of the coefficients of $f(x)$ and $g(x)$ is greater than the span of the magnitudes of the coefficients of $h(x)$.

The singular values of the sets of (2×3) and (3×3) subresultant matrices (i) $\{\tilde{S}_k(f(x), g(x), h(x))\}$, (ii) $\{\hat{S}_k(f(x), g(x), h(x))\}$, (iii) $\{\hat{S}_k(g(x), h(x), f(x))\}$ and (iv) $\{\hat{S}_k(h(x), g(x), f(x))\}$ are plotted in Figures 5.3i to 5.3iv. The singular values

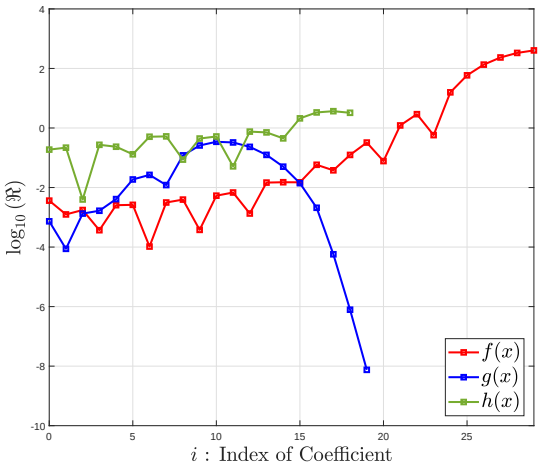
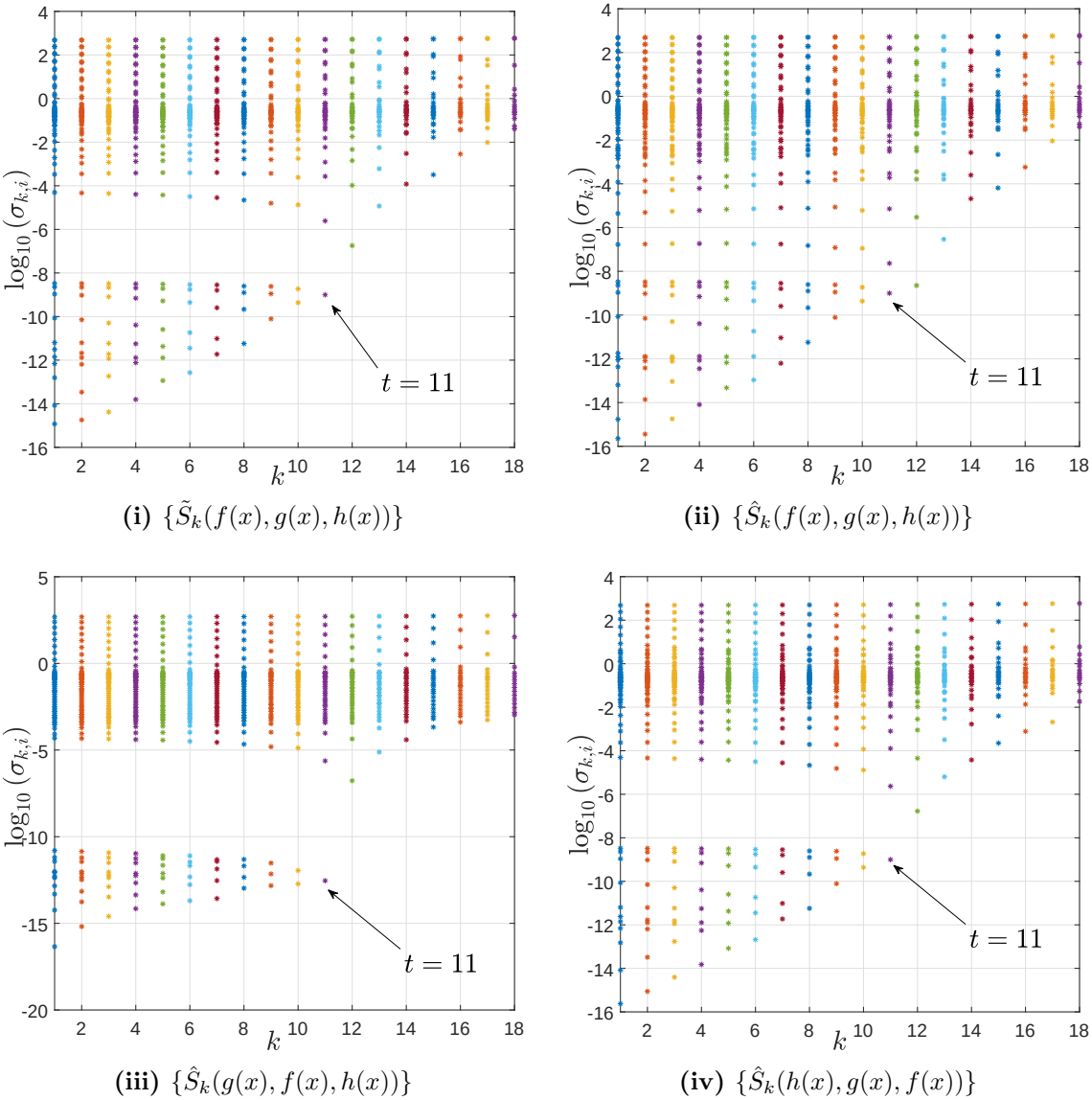


Figure 5.2: The coefficients of the polynomials $f(x)$, $g(x)$ and $h(x)$ in Example 5.1.3



(i) $\{\tilde{S}_k(f(x), g(x), h(x))\}$ **(ii)** $\{\hat{S}_k(f(x), g(x), h(x))\}$

(iii) $\{\hat{S}_k(g(x), f(x), h(x))\}$ **(iv)** $\{\hat{S}_k(h(x), g(x), f(x))\}$

Figure 5.3: The singular values $\{\sigma_{k,i}\}$ of the subresultant matrices (i)$\{\tilde{S}_k(f(x), g(x), h(x))\}$, (ii)$\{\hat{S}_k(f(x), g(x), h(x))\}$, (iii)$\{\hat{S}_k(g(x), f(x), h(x))\}$ and (iv)$\{\hat{S}_k(h(x), g(x), f(x))\}$ in Example 5.1.3

of the subresultant matrices $\{\hat{S}_k(g(x), f(x), h(x))\}$ (Figure 5.3iii) and to a lesser extent, those of $\{\hat{S}_k(h(x), g(x), f(x))\}$ (Figure 5.3iv) can be used to correctly deduce the degree of the GCD. The optimal variation of the subresultant matrices, with maximal separation amongst the numerically zero and non-zero singular values, is given by $\{\hat{S}_k(g(x), f(x), h(x))\}$.

The singular values of the (3×3) subresultant matrices (Figure 5.3i) reveal the degree of the GCD with a significant separation between the zero and non-zero singular values. However, the separation is far less significant than the separation of the singular values of $\{\hat{S}_k(g(x), f(x), h(x))\}$ in Figure 5.3iii.

The three pairwise GCDs are now considered, and the analysis of these gives some insight as to why certain variations of the (2×3) partitioned three-polynomial subresultant matrices cannot be used in the computation of the degree of the GCD. The computation of the degree of the pairwise GCDs (i) $d_a(x) = \text{GCD}(f(x), g(x))$, (ii) $d_b(x) = \text{GCD}(f(x), h(x))$ and (iii) $d_c(x) = \text{GCD}(g(x), h(x))$ are now considered. The singular values of the subresultant matrices (i) $\{ S_k(f(x), g(x)) \mid k = 1, \ldots \min(m, n) \}$, (ii) $\{ S_k(f(x), h(x)) \mid k = 1, \ldots, \min(m, o) \}$ and (iii) $\{ S_k(g(x), h(x)) \mid k = 1, \ldots, \min(n, o) \}$ are plotted in Figures 5.4i to 5.4iii. The degree of the GCD is correctly determined from the singular values of $\{S_k(g(x), h(x))\}$ in Figure 5.4iii, due to a large separation between the zero and non-zero singular values. There is a small but significant separation between the numerically zero and non-zero singular values of $\{S_k(f(x), g(x))\}$, such that the degree of the GCD can be recovered. Yet in Figure 5.4ii the separation between the zero and non-zero singular values is indicative of the degree of the GCD being given by $t = 14$, but this is incorrect and is possibly due to a factor of $f(x)$ which is similar to a factor of $g(x)$.

These three pairwise GCDs reveal why the degree of the GCD can be obtained using one variant of the (2×3) subresultant and why another variant returns an incorrect result.

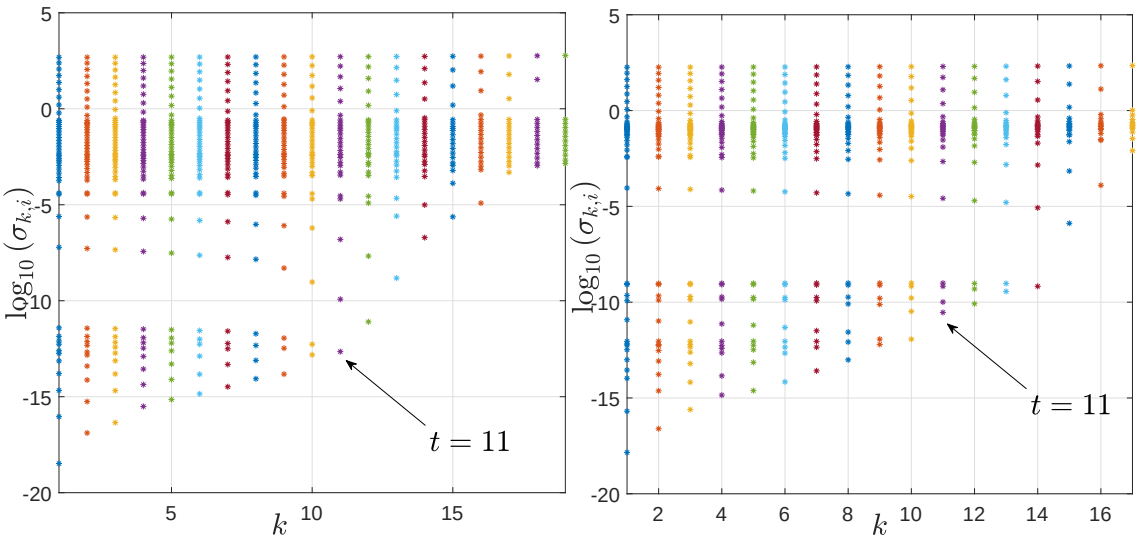
For instance, the kth subresultant matrix in the set of (2×3) partitioned subresultant matrices $\{\hat{S}_k(g(x), f(x), h(x))\}$ contains the row-partitions $\mathcal{R}_{a,k}$ and $\mathcal{R}_{c,k}$. These are derived from the two-polynomial subresultant matrices used in the computation of the degree of the GCD of $f(x)$ and $g(x)$, and the degree of the GCD of $g(x)$ and $h(x)$.

From Figures 5.4i and 5.4iii, these GCD computations are well defined and the numerically zero and non-zero singular values are well separated. However, the degree of the GCD of $f(x)$ and $h(x)$ is incorrectly determined and therefore the (2×3) partitioned subresultant matrices which include the row-partition $\mathcal{R}_{b,k}$ are less reliable in the computation of the GCD degree, as shown in Figure 5.3ii.

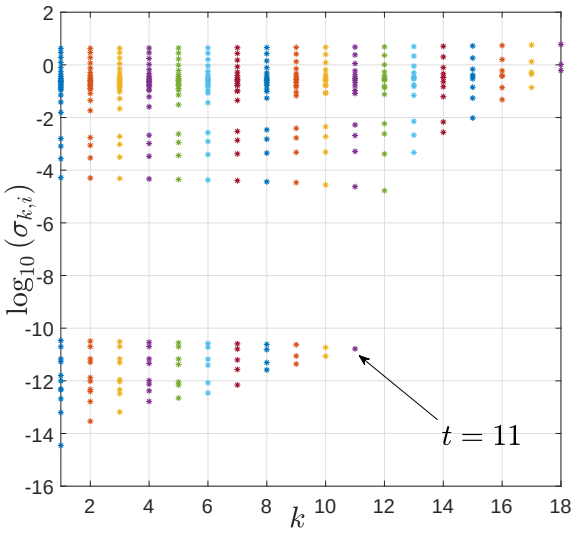
□

In this section the optimal ordering of the three polynomials in the (2×3) partitioned subresultant matrices has been discussed. It was shown by example that good scaling of the row-partitions is necessary to reliably compute the degree of the GCD of three polynomials. Examples have shown that it is possible that the degree of the GCD can be correctly computed from one variation of the (2×3) subresultant matrix sequence while being incorrectly computed from an alternative ordering of polynomials $f(x)$, $g(x)$ and $h(x)$.

The GCD of three polynomials has also been considered by computing pairwise GCDs,



(i) $\{S_k(f(x), g(x))\}$ **(ii)** $\{S_k(f(x), h(x))\}$



(iii) $\{S_k(g(x), h(x))\}$

Figure 5.4: The singular values $\{\sigma_{k,i}\}$ of the unprocessed subresultant matrices (i) $\{S_k(f(x), g(x))\}$, (ii) $\{S_k(f(x), h(x))\}$ and (iii) $\{S_k(g(x), h(x))\}$ in Example 5.1.3

and it was shown by example that for three polynomials $f(x)$, $g(x)$ and $h(x)$ with a common divisor $d(x)$, the computation of the degree of the pairwise GCD of $f(x)$ and $g(x)$ may return an incorrect result where computing the degree of the GCD of all three polynomials simultaneously returns the correct result.

It has also been shown that in the computation of the degree of the GCD of three polynomials it is only necessary to consider two of the three equations (5.1,5.2,5.3). Therefore, the (2×3) partitioned subresultant matrices are sufficient to describe the complete system of three equations, but these two equations must be chosen with consideration.

5.2 Optimal Variants of the (2×3) and (3×3) Partitioned Subresultant Matrices

In Section 3.1.4 the five variants of the two-polynomial subresultant matrices were considered, where each variant was defined by the inclusion or exclusion of matrices D^{-1}_{m+n-k} and $\hat{Q}_k$. In this section the optimal variant of the three-polynomial subresultant matrices must be considered. These are defined in a similar way to the variants of the two-polynomial subresultant matrices in Section 3.1.4, but each of these variants must be considered for both the (2×3) and (3×3) partitioned variants.

The variants of the kth (3×3) partitioned subresultant matrix, given in (5.6), are defined in terms of matrices $\tilde{D}_k^{-1}$, $\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ and $\tilde{Q}_k$. Therefore, the four variants are given by (i) $\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$, (ii) $\tilde{D}_k^{-1}\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$, (iii) $\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k$ and (iv) $\tilde{D}_k^{-1}\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k = \tilde{S}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$.

The kth (2×3) partitioned kth subresultant matrix has a similar definition and is given by

$$\hat{S}_k\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right) = \hat{D}_k^{-1}\hat{T}_k\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right)\tilde{Q}_k. \tag{5.24}$$

The block diagonal matrix $\hat{D}_k^{-1}$ of order $(2m + n + o - 2k)$ is given by

$$\hat{D}_k^{-1} = \text{diag}\left[\begin{array}{cc} D^{-1}_{m+n-k}, & D^{-1}_{m+o-k} \end{array}\right], \tag{5.25}$$

where matrices D^{-1}_{m+n-k} and D^{-1}_{m+o-k} are of the same structure as the matrix in (2.10). The matrix $\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ is given by

$$\left[\begin{array}{ccc} T_{n-k}\left(\hat{f}(x)\right) & & T_{m-k}\left(\hat{g}(x)\right) \\ & T_{o-k}\left(\hat{f}(x)\right) & T_{m-k}\left(\hat{h}(x)\right) \end{array}\right], \tag{5.26}$$

where the partitions of the form $T_{n-k}(\hat{f}(x)) \in \mathbb{R}^{(m+n-k)\times(n-k+1)}$ are Toeplitz matrices and have the same structure as the matrix in (2.11). The matrix $\tilde{Q}_k$ in (5.24) is given by

$$\tilde{Q}_k = \text{diag}\left[\begin{array}{ccc} Q_{n-k}, & Q_{o-k}, & Q_{m-k} \end{array}\right].$$

Similar to the (2×3) partitioned matrices, the (2×3) partitioned subresultant matrices have four variants defined by the inclusion or exclusion of $\hat{D}_k^{-1}$ and $\tilde{Q}_k$, and these are given by (i) $\{\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\}$, (ii) $\{\hat{D}_k^{-1}\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\}$, (iii) $\{\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k\}$ and (iv) $\{\hat{D}_k^{-1}\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k\}$.

The following example shows how the sets of coefficient multipliers in the different variants of the subresultant matrices can cause significant scaling issues amongst the row and column-partitions.

Example 5.2.1. Consider the polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ of degrees $m = 5$, $n = 15$ and $o = 7$. This example considers the scaling effect of the coefficient multipliers for the (2×3) and (3×3) partitioned subresultant matrices $\hat{S}_3(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ and $\tilde{S}_3(\hat{f}(x), \hat{g}(x), \hat{h}(x))$.

Figures 5.5i to 5.5iv show heat maps of the coefficient multipliers (on a logarithmic

scale) of the entries of the four (2×3) partitioned subresultant matrix variants. First consider the coefficient multipliers in the variant given by $T_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ shown in Figure 5.5i. The non-zero entries of the top right partition are significantly larger than the entries in the three remaining non-zero partitions. This is because the degree of $\hat{g}(x)$ is significantly larger than the degree of $\hat{f}(x)$ or $\hat{h}(x)$, so coefficients of $\hat{g}(x)$ appearing in the top right partition $T_{m-k}(\hat{g}(x))$ are multiplied by $\binom{n}{i} = \binom{15}{i}$, which has a maximum of $\binom{15}{7} = 6435$, while the coefficient multipliers of $T_{n-k}(\hat{f}(x))$ and $T_{m-k}(\hat{h}(x))$ are at most equal to $\binom{5}{2} = 10$ and $\binom{7}{3} = 35$.

Scaling due to coefficient multipliers is also unbalanced in $\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k$ as shown in Figure 5.5iii. The coefficient multipliers in the second row-partition $\mathcal{R}_{b,k}$ are of significantly smaller magnitude than those in the first row-partition $\mathcal{R}_{a,k}$, and again this is due to n being significantly larger than m and o.

The variant $\hat{D}_k^{-1}\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ also suffers from poor scaling. The top left partition of this subresultant matrix contains entries in its middle rows which are significantly smaller than those in the other rows.

Figure 5.6 shows a heat map of the coefficient multipliers in the entries of the four subresultant variants where the subresultant matrix has a (3×3) partitioned structure, and again the same arguments are used to suggest that the optimal variant is given by $\tilde{D}_k^{-1}\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k$.

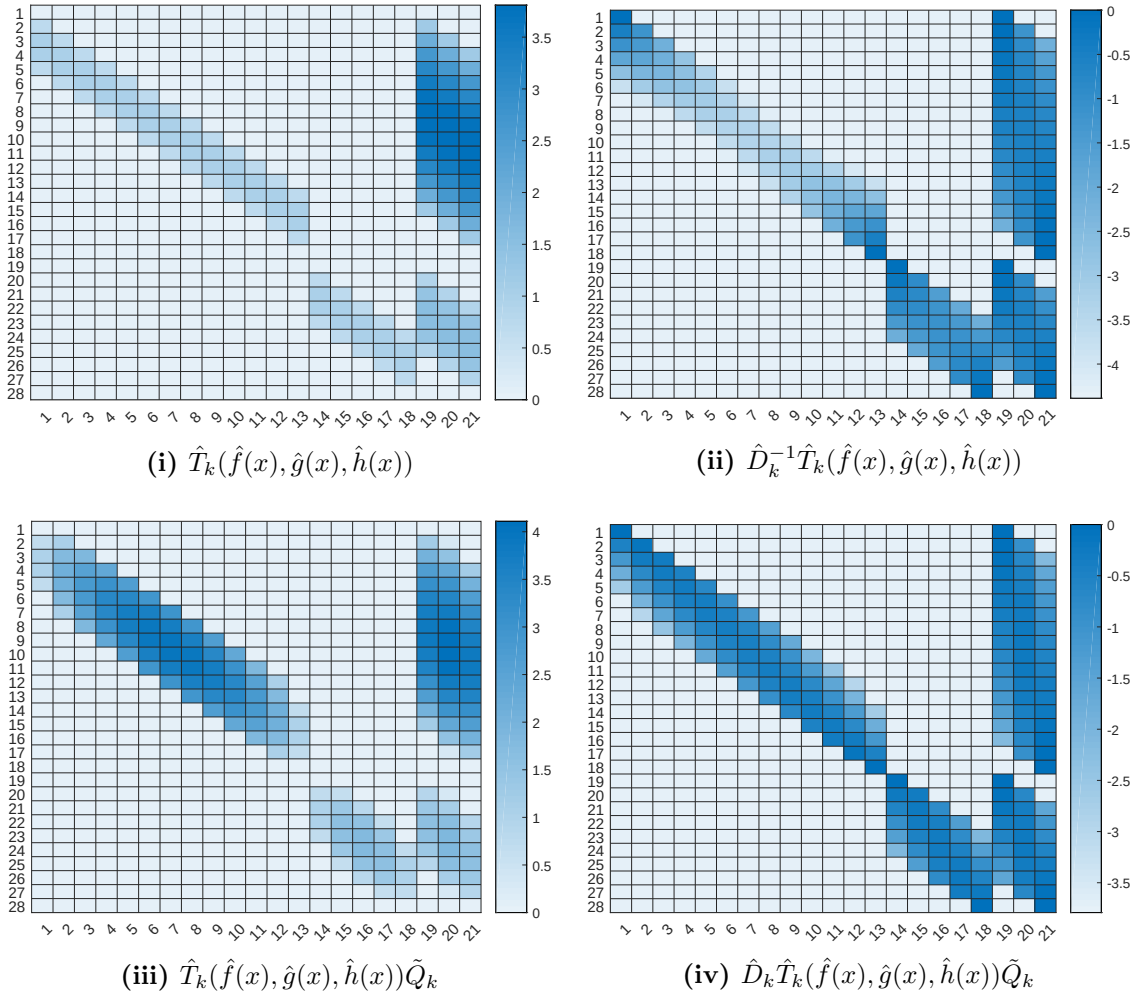
□

Example 5.1.2 showed that poorly scaled polynomials give incorrect results for the computation of the degree of the GCD. The combination of binomial terms in the non-zero entries of the sets of subresultant matrices T_k, $D_k^{-1}T_k$, T_kQ_k and $D_k^{-1}T_kQ_k$[ii], for both (2×3) and (3×3) partitioned forms, can cause further scaling problems. As such, it is necessary to consider the variant which has the optimal scaling amongst its entries. The heat maps of the entries of the matrices $\hat{S}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x)) = \hat{D}_k^{-1}\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k$ and $\tilde{S}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x)) = \tilde{D}_k^{-1}\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k$ suggest these variants are optimal for the (2×3) and (3×3) subresultant matrices respectively. This outcome is consistent with the variants of the two-polynomial subresultant matrices.

By Example 5.2.1 it is shown that the optimal subresultant variant of the (2×3) subresultant matrices is given by $\hat{D}_k^{-1}\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k$, where $\hat{D}_k^{-1}$ and $\tilde{Q}_k$ have the effect of dividing the rows and multiplying the columns by binomial terms such that the entries in the partitions of the subresultant matrices contain the coefficients of $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ multiplied by a scalar in the unit interval. Similarly, the variant $\tilde{D}_k^{-1}\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k$ is the optimal variant of the (3×3) partitioned subresultant matrices.

As seen in Example 5.1.2, the entries in the set of subresultant matrices of the form $\{D_k^{-1}T_kQ_k\}$ for both (2×3) and (3×3) variants may still be badly scaled. Preprocessing the thee-polynomial subresultant matrices goes some way to mitigating the effect of poor scaling and this will be discussed in Section 6.4.

[ii] These are loosely defined terms which refer to the four subresultant matrix variants.



(i) $\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ **(ii)** $\hat{D}_k^{-1}\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$

(iii) $\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k$ **(iv)** $\hat{D}_k\hat{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k$

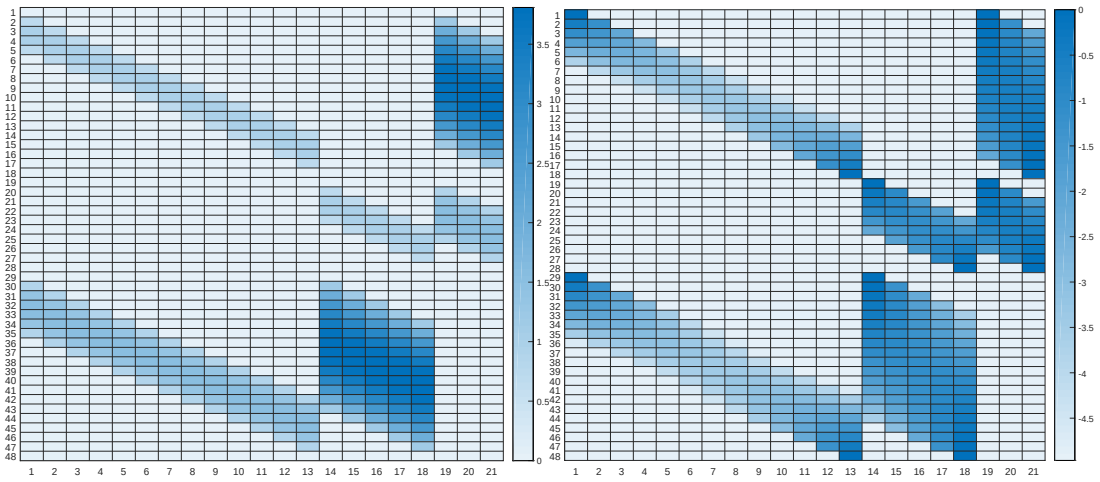
Figure 5.5: Heat map of the coefficient multipliers in the entries of the four (2×3) subresultant matrix variants in Example 5.2.1

5.3 Preprocessing the Three-Polynomial Subresultant Matrices

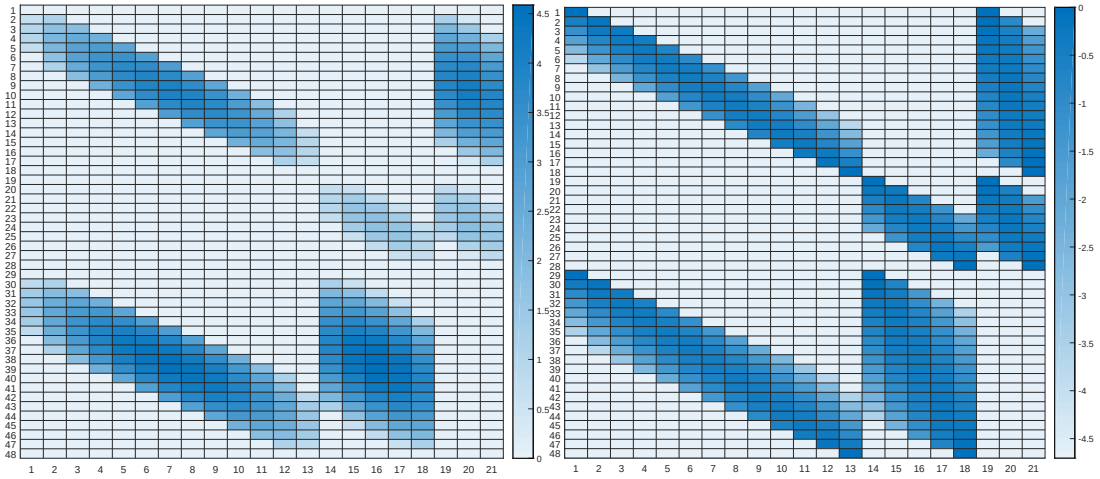
In Section 3.4 it was shown that preprocessing the set of two-polynomial subresultant matrices yields improved results in the computation of both the degree and coefficients of the GCD. These preprocessing operations can be extended to preprocess the three-polynomial subresultant matrices which have been defined in this section.

Firstly, the polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ are normalised by the geometric mean of their respective entries in $\hat{S}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ or $\hat{S}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$. When normalising $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ by the geometric mean of their entries in the (3×3) partitioned matrix, all three polynomials are normalised by the geometric mean of the non-zero entries in two partitions each. The normalised polynomials $\bar{f}_k(x)$, $\bar{g}_k(x)$ and $\bar{h}_k(x)$ are given by

$$\bar{f}(x) = \frac{\hat{f}(x)}{\hat{\mathcal{G}}_k\left(\hat{f}(x)\right)}, \quad \hat{g}(x) = \frac{\hat{g}(x)}{\hat{\mathcal{G}}_k\left(\hat{g}(x)\right)} \quad \text{and} \quad \hat{h}(x) = \frac{\hat{h}(x)}{\hat{\mathcal{G}}_k\left(\hat{h}(x)\right)},$$



(i) $\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ **(ii)** $\tilde{D}_k^{-1}\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$



(iii) $\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k$ **(iv)** $\tilde{D}_k^{-1}\tilde{T}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))\tilde{Q}_k$

Figure 5.6: Heat map of the coefficient multipliers in the entries of the four (3×3) partitioned subresultant matrix variants in Example 5.2.1

where

$$\hat{\mathcal{G}}_k\left(\hat{f}(x)\right) = \left(\prod_{j=0}^{n-k}\prod_{i=0}^{m}\frac{\hat{a}_i\binom{m}{i}\binom{n-k}{j}}{\binom{m+n-k}{i+j}} \times \prod_{p=0}^{o-k}\prod_{i=0}^{m}\frac{\hat{a}_i\binom{m}{i}\binom{o-k}{p}}{\binom{m+o-k}{i+p}}\right)^{\frac{1}{(m+1)(n+o-2k+2)}}. \tag{5.27}$$

The new notation $\hat{\mathcal{G}}_k(\hat{f}(x))$ is used to denote the geometric mean of the polynomial $\hat{f}(x)$ in the two partitions $C_{n-k}(\hat{f}(x))$ and $C_{o-k}(\hat{f}(x))$.

In the (2×3) partitioned subresultant matrices $\hat{S}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ only the coefficients of the polynomial $\hat{f}(x)$ appear in two partitions, namely, $C_{n-k}(\hat{f}(x))$ and $C_{o-k}(\hat{f}(x))$, and the geometric mean of the non-zero entries in these two partitions is given by $\hat{\mathcal{G}}_k(\hat{f}(x))$ which is defined in (5.27). Coefficients of the polynomials $\hat{g}(x)$ and $\hat{h}(x)$ are normalised by $\mathcal{G}_{m-k}(\hat{g}(x))$ and $\mathcal{G}_{m-k}(\hat{h}(x))$ respectively, where these expressions are of the same form as (3.33).

5.3.1 The Minimisation Problem

The second and third preprocessing operations described in Section 3.4 scaled the second partition of the kth two-polynomial subresultant matrix by α_k and replaced the independent variable x with $\theta_k\omega$, where α_k and θ_k were optimally chosen to minimise the ratio of entry of maximum magnitude of entry of minimum magnitude in the kth subresultant matrix.

The method for preprocessing the (2×3) partitioned three-polynomial subresultant matrix $\hat{S}_k(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ is now described, and this is easily extended to consider the two other variations $\hat{S}_k(\hat{g}(x), \hat{f}(x), \hat{h}(x))$ and $\hat{S}_k(\hat{h}(x), \hat{g}(x), \hat{f}(x))$ given by alternative polynomial orderings.

The independent variable x is again replaced by $\theta\omega$ and polynomials $\hat{f}(x)$ and $\hat{h}(x)$ are scaled by λ and ρ respectively. The optimal values of λ, ρ and θ are to be determined for each subresultant matrix.

The polynomials which are to be optimised are given by

$$\begin{aligned} \lambda\ddot{f}(\omega,\theta) &= \lambda\sum_{i=0}^{m}\bar{a}_i\theta^i\binom{m}{i}(1-\theta\omega)^{m-i}\omega^i, \\ \ddot{g}(\omega,\theta) &= \sum_{i=0}^{n}\bar{b}_i\theta^i\binom{n}{i}(1-\theta\omega)^{n-i}\omega^i, \\ \text{and} \qquad \rho\ddot{h}(\omega,\theta) &= \rho\sum_{i=0}^{o}\bar{c}_i\theta^i\binom{o}{i}(1-\theta\omega)^{o-i}\omega^i. \end{aligned} \tag{5.28}$$

The unprocessed (2×3) subresultant matrix is given by

$$\hat{S}_k(\lambda\ddot{f}(\theta,\omega), \ddot{g}(\theta,\omega), \rho\ddot{h}(\theta,\omega)) = \begin{bmatrix} C_{n-k}\left(\lambda\ddot{f}(\theta,\omega)\right) & 0 & C_{m-k}\left(\ddot{g}(\theta,\omega)\right) \\ 0 & C_{o-k}\left(\lambda\ddot{f}(\theta,\omega)\right) & C_{m-k}\left(\rho\ddot{h}(\theta,\omega)\right) \end{bmatrix}. \tag{5.29}$$

The optimal values λ_k, ρ_k, and θ_k of λ, ρ and θ are sought such that the ratio of entry of maximum magnitude to entry of minimum magnitude is minimised in the kth subresultant matrix.

It is convenient to define the sets of all non-zero entries in each of the partitions $C_{n-k}(\lambda\ddot{f}(\theta,\omega))$, $C_{m-k}(\ddot{g}(\theta,\omega))$, $C_{o-k}(\lambda\ddot{f}(\theta,\omega))$ and $C_{m-k}(\rho\ddot{h}(\theta,\omega))$. Let the sets of non-zero entries of these partitions be given by $\mathscr{P}_{1,k}(\lambda,\theta)$, $\mathscr{P}_{2,k}(\theta)$, $\mathscr{P}_{3,k}(\lambda,\theta)$ and $\mathscr{P}_{4,k}(\rho,\theta)$,

where

$$\mathscr{P}_{1,k}\left(\lambda,\theta\right)=\left\{\frac{\left|\lambda\bar{a}_i\theta^i\binom{m}{i}\binom{n-k}{j}\right|}{\binom{m+n-k}{i+j}}\right\}\qquad i=0,\ldots,m;\ j=0,\ldots,n-k,$$

$$\mathscr{P}_{2,k}\left(\theta\right)=\left\{\frac{\left|\bar{b}_i\theta^i\binom{n}{i}\binom{m-k}{j}\right|}{\binom{m+n-k}{i+j}}\right\}\qquad i=0,\ldots,n;\ j=0,\ldots,m-k,$$

$$\mathscr{P}_{3,k}\left(\lambda,\theta\right)=\left\{\frac{\left|\lambda\bar{a}_i\theta^i\binom{m}{i}\binom{o-k}{j}\right|}{\binom{m+o-k}{i+j}}\right\}\qquad i=0,\ldots,m;\ j=0,\ldots,o-k,$$

$$\mathscr{P}_{4,k}\left(\rho,\theta\right)=\left\{\frac{\left|\rho\bar{c}_i\theta^i\binom{o}{i}\binom{m-k}{j}\right|}{\binom{m+o-k}{i+j}}\right\}\qquad i=0,\ldots,o;\ j=0,\ldots,m-k,$$

such that the minimisation problem is given by

$$(\lambda_k,\rho_k,\theta_k)=\arg\min_{\lambda,\rho,\theta}\left\{\frac{\max\{\max\{\mathscr{P}_{1,k}(\lambda,\theta)\},\max\{\mathscr{P}_{2,k}(\theta)\},\max\{\mathscr{P}_{3,k}(\lambda,\theta)\},\max\{\mathscr{P}_{4,k}(\rho,\theta)\}\}}{\min\{\min\{\mathscr{P}_{1,k}(\lambda,\theta)\},\min\{\mathscr{P}_{2,k}(\theta)\},\min\{\mathscr{P}_{3,k}(\lambda,\theta)\},\min\{\mathscr{P}_{4,k}(\rho,\theta)\}\}}\right\}.$$

The optimal values λ_k, ρ_k and θ_k are computed as solutions of the linear programming problem which is similar to that used for preprocessing the two-polynomial subresultant matrices in Section 3.4. A full description of this linear programming problem is found in Appendix C.1.1 and the preprocessed polynomials $\lambda_k\tilde{f}_k(\omega)$, $\tilde{g}_k(\omega)$ and $\rho_k\tilde{h}_k(\omega)$ are given by

$$\lambda_k\tilde{f}_k(\omega)=\lambda_k\sum_{i=0}^{m}\bar{a}_i\theta_k^i\binom{m}{i}(1-\theta_k\omega)^{m-i}\omega^i,$$

$$\tilde{g}_k(\omega)=\sum_{i=0}^{n}\bar{b}_i\theta_k^i\binom{n}{i}(1-\theta_k\omega)^{n-i}\omega^i,$$

$$\rho_k\tilde{h}_k(\omega)=\rho_k\sum_{i=0}^{o}\bar{c}_i\theta_k^i\binom{o}{i}(1-\theta_k\omega)^{o-i}\omega^i.$$

A trivial extension is required to compute the optimal values λ_k, ρ_k, and θ_k for preprocessing the polynomials of the (3×3) subresultant matrices. The sets $\mathscr{P}_{1,k}(\lambda,\theta)$, $\mathscr{P}_{2,k}(\theta)$, $\mathscr{P}_{3,k}(\lambda,\theta)$ and $\mathscr{P}_{4,k}(\rho,\theta)$ are defined above. Now, the sets $\mathscr{P}_{5,k}(\rho,\theta)$ and $\mathscr{P}_{6,k}(\theta)$ are defined to account for the additional row-partition in the (3×3) partitioned subresultant matrices. These are given by

$$\mathscr{P}_{5,k}\left(\lambda,\theta\right)=\left\{\frac{\left|\rho\bar{c}_i\theta^i\binom{o}{i}\binom{n-k}{j}\right|}{\binom{n+o-k}{i+j}}\right\}\qquad i=0,\ldots,o;\ j=0,\ldots,n-k,$$

$$\mathscr{P}_{6,k}\left(\theta\right)=\left\{\frac{\left|\bar{b}_i\theta^i\binom{n}{i}\binom{o-k}{j}\right|}{\binom{n+o-k}{i+j}}\right\}\qquad i=0,\ldots,n;\ j=0,\ldots,o-k.$$

The extended minimisation problem is written as

$$(\lambda_k, \rho_k, \theta_k) = \arg\min_{\lambda,\rho,\theta} \left\{ \frac{\max\{\max\{\mathscr{P}_{1,k}(\lambda,\theta)\}, \max\{\mathscr{P}_{2,k}(\theta)\}, \max\{\mathscr{P}_{3,k}(\lambda,\theta)\}, \max\{\mathscr{P}_{4,k}(\rho,\theta)\}, \max\{\mathscr{P}_{5,k}(\rho,\theta)\}, \max\{\mathscr{P}_{6,k}(\theta)\}\}}{\min\{\min\{\mathscr{P}_{1,k}(\lambda,\theta)\}, \min\{\mathscr{P}_{2,k}(\theta)\}, \min\{\mathscr{P}_{3,k}(\lambda,\theta)\}, \min\{\mathscr{P}_{4,k}(\rho,\theta)\}, \min\{\mathscr{P}_{5,k}(\rho,\theta)\}, \min\{\mathscr{P}_{6,k}(\theta)\}\}} \right\},$$

where the optimal values of λ_k, ρ_k, and θ_k are the solutions of a linear programming problem similar to the one defined for the (2×3) partitioned subresultant matrix, with an additional set of constraints.

The degree and coefficients of the GCD are more reliably approximated from the sequence of the preprocessed subresultant matrices $\{\hat{S}_k(\lambda_k \tilde{f}_k(\omega), \tilde{g}_k(\omega), \rho_k \tilde{h}_k(\omega))\}$ than the unprocessed forms due to improved scaling of the entries. Examples will show that due to the inclusion of preprocessing, the separation between the zero and non-zero singular values is increased, and the approximations of cofactor polynomials are significantly better. Examples are given in Section 5.5, but first the method of approximating the coefficients of cofactor polynomials and the GCD is considered.

5.4 Approximating the Coefficients of Cofactor Polynomials and the GCD

In Section 3.5.1 the computation of approximations of the cofactor polynomials $\hat{u}_t(x)$ and $\hat{v}_t(x)$ and the GCD $\hat{d}_t(x)$ was considered. This section describes the computation of approximations of the three cofactor polynomials $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{w}_t(x)$ and the GCD $\hat{d}_t(x)$ using a similar least squares based method.

In this section the definitions of polynomials f, g and h are deliberately not specified as they may represent (i) inexact polynomials $f(x)$, $g(x)$ and $h(x)$ or (ii) preprocessed polynomials $\lambda_t \tilde{f}_t(\omega)$, $\tilde{g}_t(\omega)$ and $\rho_t \tilde{h}_t(\omega)$.

The tth (2×3) subresultant matrix $\hat{S}_t(f, g, h)$ and the tth (3×3) subresultant matrix $\tilde{S}_t(f, g, h)$ are numerically rank deficient, and therefore there exist non-zero vectors $\hat{\mathbf{x}}_t$ and $\tilde{\mathbf{x}}_t$ such that

$$\tilde{S}_t(f, g, h)\, \tilde{\mathbf{x}}_t \approx \mathbf{0} \quad \text{and} \quad \hat{S}_t(f, g, h)\, \hat{\mathbf{x}}_t \approx \mathbf{0}. \tag{5.30}$$

One of the columns of each of the subresultant matrices $\tilde{S}_t(f, g, h)$ and $\hat{S}_t(f, g, h)$ lies in the space spanned by the remaining columns (with a minimal residual) and these columns are denoted $\tilde{\mathbf{c}}_{t,q}$ and $\hat{\mathbf{c}}_{t,q}$ respectively. The remaining columns are denoted $\tilde{A}_{t,q}(f, g, h)$ and $\hat{A}_{t,q}(f, g, h)$

$$\tilde{A}_{t,q}(f, g, h)\tilde{\mathbf{x}}_{t,q} \approx \mathbf{c}_{t,q} \quad \text{and} \quad \hat{A}_{t,q}(f, g, h)\hat{\mathbf{x}}_{t,q} \approx \mathbf{c}_{t,q},$$

where the vectors $\tilde{\mathbf{x}}_{t,q}$ and $\hat{\mathbf{x}}_{t,q}$ can be computed by the standard least squares method. The vectors $\tilde{\mathbf{x}}_t$ and $\hat{\mathbf{x}}_t$ in (5.30) are given by the insertion of '-1' into the qth position of both $\tilde{\mathbf{x}}_{t,q}$ and $\hat{\mathbf{x}}_{t,q}$.

The vectors $\tilde{\mathbf{x}}_t$ and $\hat{\mathbf{x}}_t$ contain the coefficients of the polynomials u_t, v_t and w_t and are

given by

$$\tilde{\mathbf{x}}_t = \hat{\mathbf{x}}_t = \begin{bmatrix} \mathbf{v}_t, & \mathbf{w}_t, & -\mathbf{u}_t \end{bmatrix}^T.$$

The coefficients of the approximation of the AGCD are given by the solution to

$$\begin{bmatrix} C_t(u_t) \\ C_t(v_t) \\ C_t(w_t) \end{bmatrix} \mathbf{d}_t \approx \begin{bmatrix} \mathbf{f} \\ \mathbf{g} \\ \mathbf{h} \end{bmatrix}. \tag{5.31}$$

It was stated that f, g and h were deliberately not specified, and different definitions of these polynomials give different approximations:

1. If f, g, and h represent the unprocessed polynomials $f(x)$, $g(x)$ and $h(x)$, then the vectors $\mathbf{u}_t$, $\mathbf{v}_t$, $\mathbf{w}_t$ and $\mathbf{d}_t$ contain the coefficients of the approximations $u_t(x)$, $v_t(x)$, $w_t(x)$ and $d_t(x)$.

2. Otherwise, if f, g and h represent the preprocessed polynomials $\tilde{f}_t(\omega)$, $\tilde{g}_t(\omega)$ and $\tilde{h}_t(\omega)$, then the vectors $\mathbf{u}_t$, $\mathbf{v}_t$, $\mathbf{w}_t$ and $\mathbf{d}_t$ contain the coefficients of $\tilde{u}_t(\omega)$, $\tilde{v}_t(\omega)$, $\tilde{w}_t(\omega)$ and $\tilde{d}_t(\omega)$ from which the polynomials $\tilde{u}_t(x)$, $\tilde{v}_t(x)$, $\tilde{w}_t(x)$ and $\tilde{d}_t(x)$ are obtained by a change of variable $\omega = x/\theta$.

The errors in the approximations $u_t(x)$, $v_t(x)$, $w_t(x)$ and $d_t(x)$ obtained from the unprocessed polynomials $f(x)$, $g(x)$ and $h(x)$ are defined in (3.64), while the errors in the approximations $\tilde{u}_t(x)$, $\tilde{v}_t(x)$, $\tilde{w}_t(x)$ and $\tilde{d}_t(x)$ obtained from the preprocessed polynomials $\tilde{f}_t(\omega)$, $\tilde{g}_t(\omega)$ and $\tilde{h}_t(\omega)$ are defined in (3.65).

5.5 Results

Example 5.5.1. This example returns to the set of polynomials $f(x)$, $g(x)$ and $h(x)$ defined in Example 5.1.2. Scaling was introduced such that the degree of the AGCD was incorrectly determined. The polynomials $f(x)$, $g(x)$ and $h(x)$ are now preprocessed to obtain the sets of scaled polynomials $\{\lambda_k \tilde{f}_k(\omega)\}$, $\{\tilde{g}_k(\omega)\}$ and $\{\rho_k \tilde{h}_k(\omega)\}$. The coefficients of the unprocessed and preprocessed polynomials $\lambda_1 \tilde{f}_1(\omega)$, $\tilde{g}_1(\omega)$ and $\rho_1 \tilde{h}_1(\omega)$ are plotted in Figure 5.7. The coefficients of the unprocessed forms span many orders of magnitude (due to the scaling introduced at the beginning of the example), while the coefficients of the preprocessed polynomials all span a similar range.

Preprocessing is applied to the three sets of (2×3) partitioned subresultant matrices and the single set of (3×3) partitioned subresultant matrices. These sets are given by (i) $\{\hat{S}_k(f(x), g(x), h(x))\}$, (ii) $\{\hat{S}_k(g(x), f(x), h(x))\}$, (iii) $\{\hat{S}_k(h(x), g(x), f(x))\}$ and (iv) $\{\tilde{S}_k(f(x), g(x), h(x))\}$.

The SVDs of the subresultant matrices of the preprocessed polynomials are computed, and the sets of singular values $\{\sigma_{k,i}\}$ of the three subresultant matrix variations (i) $\{\tilde{S}_k(\lambda_k \tilde{f}_k(\omega), \tilde{g}_k(\omega), \rho_k \tilde{h}_k(\omega))\}$, (ii) $\{S_k(\lambda_k \tilde{f}_k(\omega), \tilde{g}_k(\omega), \rho_k \tilde{h}_k(\omega))\}$, (iii) $\{S_k(\tilde{g}_k(\omega), \lambda_k \tilde{f}_k(\omega), \rho_k \tilde{h}_k(\omega))\}$ and (iv) $\{S_k(\rho_k \tilde{h}_k(\omega), \tilde{g}_k(\omega), \lambda_k \tilde{f}_k(\omega))\}$ are plotted in Figures 5.8i to 5.8iv. The separation of the numerically zero and non-zero singular values indicates that the degree of the AGCD is given by $t = 6$ for each of the three variations

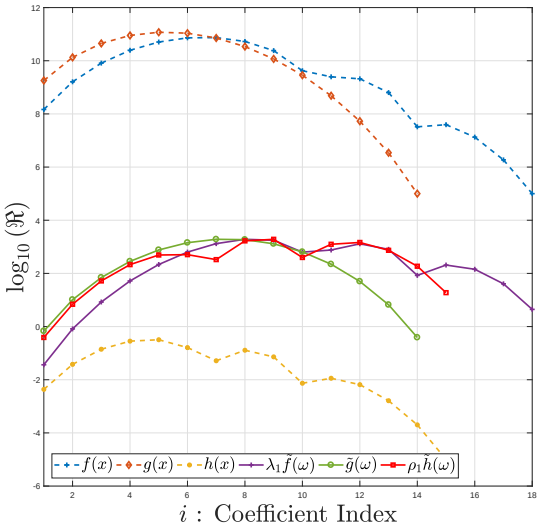
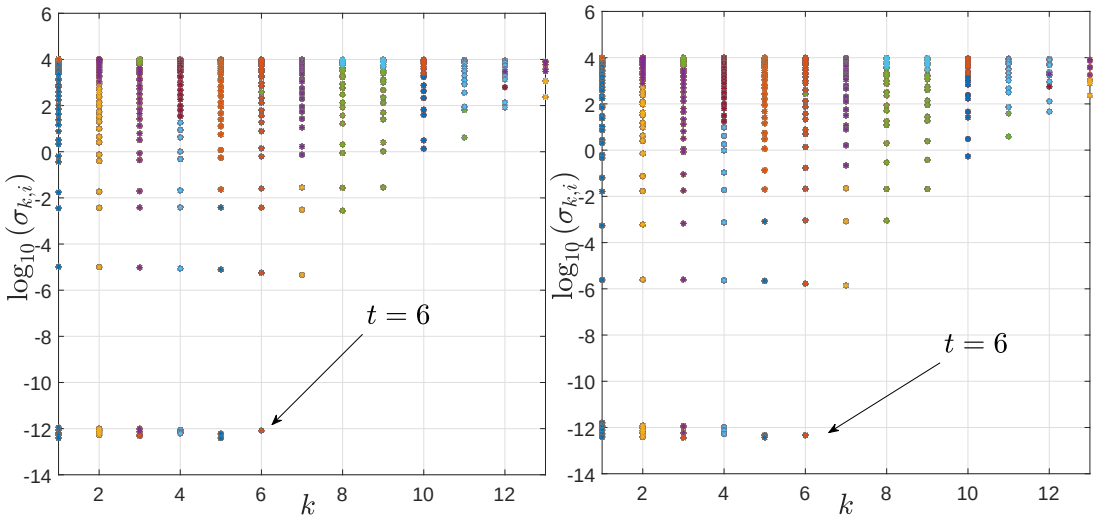


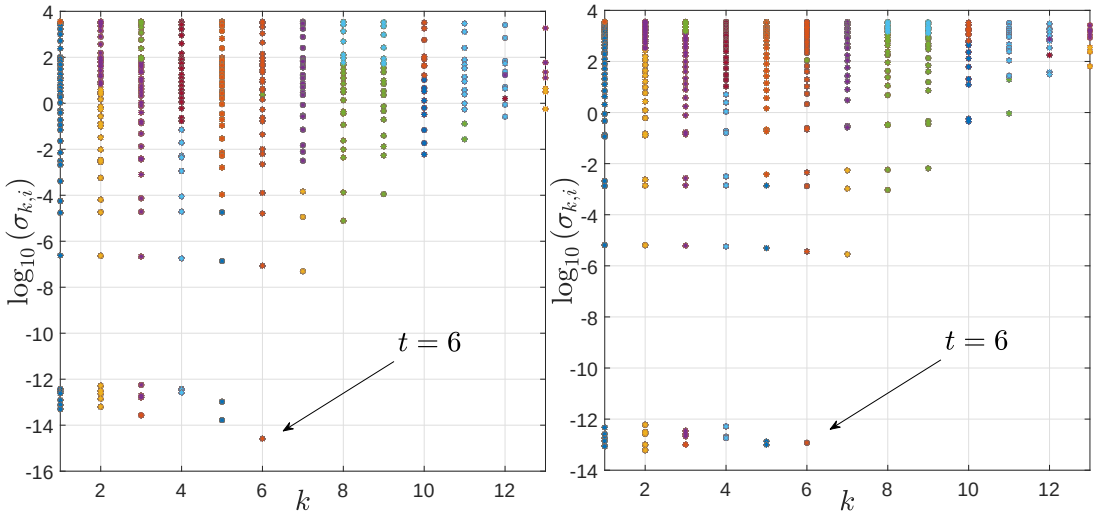
Figure 5.7: The coefficients of both the unprocessed polynomials $f(x)$, $g(x)$ and $h(x)$ and the preprocessed polynomials $\lambda_1\tilde{f}_1(\omega)$, $\tilde{g}_1(\omega)$ and $\rho_1\tilde{h}_1(\omega)$ in Example 5.1.2

of the (2×3) partitioned subresultant matrices and the (3×3) partitioned subresultant matrices.

□



(i) $\{\tilde{S}_k(\lambda_k \tilde{f}_k(\omega), \tilde{g}_k(\omega), \rho_k \tilde{h}_k(\omega))\}$ **(ii)** $\{\hat{S}_k(\lambda_k \tilde{f}_k(\omega), \tilde{g}_k(\omega), \rho_k \tilde{h}_k(\omega))\}$



(iii) $\{\hat{S}_k(\tilde{g}_k(\omega), \lambda_k \tilde{f}_k(\omega), \rho_k \tilde{h}_k(\omega))\}$ **(iv)** $\{\hat{S}_k(\rho_k \tilde{h}_k(\omega), \tilde{g}_k(\omega), \lambda_k \tilde{f}_k(\omega))\}$

Figure 5.8: The singular values $\{\sigma_{k,i}\}$ of the (3×3) and (2×3) preprocessed subresultant matrices in Example 5.1.2

Example 5.5.2. Consider the Bernstein form of the exact polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$, whose factorised forms are given by

$$\begin{aligned}
\hat{f}(x) &= (x-0.5654654561)^5(x-0.21657894)^{10}(x-0.01564897)^2\times \\
&\quad (x+1.234)^3(x+1.2468796514)^3 \\
\hat{g}(x) &= (x-0.99851354877)^3(x-0.5654654561)^5(x-0.21657894)^{10}\times \\
&\quad (x+1.2468796514)^3(x-1.75292)^4 \\
\hat{h}(x) &= (x-0.5654654561)^5(x-0.21657894)^{10}(x+0.778912324654)^4\times \\
&\quad (x+1.2468796514)^3(x+1.75)^2
\end{aligned}$$

and whose GCD $\hat{d}(x)$ of degree $t = 18$ is given by

$$\hat{d}(x) = (x-0.5654654561)^5(x-0.21657894)^{10}(x+1.2468796514)^3.$$

Noise is added to the coefficients of $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ such that the coefficients of the inexact polynomials are given by (5.23), where $\{r_{f,i}\}$, $\{r_{g,j}\}$ and $\{r_{h,p}\}$ are uniformly distributed random variables in the interval $[-1, 1]$ and $\{\epsilon_{f,i}\}$, $\{\epsilon_{g,j}\}$ and $\{\epsilon_{h,p}\}$ are uniformly distributed random variables in the interval $\left[10^{-8}, 10^{-6}\right]$ for $i = 0, \ldots, m$, $j = 0, \ldots, n$ and $p = 0, \ldots, o$. The inexact polynomials $f(x)$, $g(x)$ and $h(x)$ are preprocessed and the coefficients of the (i) unprocessed and (ii) preprocessed polynomials are shown in Figure 5.9.

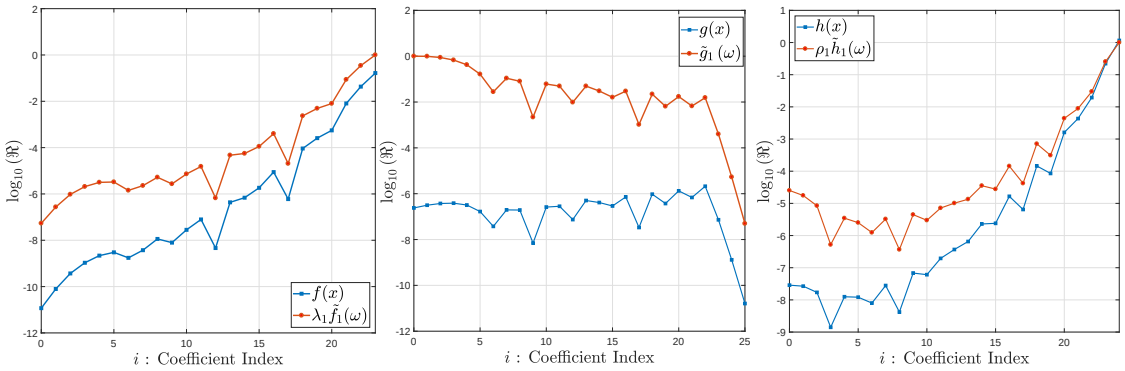
In Figure 5.10 the complete sets of singular values of each subresultant matrix are plotted. From Figure 5.10i, there is no clear separation between the numerically zero and non-zero singular values, and therefore the degree of the AGCD cannot be accurately determined by this method. However, the zero-like singular values of the preprocessed subresultant matrices $\{\hat{S}_k(\lambda_k\tilde{f}(\omega), \tilde{g}(\omega), \rho_k\tilde{h}(\omega))\}$ are clearly separated from the non-zero values in Figure 5.10ii, and the degree of the AGCD is correctly determined to be $t = 18$.

The set of the minimum singular values of the (i) unprocessed (ii) preprocessed subresultant matrices are shown in Figure 5.11. There is a large change in the minimum singular values of the preprocessed subresultant matrices, and it is seen that $\dot{\sigma}_{19} >> \dot{\sigma}_{18}$, so the degree of the AGCD is correctly identified. However, this is not the case for the unprocessed subresultant matrices.

□

Example 5.5.3. Consider the Bernstein form of the exact polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$, whose factorisations are given by

$$\begin{aligned}
\hat{f}(x) &= (x-0.5654654561)^5(x-0.21657894)(x-0.01564897)^2\times \\
&\quad (x+0.2468796514)^3(x+0.7879734) \\
\hat{g}(x) &= (x-0.99851354877)^7(x-0.75292)^{20}(x-0.5654654561)^5\times \\
&\quad (x-0.21657894)(x+0.2468796514)^3 \\
\hat{h}(x) &= (x-0.5654654561)^5(x-0.21657894)(x+0.2468796514)^3\times \\
&\quad (x+0.778912324654)^4(x+1.75)^2,
\end{aligned}$$

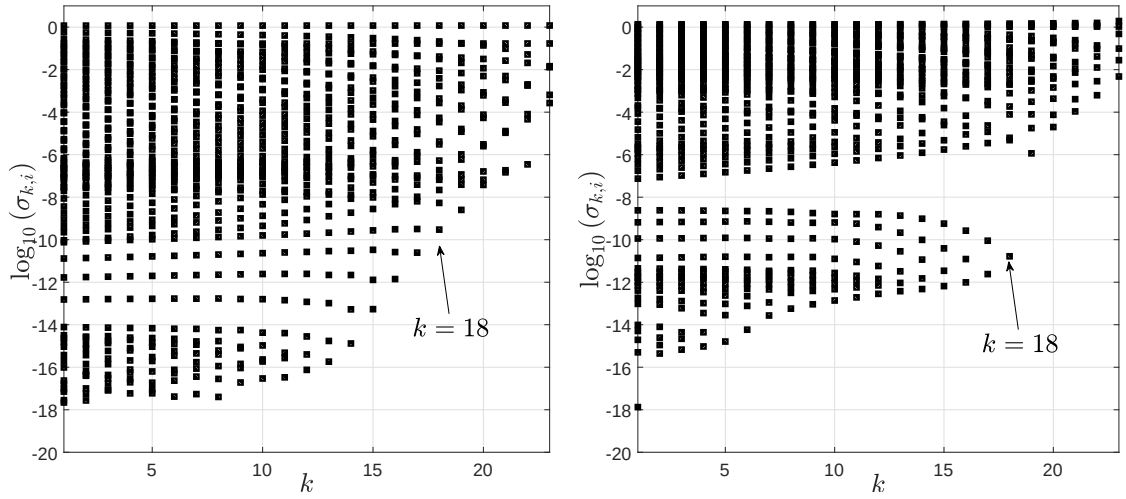


(i) The coefficients of $f(x)$ (○) and $\lambda_1\tilde{f}_1(\omega)$ (■)

(ii) The coefficients of $g(x)$ (○) and $\tilde{g}_1(\omega)$ (■)

(iii) The coefficients of $h(x)$ (○) and $\rho_1\tilde{h}_1(\omega)$ (■)

Figure 5.9: The coefficients of both the unprocessed polynomials $f(x)$, $g(x)$ and $h(x)$ and the preprocessed polynomials $\lambda_1\tilde{f}_1(\omega)$, $\tilde{g}_1(\omega)$ and $\rho_1\tilde{h}_1(\omega)$ in Example 5.5.2



(i) The singular values $\{\sigma_{k,i}\}$ of the unprocessed subresultant matrices $\{\hat{S}_k(f(x), g(x), h(x))\}$

(ii) The singular values $\{\sigma_{k,i}\}$ of the preprocessed subresultant matrices $\{\hat{S}_k(\lambda_k\tilde{f}(\omega), \tilde{g}(\omega), \rho_k\tilde{h}(\omega))\}$

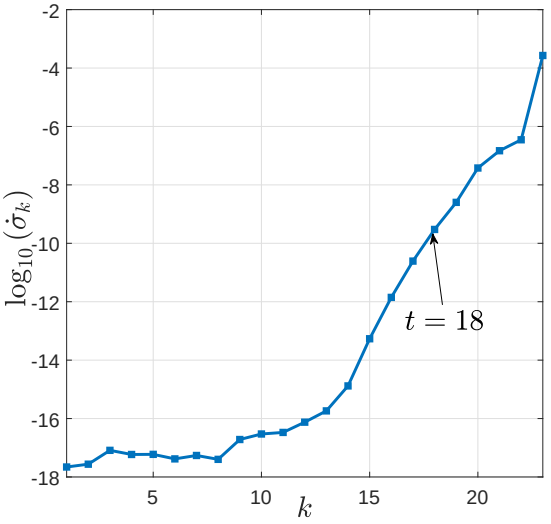
Figure 5.10: The singular values $\{\sigma_{k,i}\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 5.5.2

and whose GCD $\hat{d}_t(x)$ of degree $t = 9$ is given by

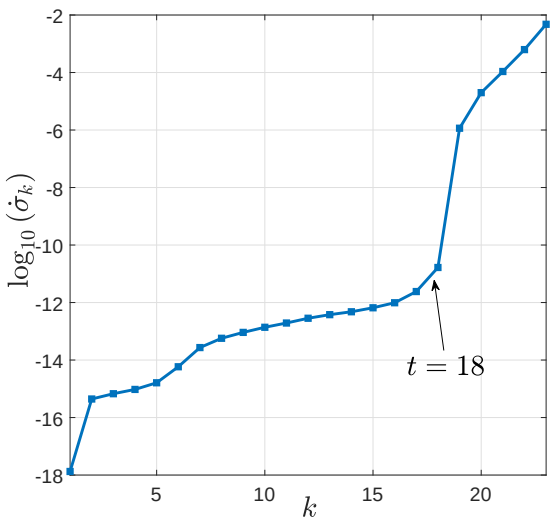
$$\hat{d}(x) = (x - 0.5654654561)^5(x - 0.21657894)(x + 0.2468796514)^3.$$

Noise is added to the coefficients of $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$ such that the coefficients of the inexact polynomials $f(x)$, $g(x)$ and $h(x)$ are given by (5.23), where $\{\epsilon_{f,i}\}$, $\{\epsilon_{g,j}\}$ and $\{\epsilon_{h,p}\}$ are uniformly distributed random variables in the interval $[1e-7, 1e-4]$. The polynomials $f(x)$, $g(x)$ and $h(x)$ are preprocessed to obtain the sets $\{\lambda_i\tilde{f}_i(\omega)\}$, $\{\tilde{g}_i(\omega)\}$ and $\{\rho_i\tilde{h}_i(\omega)\}$ and the coefficients of $\lambda_1\tilde{f}_1(\omega)$, $\tilde{g}_1(\omega)$ and $\rho_1\tilde{h}_1(\omega)$ are plotted in Figure 5.12 alongside their corresponding unprocessed forms $f(x)$, $g(x)$ and $h(x)$.

The singular value decomposition of each of the subresultant matrices $\{S_k(f(x), g(x), h(x))\}$ and $\{S_k(\lambda_k\tilde{f}_k(\omega), \tilde{g}_k(\omega), \rho_k\tilde{h}_k(\omega))\}$ is computed, and the sets of singular values of the unprocessed and preprocessed subresultant matrices are plotted in Figure 5.13i and Figure 5.13ii respectively. There is no distinct separation between



(i) The minimum singular values $\{\dot{\sigma}_k\}$ of the unprocessed subresultant matrices $\{\hat{S}_k(f(x), g(x), h(x))\}$



(ii) The minimum singular values $\{\dot{\sigma}_k\}$ of the preprocessed subresultant matrices $\{\hat{S}_k(\lambda_k \tilde{f}_k(\omega), \tilde{g}_k(\omega), \rho_k \tilde{h}_k(\omega))\}$

Figure 5.11: The minimum singular values $\{\dot{\sigma}_k\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 5.5.2

the zero and non-zero singular values of the unprocessed subresultant matrices. However, there is a clear separation between the zero and non-zero sets of singular values of the preprocessed subresultant matrices shown in Figure 5.13ii.

□

Example 5.5.4. Consider the Bernstein form of the exact polynomials $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$, whose factorisations are given by

$$\hat{f}(x) = (x - 1.46)^2(x - 1.37)^3(x - 1.2)(x - 0.82)^3(x - 0.75)^5(x - 0.56)^4(x - 0.1)^2(x + 0.27)^4$$
$$\hat{g}(x) = (x - 0.99)^4(x - 0.12)^4(x + 0.2)^3(x - 0.1)^2(x - 0.56)^4(x - 0.75)^5(x - 1.37)^3$$
$$\hat{h}(x) = (x - 1.37)^3(x - 0.75)^5(x - 0.72)^8(x - 0.56)^4(x - 0.1)^2(x + 0.75)^2$$

and whose GCD $\hat{d}(x)$ of degree $t = 14$ is given by

$$\hat{d}(x) = (x - 0.1)^2(x - 0.56)^4(x - 0.75)^5(x - 1.37)^3.$$

Noise is added to the coefficients of $\hat{f}(x)$, $\hat{g}(x)$ and $\hat{h}(x)$, where $\{\epsilon_{f,i}\}$, $\{\epsilon_{g,j}\}$ and $\{\epsilon_{h,p}\}$ are all uniformly distributed random variables in the interval $[10^{-4}, 10^{-6}]$.

The inexact polynomials are preprocessed, and the optimal values λ, ρ and θ for the polynomials in the first subresultant matrix are given by $\lambda_1 = 1.4804$, $\rho_1 = 2.3823$ and $\theta_1 = 1.5543$. The coefficients of the unprocessed polynomials $f(x)$, $g(x)$ and $h(x)$ and the preprocessed polynomials $\lambda_1 \tilde{f}_1(\omega)$, $\tilde{g}_1(\omega)$ and $\rho_1 \tilde{h}_1(\omega)$ are plotted in Figure 5.14, where it shown that the coefficients of the preprocessed polynomials span many fewer orders of magnitude than the unprocessed polynomials. The singular values $\{\sigma_{k,i}\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices are plotted in Figure 5.15. In Figure 5.15i there is no clear separation between the numerically zero and non-zero singular values. However, in Figure 5.15ii there is a clear separation between these two

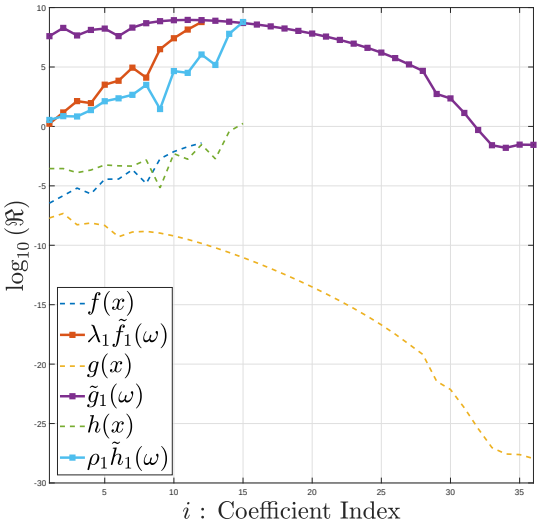
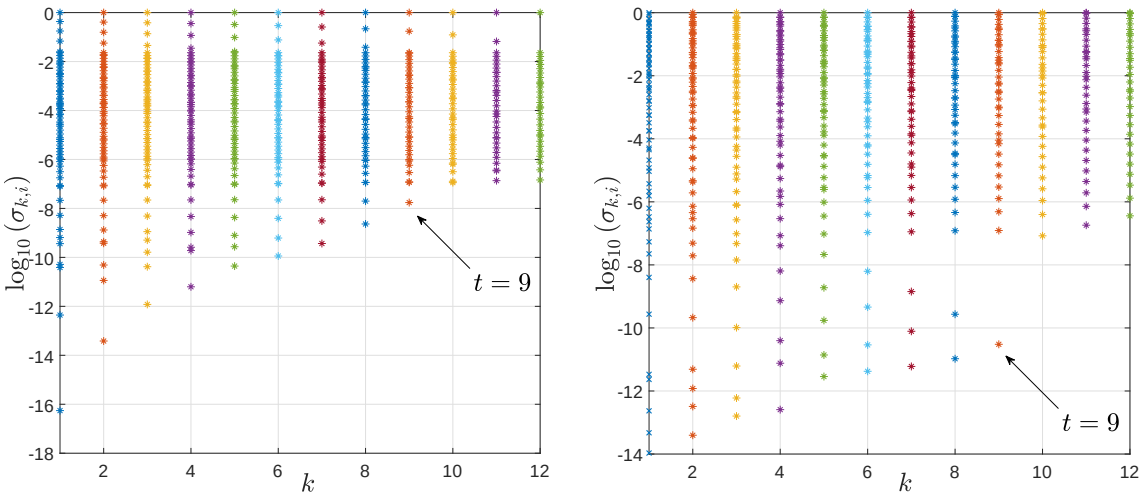


Figure 5.12: The coefficients of both the unprocessed polynomials $f(x)$, $g(x)$ and $h(x)$ and the preprocessed polynomials $\lambda_1\tilde{f}_1(\omega)$, $\tilde{g}_1(\omega)$ and $\rho_1\tilde{h}_1(\omega)$ in Example 5.5.3



(i) $\{\hat{S}_k(f(x), g(x), h(x))\}$ **(ii)** $\{\hat{S}_k(\lambda_k\tilde{f}_k(\omega), \tilde{g}_k(\omega), \rho_k\tilde{h}_k(\omega))\}$

Figure 5.13: The singular values $\{\sigma_{k,i}\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 5.5.3

sets of values and the degree of the GCD can effectively be determined by (i) the set of singular values of the first subresultant matrix given by $\{\sigma_{1,i}\}$ by DC1 and (ii) the set of minimum singular values of the set of subresultant matrices $\{\dot{\sigma}_k\}$ by DC2. From this figure, the degree of the GCD is well defined and is given by $t = 14$.

Since analysis of the singular values of the unprocessed subresultant matrices fails to reveal the degree of the GCD, the coefficients of the approximations $u_t(x)$, $v_t(x)$, $w_t(x)$ and $d_t(x)$ cannot be determined and so the corresponding column of measured errors in Table 5.1 remains blank. However, the tth preprocessed subresultant matrix $\hat{S}_t(\lambda_t\tilde{f}_t(\omega), \tilde{g}_t(\omega), \rho_t\tilde{h}_t(\omega))$ is used to approximate the coefficients of $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{w}_t(x)$. Polynomials $\tilde{u}_t(\omega)$, $\tilde{v}_t(\omega)$ and $\tilde{w}_t(\omega)$ are computed by least squares (5.30) and the coefficients of $\tilde{d}_t(\omega)$ are given by the solution of (5.31). Replacing the independent variable $\omega = x/\theta$, the polynomials $\tilde{u}_t(x)$, $\tilde{v}_t(x)$ and $\tilde{w}_t(x)$ are approximations of $\hat{u}_t(x)$, $\hat{v}_t(x)$ and $\hat{w}_t(x)$ and the errors in these values are shown in Table 5.1.

The amount of additive noise is reduced and the values of $\{\epsilon_{f,i}\}$, $\{\epsilon_{g,j}\}$ and $\{\epsilon_{h,p}\}$

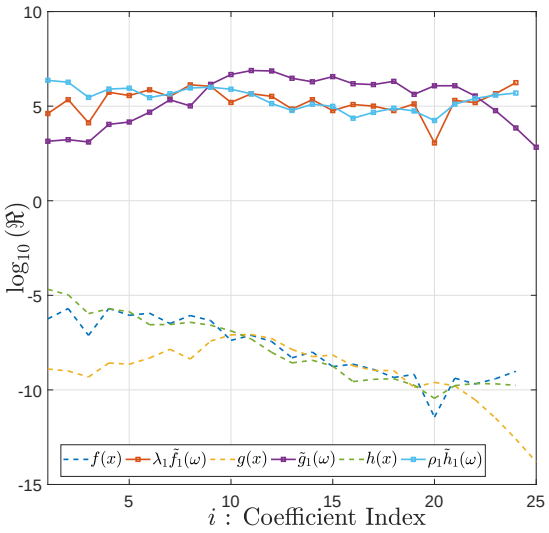


Figure 5.14: The coefficients of both the unprocessed polynomials $f(x)$, $g(x)$ and $h(x)$ and preprocessed polynomials $\lambda_1\tilde{f}_1(\omega)$, $\tilde{g}_1(\omega)$ and $\mu_1\tilde{h}_1(\omega)$ in Example 5.5.4

	Unprocessed $u_t(x)$, $v_t(x)$, $w_t(x)$, and $d_t(x)$	Preprocessed $\tilde{u}_t(x)$, $\tilde{v}_t(x)$, $\tilde{w}_t(x)$ and $\tilde{d}_t(x)$
Error $\hat{u}_t(x)$	-	$1.744320e-05$
Error $\hat{v}_t(x)$	-	$8.334257e-05$
Error $\hat{w}_t(x)$	-	$3.447570e-05$
Error $\hat{d}_t(x)$	-	$8.504447e-06$
Average	-	$3.643007e-05$

Table 5.1: Error in the approximations of $\hat{u}_t(x)$, $\hat{v}_t(x)$, $\hat{w}_t(x)$ and $\hat{d}_t(x)$ with $\{\epsilon_{f,i}\}$, $\{\epsilon_{g,j}\}$ and $\{\epsilon_{h,p}\}$ in the interval $[1e-6, 1e-4]$ in Example 5.5.4

are now set to be uniformly distributed random variables in the interval $[10^{-10}, 10^{-8}]$. The degree of the GCD can now be determined from the sets of singular values of the unprocessed and preprocessed subresultant matrices.

Two sets of approximations of the cofactor polynomials and the GCD can now be computed as stated in Section 5.4:

1. The approximations $u_t(x)$, $v_t(x)$, $w_t(x)$ and $d_t(x)$ are computed from the tth unprocessed subresultant matrix.

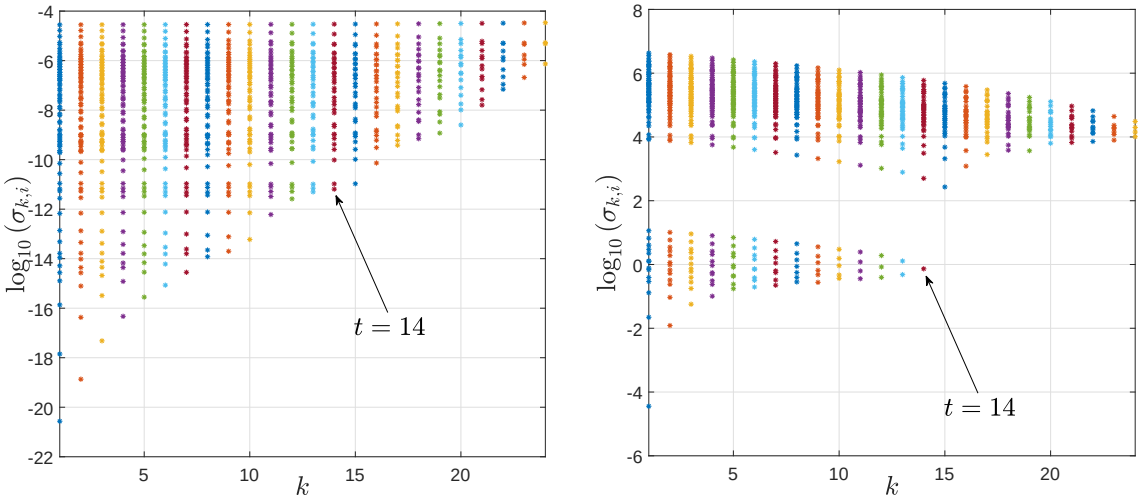
2. The approximations $\tilde{u}_t(x)$, $\tilde{v}_t(x)$, $\tilde{w}_t(x)$ and $\tilde{d}_t(x)$ are computed from the tth preprocessed subresultant matrix.

The errors in these approximations are given in Table 5.2, from which it can be seen that the approximations $(\tilde{u}_t(x), \tilde{v}_t(x), \tilde{w}_t(x), \tilde{d}_t(x))$ are significantly better than the approximations $(u_t(x), v_t(x), w_t(x), d_t(x))$.

□

	Unprocessed $u_t(x)$, $v_t(x)$, $w_t(x)$ and $d_t(x)$	Preprocessed $\tilde{u}_t(x)$, $\tilde{v}_t(x)$, $\tilde{w}_t(x)$ and $\tilde{d}_t(x)$
Error $\hat{u}_t(x)$	$4.230783e-06$	$1.789312e-09$
Error $\hat{v}_t(x)$	$1.588693e-05$	$8.260332e-09$
Error $\hat{w}_t(x)$	$1.111483e-06$	$3.481161e-09$
Error $\hat{d}_t(x)$	$1.448698e-06$	$7.129575e-10$
Average	$7.188802e-06$	$3.587534e-09$

Table 5.2: Error in the approximations of $\hat{u}_t(x)$, $\hat{v}_t(x)$, $\hat{w}_t(x)$ and $\hat{d}_t(x)$ with $\{\epsilon_{f,i}\}$, $\{\epsilon_{g,j}\}$ and $\{\epsilon_{h,p}\}$ in the interval $[1e-10, 1e-8]$ in Example 5.5.4



(i) The singular values $\{\sigma_{k,i}\}$ of the unprocessed subresultant matrices

(ii) The singular values $\{\sigma_{k,i}\}$ of the preprocessed subresultant matrices

Figure 5.15: The singular values $\{\sigma_{k,i}\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 5.5.4

5.6 Conclusion

The two-polynomial GCD finding method was extended to compute the GCD of three univariate polynomials in Bernstein form. Initial investigations into this problem revealed two variations of the three-polynomial subresultant matrices. These variations were defined in a way which extended the two-polynomial subresultant matrix. The first of these new forms was the (2×3) partitioned structure which had three alternate variations given by $\hat{S}_k(f(x), g(x), h(x))$, $\hat{S}_k(g(x), f(x), h(x))$ and $\hat{S}_k(h(x), g(x), f(x))$, and it was shown that these forms may have different numerical rank dependent on the ordering of the three polynomials.

Some further conclusions based on the work in this chapter are :

1. Each subresultant matrix variation is equivalent to solving two pairwise GCD problems simultaneously. Suppose the pairwise GCD of $f(x)$ and $g(x)$ is poorly defined, then the two (2×3) partitioned subresultant matrix variants containing the row-partition $R_{a,k}$ (see (5.9)) are also poorly defined. The GCD is therefore most reliably computed from the third (2×3) partitioned subresultant matrix variant containing row-partitions $R_{b,k}$ and $R_{c,k}$ ((5.10), (5.11)).

2. Poorly scaled row-partitions give bad results in the computation of the degree of the GCD. Suppose $f(x)$ and $g(x)$ have a GCD of degree t_a, which is greater than the degree t of the GCD of $f(x)$, $g(x)$ and $h(x)$, then the best results are obtained from the subresultant matrix where the row-partition $R_{a,k}$ is not included.

3. Preprocessing goes some way to mitigating the effect of poorly scaled row-partitions.

The next chapter will extend the two-polynomial and three-polynomial GCD finding method to compute the GCD and AGCD of two and three bivariate polynomials in Bernstein form defined over a triangular domain.

Chapter 6

GCDs of Bivariate Polynomials over a Triangular Domain

The previous chapter discussed the extension of the UGCD method for the computation of the GCD of three univariate polynomials in Bernstein form. This chapter now considers another extension to compute the GCD of two and three bivariate polynomials in Bernstein form where the polynomials are defined over a triangular domain. The factorisation of a bivariate polynomial over a triangular domain requires the extension of the square-free factorisation algorithm (Algorithm 1).

The methods in this chapter for solving the two-polynomial and three-polynomial GCD problems are similar to the methods described in Chapter 3 and Chapter 5 respectively. However, the structure of the subresultant matrices is significantly different, and the total degree of the bivariate GCD is sought.

Section 6.1 The first section of this chapter describes the method of computing the square-free factorisation of a bivariate polynomial by extension of the square-free factorisation algorithm for univariate polynomials (Algorithm 1). Similar to the univariate polynomial factorisation, this reduces to the computation of a sequence of three-polynomial GCD problems and a set of deconvolutions.

Section 6.2 The second section considers the computation of the degree of the GCD of two or three exact bivariate polynomials and the AGCD of two or three inexact polynomials in Bernstein form. The method described in this section uses a sequence of subresultant matrices in a similar way to the UGCD method defined for the computation of the GCD of univariate polynomials.

Section 6.3 This section considers the variants of the two and three-polynomial subresultant matrices for bivariate polynomials defined over a triangular domain. As with the subresultant matrices of univariate polynomials, there exist several variants of the subresultant matrices of bivariate polynomials. The trinomial terms in the entries of the subresultant matrices can cause the entries to span many orders of magnitude. Experiments will consider which of the variants is optimal for the computation of the degree and coefficients of the GCD of two or three polynomials.

Section 6.4 Preprocessing of the two-polynomial and three-polynomial subresultant matrices for bivariate polynomials over a triangular domain is described, and some

results are shown.

Section 6.5 This section considers the computation of the coefficients of cofactor polynomials in the two-polynomial and three-polynomial GCD finding problem. The simple least squares based method will be used to the compute coefficients of the cofactor polynomials given the tth subresultant matrix.

Section 6.6 Results using the methods developed in this chapter are presented. It will be shown how the degree of the GCD of two or three bivariate polynomials is reliably computed from the set of preprocessed subresultant matrices in cases where unprocessed subresultant matrices otherwise fail.

6.1 The Bivariate Polynomial Square-Free Factorisation Algorithm

This section describes an extension to the square-free factorisation algorithm for univariate polynomials (Algorithm 1), such that a bivariate polynomial can be factorised into its irreducible factors in x and y and non-separable factors of both x and y. Let $\hat{p}_k(x)$ be the product of all factors of the bivariate polynomial $\hat{f}(x,y)$ only in x with multiplicity k. Let m_x denote the highest multiplicity of any of these factors in $\hat{f}(x,y)$ such that all factors are contained in the set $\{\,\hat{p}_k(x) \mid k = 1,\ldots,m_x\,\}$ and $\hat{p}_k(x) = 1$ if there is no factor of multiplicity k.

Similarly, let $\hat{q}_k(y)$ be the product of all factors of $\hat{f}(x,y)$, which are polynomials in y with multiplicity k, and let m_y denote the highest multiplicity of any of these factors in $\hat{f}(x,y)$ such that all factors are contained in the set $\{\,\hat{q}_k(y) \mid k = 1,\ldots,m_y\,\}$ and $\hat{q}_k(y) = 1$ if there is no factor of multiplicity k.

Finally, let $\hat{r}_k(x,y)$ be the product of all factors of $\hat{f}(x,y)$, which are non-separable with multiplicity k, and let $m_{x,y}$ denote the highest multiplicity of any of these factors in $\hat{f}(x,y)$ such that $\{\,\hat{r}_k(x,y) \mid k = 1,\ldots,m_{xy}\,\}$ and $\hat{r}_k(x,y) = 1$ when no non-separable factor of multiplicity k exists.

To compute a factorisation of $\hat{f}(x,y)$, first consider the polynomial as a product of the polynomials $\hat{p}_i(x)$, $\hat{q}_j(y)$ and $\hat{r}_k(x,y)$ for $i = 1,\ldots,m_x$, $j = 1,\ldots,m_y$ and $k = 1,\ldots,m_{xy}$

$$\begin{aligned}\hat{f}_0(x,y) = &\left(\hat{p}_1(x)\hat{p}_2^2(x)\hat{p}_3^3(x)\ldots\hat{p}_{m_x}^{m_x}(x)\right)\left(\hat{q}_1(y)\hat{q}_2^2(y)\hat{q}_3^3(y)\ldots\hat{q}_{m_y}^{m_y}(y)\right)\times\\ &\left(\hat{r}_1(x,y)\hat{r}_2^2(x,y)\hat{r}_3^3(x,y)\ldots\hat{r}_{m_{xy}}^{m_{xy}}(x,y)\right).\end{aligned}$$

The partial derivative of $\hat{f}_0(x,y)$ with respect to x is given by

$$\begin{aligned}\frac{\partial\hat{f}_0(x,y)}{\partial x} = &\left(\hat{p}_2(x)\hat{p}_3^2(x)\ldots\hat{p}_{m_x}^{m_x-1}(x)\right)\left(\hat{q}_1(y)\hat{q}_2^2(y)\hat{q}_3^3(y)\ldots\hat{q}_{m_y}^{m_y}(y)\right)\times\\ &\left(\hat{r}_2(x,y)\hat{r}_3^2(x,y)\ldots\hat{r}_{m_{xy}}^{m_{xy}-1}(x,y)\right)\times k_x(x,y),\end{aligned}$$

where $k_x(x,y)$ is a bivariate polynomial whose degree is such that $\frac{\partial\hat{f}_0(x,y)}{\partial x}$ is of the same degree as $\hat{f}_0(x,y)$ in y and has degree of one less than $\hat{f}_0(x,y)$ in x. Details of the degree structure of $k_x(x,y)$ need not be known for the purpose of this algorithm.

The partial derivative of $\hat{f}_0(x,y)$ with respect to y is given by

$$\frac{\partial \hat{f}_0(x,y)}{\partial y} = \left(\hat{p}_1(x)\hat{p}_2^2(x)\hat{p}_3^3(x)\dots\hat{p}_{m_x}^{m_x}(x)\right)\left(\hat{q}_2(y)\hat{q}_3^2(y)\dots\hat{q}_{m_y}^{m_y-1}(y)\right)$$
$$\left(\hat{r}_2(x,y)\hat{r}_3^2(x,y)\dots\hat{r}_{m_{xy}}^{m_{xy}-1}(x,y)\right)\times k_y(x,y),$$

where $k_y(x,y)$ is defined in a similar way to $k_x(x,y)$. Its degree structure is such that $\frac{\partial \hat{f}_0(x,y)}{\partial y}$ has the same degree as $\hat{f}_0(x,y)$ in x and has degree one less than $\hat{f}_0(x,y)$ in y.

The polynomial $\hat{f}_1(x,y)$ is given by

$$\hat{f}_1(x,y) = \mathrm{GCD}\left(\hat{f}_0(x,y), \frac{\partial \hat{f}_0(x,y)}{\partial x}, \frac{\partial \hat{f}_0(x,y)}{\partial y}\right)$$
$$= \left(\hat{p}_2(x)\hat{p}_3^2(x)\dots\hat{p}_{m_x}^{m_x-1}(x)\right)\left(\hat{q}_2(y)\hat{q}_3^2(y)\dots\hat{q}_{m_y}^{m_y-1}(y)\right)\left(\hat{r}_2(x,y)\hat{r}_3^2(x,y)\dots\hat{r}_{m_{xy}}^{m_{xy}-1}(x,y)\right).$$

The sequence of polynomials $\{\hat{f}_k(x,y)\}$ is generated, where $\hat{f}_k$ is given by computing the GCD of $\hat{f}_{k-1}(x,y)$ and its derivatives with respect to x and y

$$\hat{f}_k(x,y) = \mathrm{GCD}\left(\hat{f}_{k-1}(x,y), \frac{\partial \hat{f}_{k-1}(x,y)}{\partial x}, \frac{\partial \hat{f}_{k-1}(x,y)}{\partial y}\right)$$
$$= \left(\hat{p}_{k+1}(x)\hat{p}_{k+2}^2(x)\dots\hat{p}_{m_x}^{m_x-k}(x)\right)\left(\hat{q}_{k+1}(y)\hat{q}_{k+2}^2(y)\dots\hat{q}_{m_y}^{m_y-k}(y)\right)$$
$$\left(\hat{r}_{k+1}(x,y)\hat{r}_{k+2}^2(x,y)\dots\hat{r}_{m_{xy}}^{m_{xy}-k}(x,y)\right),$$

which terminates when $\hat{f}_k$ is square-free and $\hat{f}_k(x,y)$ and its derivatives are coprime.

Suppose that $m_x > m_y > m_{xy}$, then:

1. The polynomials in the set $\{\hat{f}_k(x,y) \mid k = 0,\dots,(m_{xy}-1)\}$ can be written as the product of polynomials of the form $\hat{p}(x)$, $\hat{q}(y)$ and $\hat{r}(x,y)$.

2. The polynomials in the set $\{\hat{f}_k(x,y) \mid k = m_{xy},\dots,m_y-1\}$ can be written as the product of polynomials of the form $\hat{p}(x)$ and $\hat{q}(y)$.

3. Finally, the polynomials in the set $\{\hat{f}_k(x,y) \mid k = m_y,\dots,m_x-1\}$ are all univariate in x and are written as the product of polynomials of the form $\hat{p}(x)$.

As with the univariate polynomial factorisation algorithm, the set of polynomials $\{\hat{h}_i(x,y)\}$ is given by a series of deconvolutions of the set of $\hat{f}_i(x,y)$ and each

$\hat{h}_i(x,y) = \hat{f}_{i-1}(x,y)/\hat{f}_i(x,y)$

$$\begin{array}{rcccl}
\hat{h}_1(x,y) & = & \frac{\hat{f}_0(x,y)}{\hat{f}_1(x,y)} & = & (\hat{p}_1(x)\hat{p}_2(x)\hat{p}_3(x)\dots\hat{p}_{m_x})\,(\hat{q}_1(y)\hat{q}_2(y)\hat{q}_3(y)\dots\hat{q}_{m_y}(y)) \\
 & & & & \times\,(\hat{r}_1(x,y)\hat{r}_2(x,y)\hat{r}_3(x,y)\dots\hat{r}_{m_{xy}}(x,y)) \\
\hat{h}_2(x,y) & = & \frac{\hat{f}_1(x,y)}{\hat{f}_2(x,y)} & = & (\hat{p}_2(x)\hat{p}_3(x)\dots\hat{p}_{m_x}(x))\,(\hat{q}_2(y)\hat{q}_3(y)\dots\hat{q}_{m_y}(y)) \\
 & & & & \times\,(\hat{r}_2(x,y)\hat{r}_3(x,y)\dots\hat{r}_{m_{xy}}(x,y)) \\
\vdots & & \vdots & & \vdots \\
\hat{h}_{m_{xy}}(x,y) & = & \frac{\hat{f}_{m_{xy}-1}(x,y)}{\hat{f}_{m_{xy}}(x,y)} & = & (\hat{p}_{m_{xy}}(x)\hat{p}_{m_{xy}+1}(x)\dots\hat{p}_{m_x}(x))\,(\hat{q}_{m_{xy}}(y)\hat{q}_{m_{xy}+1}(y)\dots\hat{q}_{m_y}(y)) \\
 & & & & \times\,(\hat{r}_{m_{x,y}}(x,y)) \\
\vdots & & \vdots & & \vdots \\
\hat{h}_{m_y}(x,y) & = & \frac{\hat{f}_{m_y-1}(x,y)}{\hat{f}_{m_y}(x,y)} & = & \hat{p}_{m_y}(x)\hat{p}_{m_y+1}(x)\dots\hat{p}_{m_x}(x) \\
\vdots & & \vdots & & \vdots \\
\hat{h}_{m_x}(x,y) & = & \frac{\hat{f}_{m_x-1}(x,y)}{\hat{f}_{m_x}(x,y)} & = & \hat{p}_{m_x}(x).
\end{array}$$

The set of polynomials $\{\,\hat{w}_i(x,y) \mid i = 1,\dots,m_x\,\}$ is given by $\hat{w}_i = \hat{h}_i(x,y)/\hat{h}_{i+1}(x,y)$

$$\begin{array}{rcccl}
\hat{w}_1(x,y) & = & \frac{\hat{h}_1(x,y)}{\hat{h}_2(x,y)} & = & \hat{p}_1(x)\hat{q}_1(y)\hat{r}_1(x,y) \\
\hat{w}_2(x,y) & = & \frac{\hat{h}_2(x,y)}{\hat{h}_3(x,y)} & = & \hat{p}_2(x)\hat{q}_2(y)\hat{r}_2(x,y) \\
\vdots & & & & \\
\hat{w}_{m_{xy}}(x,y) & = & \frac{\hat{h}_{m_{xy}}(x,y)}{\hat{h}_{m_{xy}+1}(x,y)} & = & \hat{p}_{m_{xy}}(x)\hat{q}_{m_{xy}}(y)\hat{r}_{m_{xy}}(x,y) \\
\vdots & & & & \\
\hat{w}_{m_y}(x,y) & = & \frac{\hat{h}_{m_y}(x,y)}{\hat{h}_{m_y+1}(x,y)} & = & \hat{p}_{m_y}(x)\hat{q}_{m_y}(y) \\
\vdots & & & & \\
\hat{w}_{m_x}(x,y) & = & \hat{h}_{m_x} & = & \hat{p}_{m_x}(x).
\end{array}$$

Each polynomial in the set of square-free polynomials $\{\hat{w}_i(x,y)\}$ contains the irreducible factors of $\hat{f}(x,y)$ which have multiplicity i in $\hat{f}(x,y)$.

Example 6.1.1. Consider the exact polynomial $\hat{f}(x,y)$ given by

$$\hat{f}(x,y) = (x^2+y^2-0.2)^2(x+0.5)^3(y-0.7)^4.$$

The partial derivative of $\hat{f}_0(x,y)$ with respect to x is given by

$$\frac{\partial \hat{f}_0(x,y)}{\partial x} = (x^2+y^2-0.2)(x+0.5)^2(y-0.7)^4 \times 7(-0.0857143 + 0.285714x + x^2 + 0.428571y^2)$$

and the partial derivative with respect to y is given by

$$\frac{\partial \hat{f}_0(x,y)}{\partial y} = (x^2+y^2-0.2)(x+0.5)^3(y-0.7)^3 \times 4(-0.2 + x^2 - 0.7y + 2y^2).$$

The polynomial $\hat{f}_1(x,y)$ is the GCD of $\hat{f}_0(x,y)$ and its partial derivatives, and is given by

$$\hat{f}_1(x) = (x^2+y^2-0.2)(x+0.5)^2(y-0.7)^3.$$

The partial derivative of $\hat{f}_1(x)$ with respect to x is given by

$$\frac{\partial \hat{f}_1(x,y)}{\partial x} = (x+0.5)(y-0.7)^3 \times 2(-0.2+0.5x+2x^2+y^2)$$

and the partial derivative with respect to y is given by

$$\frac{\partial \hat{f}_1(x,y)}{\partial y} = (x+0.5)^2(y-0.7)^2 \times 3(-0.2+x^2-0.466667y+1.66667y^2),$$

so $\hat{f}_2(x,y)$ is given by

$$\hat{f}_2(x,y) = \mathrm{GCD}\left(\hat{f}_1(x,y), \frac{\partial \hat{f}_1(x,y)}{\partial x}, \frac{\partial \hat{f}_1(x,y)}{\partial y}\right) = (x+0.5)(y-0.7)^2.$$

The partial derivatives of $\hat{f}_2(x,y)$ with respect to x and y are given by

$$\frac{\partial \hat{f}_2(x,y)}{\partial x} = (y-0.7)^2 \quad \text{and} \quad \frac{\partial \hat{f}_2(x,y)}{\partial y} = 2(x+0.5)(y-0.7),$$

and $\hat{f}_3(x,y)$ is the GCD of $\hat{f}_2(x,y)$ and its partial derivatives so is given by

$$\hat{f}_3(x,y) = (y-0.7).$$

Finally, the partial derivatives with respect to x and y are given by

$$\frac{\partial \hat{f}_3(x,y)}{\partial x} = 0, \quad \text{and} \quad \frac{\partial \hat{f}_3(x,y)}{\partial y} = 1$$

and the GCD of $\hat{f}_3(x,y)$ and its derivatives is given by $\hat{f}_4(x,y)$, where

$$\hat{f}_4(x,y) = 1.$$

The set of polynomials $\{\hat{h}_i(x,y) \mid i = 1,\ldots,4\}$ is given by

$$\begin{aligned}
\hat{h}_1(x,y) &= \frac{\hat{f}_0(x,y)}{\hat{f}_1(x,y)} = (x^2+y^2-0.2)(x+0.5)(y-0.7)\\
\hat{h}_2(x,y) &= \frac{\hat{f}_1(x,y)}{\hat{f}_2(x,y)} = (x^2+y^2-0.2)(x+0.5)(y-0.7)\\
\hat{h}_3(x,y) &= \frac{\hat{f}_2(x,y)}{\hat{f}_3(x,y)} = (x+0.5)(y-0.7)\\
\hat{h}_4(x,y) &= \frac{\hat{f}_3(x,y)}{\hat{f}_4(x,y)} = (y-0.7).
\end{aligned}$$

and the set of polynomials $\{\, \hat{w}_i(x,y) \mid i = 1, \ldots 4 \,\}$ is given by

$$\begin{aligned}
\hat{w}_1(x,y) &= \frac{\hat{h}_1(x,y)}{\hat{h}_2(x,y)} = 1 \\
\hat{w}_2(x,y) &= \frac{\hat{h}_2(x,y)}{\hat{h}_3(x,y)} = x^2 + y^2 - 0.2 \\
\hat{w}_3(x,y) &= \frac{\hat{h}_3(x,y)}{\hat{h}_4(x,y)} = x + 0.5 \\
\hat{w}_4(x,y) &= \hat{h}_4(x,y) = y - 0.7
\end{aligned}$$

The polynomials $\{\hat{w}_i(x,y)\}$ are the factors of $\hat{f}(x,y)$ of multiplicity i. □

The factorisation of a bivariate polynomial has been reduced to a sequence of three-polynomial GCD problems followed by a set of deconvolutions. In the remainder of this chapter the two-polynomial and three-polynomial GCD finding problems are considered for Bernstein polynomials defined over a triangular domain.

6.2 The GCD of Two or Three Bivariate Polynomials in Bernstein Form over a Triangular Domain

In Section 3.1.1 and Section 5.1 the two-polynomial and three-polynomial subresultant matrices were defined for univariate polynomials. These subresultant matrices were used in the computation of the degree and coefficients of the GCD of two or three univariate polynomials respectively.

The determination of the degree of the GCD of two or three bivariate polynomials is similarly reduced to the determination of the numerical rank of each matrix in a sequence of two-polynomial or three-polynomial subresultant matrices defined for bivariate polynomials.

6.2.1 The Degree of the GCD of Two Bivariate Polynomials

The bivariate polynomial in Bernstein form defined over a triangular domain was defined in Section 2.2.3. Consider two bivariate polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ of total degree m and n respectively, which are given by

$$\hat{f}(x,y) = \sum_{i_1+i_2=0}^{m} \hat{a}_{i_1,i_2} B^m_{i_1,i_2}(x,y) \qquad \text{and} \qquad \hat{g}(x,y) = \sum_{i_1+i_2=0}^{n} \hat{b}_{i_1,i_2} B^n_{i_1,i_2}(x,y).$$

If they have a GCD $\hat{d}_t(x,y)$ of total degree t, then there exist cofactor polynomials of degrees $(m-t)$ and $(n-t)$ such that

$$\frac{\hat{f}(x,y)}{\hat{u}_t(x,y)} = \hat{d}_t(x,y) \qquad \text{and} \qquad \frac{\hat{g}(x,y)}{\hat{v}_t(x,y)} = \hat{d}_t(x,y), \tag{6.1}$$

where $\hat{d}_t(x,y)$ is defined to within a scalar constant. There also exists a set of common divisors $\{\hat{d}_k(x,y)\}^{(\mathrm{i})}$ of degree $k = 1,\ldots,t$ such that

$$\hat{f}(x,y)\hat{v}_k(x,y) - \hat{g}(x,y)\hat{u}_k(x,y) = 0 \qquad \text{for} \quad k = 1,\ldots,t. \tag{6.2}$$

Equation (6.2) can be written in matrix form as

$$S_k\left(\hat{f}(x,y),\hat{g}(x,y)\right)\mathbf{x}_k = \mathbf{0}, \tag{6.3}$$

which has a non-trivial solution for $k = 1,\ldots,t$ and the solution vector $\mathbf{x}_k$ in (6.3) is given by

$$\mathbf{x}_k = \begin{bmatrix} \hat{\mathbf{v}}_t, & -\hat{\mathbf{u}}_t \end{bmatrix}^T,$$

where $\hat{\mathbf{v}}_t \in \mathbb{R}^{\binom{n-k+2}{2}}$ and $\hat{\mathbf{u}}_t \in \mathbb{R}^{\binom{m-k+2}{2}}$ are vectors containing the coefficients of the polynomials $\hat{u}_t(x,y)$ and $\hat{v}_t(x,y)$.

The matrix $S_k(\hat{f}(x,y),\hat{g}(x,y))$ in (6.3) is the kth order subresultant matrix for two bivariate polynomials in Bernstein form and is given by

$$S_k\left(\hat{f}(x,y),\hat{g}(x,y)\right) = \left[\; C_{n-k}\left(\hat{f}(x,y)\right) \;\middle|\; C_{m-k}\left(\hat{g}(x,y)\right) \;\right], \tag{6.4}$$

where $C_{n-k}(\hat{f}(x,y))$ and $C_{m-k}(\hat{g}(x,y))$ are bivariate convolution matrices as defined in (2.33). Alternatively, the kth subresultant matrix is given by

$$S_k\left(\hat{f}(x,y),\hat{g}(x,y)\right) = D^{-1}_{m+n-k} T_k\left(\hat{f}(x,y),\hat{g}(x,y)\right)\hat{Q}_k.$$

The matrices D^{-1}_{m+n-k}, $T_k(\hat{f}(x,y),\hat{g}(x,y))$ and $\hat{Q}_k$ defined here are different from those used in the definition for the subresultant matrices of univariate polynomials. The block diagonal matrix D^{-1}_{m+n-k} of order $\binom{m+n-k+2}{2}$ is given by

$$D^{-1}_{m+n-k} = \mathrm{diag}\left[\; \binom{m+n-k}{0,0} \;\middle|\; \binom{m+n-k}{1,0} \quad \binom{m+n-k}{0,1} \;\middle|\; \ldots \;\middle|\; \binom{m+n-k}{m+n-k,0} \quad \ldots \quad \binom{m+n-k}{0,m+n-k} \;\right]. \tag{6.5}$$

The matrix $T_k(\hat{f}(x,y),\hat{g}(x,y)) \in \mathbb{R}^{\binom{m+n-k+2}{2}\times\left(\binom{m-k+2}{2}+\binom{n-k+2}{2}\right)}$ is given by

$$T_k\left(\hat{f}(x,y),\hat{g}(x,y)\right) = \left[\; T_{n-k}\left(\hat{f}(x,y)\right) \;\middle|\; T_{m-k}\left(\hat{g}(x,y)\right) \;\right],$$

where $T_{n-k}(\hat{f}(x,y)) \in \mathbb{R}^{\binom{m+n-k+2}{2}\times\binom{n-k+2}{2}}$ and $T_{m-k}\left(\hat{g}(x,y)\right) \in \mathbb{R}^{\binom{m+n-k+2}{2}\times\binom{m-k+2}{2}}$ have the same structure as the matrix defined in (2.29). The block diagonal matrix $\hat{Q}_k$ of order $\left(\binom{n-k+2}{2}+\binom{m-k+2}{2}\right)$ is given by

$$\hat{Q}_k = \mathrm{diag}\left[\; Q_{n-k}, \quad Q_{m-k} \;\right],$$

where Q_{n-k} and Q_{m-k} are of the same form as the matrix Q_n defined in (2.36).

The degree of the GCD of two bivariate polynomials over a triangular domain can

[(i)] Note that there is more than one polynomial $\hat{d}_k(x,y)$ of degree k which satisfies (6.2) and the set $\{\hat{d}_k(x,y)\}$ should be read as $\{\{\hat{d}_1(x,y)\},\{\hat{d}_2(x,y)\},\ldots,\hat{d}_t(x,y)\}$.

be computed in a similar way to the univariate two-polynomial problem (Section 3.2). Suppose that $\hat{f}(x,y)$ and $\hat{g}(x,y)$ have a GCD $\hat{d}_t(x,y)$ of degree t, then

$$\begin{aligned}
\text{rank } S_k\left(\hat{f}(x,y),\hat{g}(x,y)\right) &< \binom{m+n-k+2}{2} \qquad \text{for } k=1,\ldots,t,\\
\text{rank } S_k\left(\hat{f}(x,y),\hat{g}(x,y)\right) &= \binom{m+n-k+2}{2} \qquad \text{for } k=t,\ldots,\min(m,n).
\end{aligned}$$

Therefore, the degree of the GCD is given by the index of the last numerically rank deficient subresultant matrix.

Alternatively, the first subresultant matrix $S_1(\hat{f}(x,y),\hat{g}(x,y))$ has $\binom{t+1}{2}$ numerically zero singular values and any two-polynomial subresultant matrix $S_{t-p}(\hat{f}(x,y),\hat{g}(x,y))$ for $p \leq t$ has $\binom{p+2}{2}$ numerically zero singular values. Therefore, the degree of the GCD can be computed by the numerical rank loss of the first (or any subsequent) subresultant matrix. It has, however, been shown for univariate polynomial GCD problems that consideration of the numerical rank or each subresultant matrix in the sequence gives a more reliable method of GCD degree determination. A method for the construction of the sequence of subresultant matrices is described in Appendix B.2.

Having described how the degree of the GCD of two bivariate polynomials can be computed, an extension to the three-polynomial problem is now described.

6.2.2 The Degree of the GCD of Three Polynomials

Suppose now that $\hat{f}(x,y)$, $\hat{g}(x,y)$ and a third polynomial $\hat{h}(x,y)$ have a common divisor $\hat{d}_k(x,y)$ of degree k, then

$$\frac{\hat{f}(x,y)}{\hat{u}_k(x,y)} = \hat{d}_k(x,y), \quad \frac{\hat{g}(x,y)}{\hat{v}_k(x,y)} = \hat{d}_k(x,y) \quad \text{and} \quad \frac{\hat{h}(x,y)}{\hat{w}_k(x,y)} = \hat{d}_k(x,y).$$

This gives rise to three equations, where the first is given in (6.2) and two new equations are given by

$$\hat{f}(x,y)\hat{w}_k(x,y) - \hat{h}(x,y)\hat{u}_k(x,y) = 0 \tag{6.6}$$

$$\hat{h}(x,y)\hat{v}_k(x,y) - \hat{g}(x,y)\hat{w}_k(x,y) = 0. \tag{6.7}$$

The set of three equations (6.2, 6.6, 6.7) can be written in matrix form as

$$\tilde{S}_k\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)\mathbf{x}_k = \mathbf{0}, \tag{6.8}$$

which has a non-trivial solution for $k=1,\ldots,t$ and the vector $\mathbf{x}_k$ contains coefficients of the cofactor polynomials $\hat{u}_k(x,y)$, $\hat{v}_k(x,y)$ and $\hat{w}_k(x,y)$ and is given by

$$\mathbf{x}_k = \begin{bmatrix} \hat{\mathbf{v}}_k, & \hat{\mathbf{w}}_k, & -\hat{\mathbf{u}}_k \end{bmatrix}^T. \tag{6.9}$$

The matrix $\tilde{S}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ is the (3×3) partitioned kth subresultant matrix of three bivariate polynomials in Bernstein form and is given by

$$\tilde{S}_k\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)=\tilde{D}_k^{-1}\tilde{T}_k\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)\tilde{Q}_k.$$

The block diagonal matrix $\tilde{D}_k^{-1}$ of order $\left(\binom{m+n-k+2}{2}+\binom{m+o-k+2}{2}+\binom{n+o-k+2}{2}\right)$ is given by

$$\tilde{D}_k^{-1}=\operatorname{diag}\left[\begin{array}{ccc} D_{m+n-k}^{-1}, & D_{m+o-k}^{-1}, & D_{n+o-k}^{-1}\end{array}\right],$$

where D_{m+n-k}^{-1} is defined in (6.5) and D_{m+o-k}^{-1} and D_{n+o-k}^{-1} are of the same structure. The matrix $\tilde{T}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ is given by

$$\left[\begin{array}{ccc} T_{n-k}\left(\hat{f}(x,y)\right) & & T_{m-k}\left(\hat{g}(x,y)\right)\\ & T_{o-k}\left(\hat{f}(x,y)\right) & T_{m-k}\left(\hat{h}(x,y)\right)\\ T_{n-k}\left(\hat{h}(x,y)\right) & -T_{o-k}\left(\hat{g}(x,y)\right) & \end{array}\right], \tag{6.10}$$

where the matrix partitions of the form $T_{n-k}(\hat{f}(x,y))$ are Toeplitz-like and have the same structure as (2.35). The block diagonal matrix $\tilde{Q}_k$ of order $(m+n+o-3k+3)$ is given by

$$\tilde{Q}_k=\operatorname{diag}\left[\begin{array}{ccc} Q_{n-k}, & Q_{o-k}, & Q_{m-k}\end{array}\right], \tag{6.11}$$

where the partitions Q_{n-k}, Q_{o-k} and Q_{m-k} are diagonal matrices of the same structure as the matrix Q_n defined in (2.36).

The kth (3×3) partitioned subresultant matrix $\tilde{S}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ can be written in terms of its row-partitions and is given by

$$\left[\begin{array}{c}\mathcal{R}_{a,k}\\ \mathcal{R}_{b,k}\\ \mathcal{R}_{c,k}\end{array}\right]=\left[\begin{array}{ccc} C_{n-k}\left(\hat{f}(x,y)\right) & & C_{m-k}\left(\hat{g}(x,y)\right)\\ & C_{o-k}\left(\hat{f}(x,y)\right) & C_{m-k}\left(\hat{h}(x,y)\right)\\ C_{n-k}\left(\hat{h}(x,y)\right) & -C_{o-k}\left(\hat{g}(x,y)\right) & \end{array}\right], \tag{6.12}$$

where each of the row-partitions are defined accordingly

$$\mathcal{R}_{a,k}=\left[\begin{array}{ccc} C_{n-k}\left(\hat{f}(x,y)\right) & 0_{\binom{m+n-k+2}{2},\binom{o-k+2}{2}} & C_{m-k}\left(\hat{g}(x,y)\right)\end{array}\right], \tag{6.13}$$

$$\mathcal{R}_{b,k}=\left[\begin{array}{ccc} 0_{\binom{m+o-k+2}{2},\binom{n-k+2}{2}} & C_{o-k}\left(\hat{f}(x,y)\right) & C_{m-k}\left(\hat{h}(x,y)\right)\end{array}\right], \tag{6.14}$$

$$\mathcal{R}_{c,k}=\left[\begin{array}{ccc} C_{n-k}\left(\hat{h}(x,y)\right) & -C_{o-k}\left(\hat{g}(x,y)\right) & 0_{\binom{n+o-k+2}{2},\binom{m-k+2}{2}}\end{array}\right]. \tag{6.15}$$

The matrices $C_{n-k}(\hat{f}(x,y))$ in (6.12) and $T_{n-k}(\hat{f}(x,y))$ in (6.10) are related by the equation

$$C_{n-k}\left(\hat{f}(x,y)\right)=D_{m+n-k}^{-1}T_{n-k}\left(\hat{f}(x,y)\right)Q_{n-k} \tag{6.16}$$

and $C_{n-k}(\hat{f}(x,y))$ was originally defined in (2.33).

Alternatively, two of the three equations (6.2, 6.6, 6.7) are sufficient to describe the

system completely and the third can be derived given the other two. This gives rise to three variations of the (2×3) partitioned subresultant matrices.

Variation 1 : The equations (6.2) and (6.6) can be written in matrix form as

$$\hat{S}_k\left(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y)\right) \mathbf{x}_{k,1} = \mathbf{0}, \tag{6.17}$$

which has non-zero solutions for $k = 1, \ldots, t$ and the vector $\mathbf{x}_{k,1}$ is of the same structure as $\mathbf{x}_k$ defined in (6.9). The matrix $\hat{S}_k(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))$ is given by

$$\begin{bmatrix} \mathcal{R}_{a,k} \\ \mathcal{R}_{b,k} \end{bmatrix} = \begin{bmatrix} C_{n-k}\left(\hat{f}(x,y)\right) & & C_{m-k}\left(\hat{g}(x,y)\right) \\ & C_{o-k}\left(\hat{f}(x,y)\right) & C_{m-k}\left(\hat{h}(x,y)\right) \end{bmatrix},$$

where $\mathcal{R}_{a,k}$ and $\mathcal{R}_{b,k}$ are the row-partitions defined in (6.13) and (6.15) respectively.

Variation 2 : The two equations (6.2) and (6.7) are written in matrix form as

$$S_k\left(\hat{g}(x,y), \hat{f}(x,y), \hat{h}(x,y)\right) \mathbf{x}_{k,2} = \mathbf{0}, \tag{6.18}$$

which has non-zero solutions for $k = 1, \ldots, t$ and the vector $\mathbf{x}_{k,2}$ is given by reordering the row-partitions of $\mathbf{x}_{k,1}$

$$\mathbf{x}_{k,2} = \begin{bmatrix} \hat{\mathbf{u}}_k, & \hat{\mathbf{w}}_k, & -\hat{\mathbf{v}}_k \end{bmatrix}^T.$$

The matrix $\hat{S}_k(\hat{g}(x,y), \hat{f}(x,y), \hat{h}(x,y))$ is given by

$$\begin{bmatrix} \tilde{\mathcal{R}}_{a,k} \\ \tilde{\mathcal{R}}_{c,k} \end{bmatrix} = \begin{bmatrix} C_{m-k}\left(\hat{g}(x,y)\right) & & C_{n-k}\left(\hat{f}(x,y)\right) \\ & C_{o-k}\left(\hat{g}(x,y)\right) & C_{n-k}\left(\hat{h}(x,y)\right) \end{bmatrix}$$

and $\tilde{\mathcal{R}}_{a,k}$ and $\tilde{\mathcal{R}}_{c,k}$ are variations of $\mathcal{R}_{a,k}$ and $\mathcal{R}_{c,k}$ defined in (6.13) and (6.15).

Variation 3 : Equations (6.6) and (6.7) can be written in matrix form as

$$S_k\left(\hat{h}(x,y), \hat{g}(x,y), \hat{f}(x,y)\right) \mathbf{x}_{k,3} = \mathbf{0}, \tag{6.19}$$

which has non-trivial solutions for $k = 1, \ldots, t$ and the vector $\mathbf{x}_{k,3}$ is given by reordering the row-partitions of $\mathbf{x}_{k,1}$

$$\mathbf{x}_{k,3} = \begin{bmatrix} \hat{\mathbf{u}}_k, & \hat{\mathbf{v}}_k, & -\hat{\mathbf{w}}_k \end{bmatrix}^T.$$

The matrix $S_k(\hat{h}(x,y), \hat{g}(x,y), \hat{f}(x,y))$ is the third variation of the (2×3) partitioned subresultant matrix and is given by

$$\begin{bmatrix} \tilde{\mathcal{R}}_{b,k} \\ \tilde{\mathcal{R}}_{c,k} \end{bmatrix} = \begin{bmatrix} C_{m-k}\left(\hat{h}(x,y)\right) & & C_{o-k}\left(\hat{f}(x,y)\right) \\ & C_{n-k}\left(\hat{h}(x,y)\right) & C_{o-k}\left(\hat{g}(x,y)\right) \end{bmatrix},$$

where $\tilde{\mathcal{R}}_{a,k}$ and $\tilde{\mathcal{R}}_{c,k}$ are variations of $\mathcal{R}_{a,k}$ and $\mathcal{R}_{c,k}$ respectively.

Since there are three variations of the (2×3) partitioned subresultant matrix, the notation $\hat{S}_k(\hat{f}^*(x,y), \hat{g}^*(x,y), \hat{h}^*(x,y))$ is used to denote that the polynomials $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ can be considered in any order. Each ordering gives one of the three variations of subresultant matrix already discussed.

The degree t of the GCD is given by the index of the last rank deficient subresultant matrix in the sequence of subresultant matrices $\tilde{S}_k(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))$ and $\hat{S}_k(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))$

$$\text{rank}\left(\tilde{S}_k(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))\right) < \binom{m+n-k+2}{2} + \binom{m+o-k+2}{2} + \binom{n+o-k+2}{2}$$
$$\text{for } k = 1, \ldots, t,$$
$$\text{rank}\left(\hat{S}_k(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))\right) < \binom{m+n-k+2}{2} + \binom{m+o-k+2}{2}$$
$$\text{for } k = 1, \ldots, t.$$

The pairwise GCDs of the polynomials $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ are defined as

$$\hat{d}_a(x,y) = \text{GCD}\left(\hat{f}(x,y), \hat{g}(x,y)\right)$$
$$\hat{d}_b(x,y) = \text{GCD}\left(\hat{f}(x,y), \hat{h}(x,y)\right)$$
$$\hat{d}_c(x,y) = \text{GCD}\left(\hat{g}(x,y), \hat{h}(x,y)\right)$$

and each of the matrix equations (6.17, 6.18, 6.19) can be thought of as simultaneously solving two of the three pairwise two-polynomial GCD problems.

Problems due to scaling of the three-polynomial subresultant matrices for univariate polynomials carry over to the bivariate problem. This is exacerbated by the trinomial terms found in the subresultant matrices of bivariate polynomials, as shown in the following example.

Example 6.2.1. Consider the Bernstein form of the exact polynomials $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ of degree $m = 14$, $n = 14$ and $o = 14$ respectively. The factorised forms of $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ are given by

$$\hat{f}(x,y) = (x-0.72)(x-0.52)^2(x+0.75)(y-0.75)^2(y-0.15)(y^2-1.7)(x+y-0.5)^5$$
$$\hat{g}(x,y) = (x-0.72)(x-0.52)^2(x-0.192)(y-0.15)(x+y-0.5)^5(y^2-1.7)(x^2+y^2+0.7)$$
$$\hat{h}(x,y) = (x-1.91987)^4(x-0.72)(y-0.15)(y^2-1.7)(x^2+y^2-0.34)^3,$$

whose GCD $\hat{d}_t(x,y)$ of degree $t = 4$ is given by

$$\hat{d}_t(x,y) = (x-0.72)(y-0.15)(y^2-1.7).$$

The polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ have a GCD $\hat{d}_a(x,y)$ of degree $t_a = 11$ which is given by

$$\hat{d}_a(x,y) = (x-0.72)(y-0.15)(y^2-1.7)(x-0.52)^2(x+y-0.5)^5$$

and $\hat{d}_b(x,y) = \hat{d}_c(x,y) = \hat{d}_t(x,y)$. The coefficients of the exact polynomials $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ are multiplied by 10^5, 10^5 and 10^{-5} respectively. Noise is added to

the coefficients of $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ such that the coefficients of the inexact polynomials $f(x,y)$, $g(x,y)$ and $h(x,y)$ are given by

$$\begin{aligned} a_{i_1,i_2} &= \hat{a}_{i_1,i_2} + \hat{a}_{i_1,i_2}\,\epsilon_{f,i_1,i_2}\,r_{f,i_1,i_2} && \text{for} \quad i_1+i_2 = 0,\ldots,m, \\ b_{j_1,j_2} &= \hat{b}_{j_1,j_2} + \hat{b}_{j_1,j_2}\,\epsilon_{g,j_1,j_2}\,r_{g,j_1,j_2} && \text{for} \quad j_1+j_2 = 0,\ldots,n, \\ c_{p_1,p_2} &= \hat{c}_{p_1,p_2} + \hat{c}_{p_1,p_2}\,\epsilon_{g,p_1,p_2}\,r_{h,p_1,p_2} && \text{for} \quad p_1+p_2 = 0,\ldots,o, \end{aligned} \tag{6.20}$$

where $\{\epsilon_{f,i_1,i_2}\}$ $\{\epsilon_{g,j_1,j_2}\}$ and $\{\epsilon_{h,p_1,p_2}\}$ are uniformly distributed random variables in the interval $[10^{-8}, 10^{-6}]$. The values $\{r_{f,i_1,i_2}\}$, $\{r_{g,j_1,j_2}\}$ and $\{r_{h,p_1,p_2}\}$ are uniformly distributed random variables in the interval $[-1,1]$. The coefficients of the inexact polynomials are plotted in Figure 6.1, where it is shown that the coefficients of $h(x,y)$ are significantly smaller than those of $f(x,y)$ or $g(x,y)$.

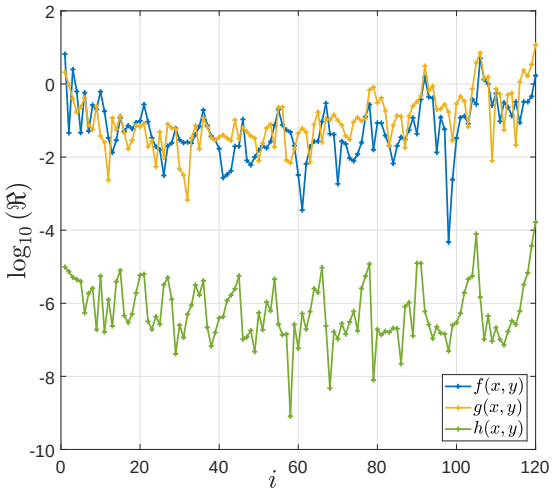
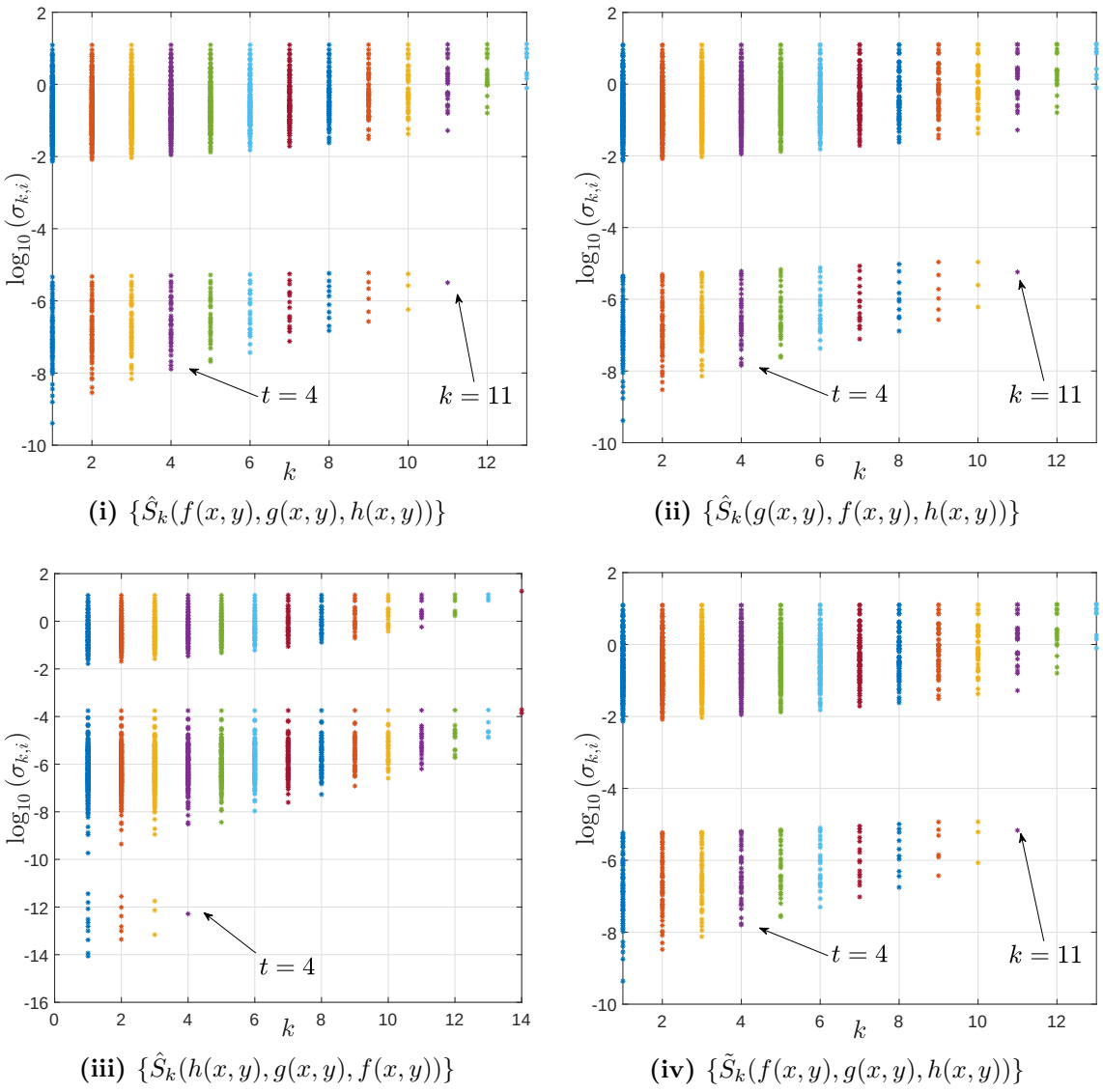


Figure 6.1: The coefficients of $f(x,y)$, $g(x,y)$ and $h(x,y)$ in Example 6.2.1

The singular values of the three variations of the (2×3) partitioned subresultant matrices (i) $\{\hat{S}_k(f(x,y), g(x,y), h(x,y))\}$, (ii) $\{\hat{S}_k(g(x,y), f(x,y), h(x,y))\}$ and (iii) $\{\hat{S}_k(h(x,y), g(x,y), h(x,y))\}$ and the (3×3) subresultant matrices $\{\tilde{S}_k(f(x,y), g(x,y), h(x,y))\}$ are plotted in Figures 6.2i to 6.2iv.

There is a large separation between the numerically zero and non-zero singular values of the subresultant matrices $\{\hat{S}_k(f(x,y), g(x,y), h(x,y))\}$ (Figure 6.2i), $\{\hat{S}_k(g(x,y), f(x,y), h(x,y))\}$ (Figure 6.2ii) and $\{\tilde{S}_k(f(x,y), g(x,y), h(x,y))\}$ (Figure 6.2iv). The numerically zero singular values span the interval $[10^{-10}, 10^{-5}]$ while the non-zero singular values span the interval $[10^{-2}, 10^{2}]$. The results in both graphs suggest that the 11th subresultant matrix is the last singular matrix, and therefore the degree of the AGCD is given by $t = 11$. However, this is incorrect and is due to the coefficients of $f(x,y)$ and $g(x,y)$ being significantly larger than the coefficients of $h(x,y)$ such that the matrices $C_{m-k}(h(x,y))$ and $C_{n-k}(h(x,y))$ appear to be zero-like in the subresultant matrices when considered next to the matrices containing the coefficients of $f(x,y)$ and $g(x,y)$.

The degree of the AGCD is given by $t = 4$, which is correctly determined by the minimum singular values of $\{\hat{S}_k(h(x,y), g(x,y), f(x,y))\}$ (Figure 6.2iii). There are two separations amongst the singular values for this set of subresultant matrices. The first separation defined by the interval $[10^{-11}, 10^{-12}]$ correctly identifies the separation of the



(i) $\{\hat{S}_k(f(x,y), g(x,y), h(x,y))\}$ **(ii)** $\{\hat{S}_k(g(x,y), f(x,y), h(x,y))\}$

(iii) $\{\hat{S}_k(h(x,y), g(x,y), f(x,y))\}$ **(iv)** $\{\tilde{S}_k(f(x,y), g(x,y), h(x,y))\}$

Figure 6.2: The singular values $\{\sigma_{k_1,k_2}\}$ of the unprocessed subresultant matrices (i)$\{\hat{S}_k(f(x,y), g(x,y), h(x,y))\}$, (ii)$\{\hat{S}_k(g(x,y), f(x,y), h(x,y))\}$, (iii)$\{\hat{S}_k(h(x,y), g(x,y), f(x,y))\}$ and (iv)$\{\tilde{S}_k(f(x,y), g(x,y), h(x,y))\}$ in Example 6.2.1

numerically zero and non-zero singular values. The second separation between 10^{-4} and 10^{-2} arises due to poor scaling of the columns within the row-partitions. This example is revisited in Section 6.6 (Example 6.6.3), where it is shown that the degree of the AGCD is correctly identified by analysis of the singular values of the preprocessed subresultant matrices.

□

6.3 Variants of the Subresultant Matrices

6.3.1 The Two-Polynomial Subresultant Matrices

Several variants of the sequence of subresultant matrices of two univariate polynomials were described in Section 3.1.4. Now, the variants of the subresultant matrices of two bivariate polynomials are considered with similar results. The entries of these variants differ from the univariate subresultant matrices and entries for the first partition of each

variant are now described:

1. The first variant of the kth subresultant matrix is given by $T_k(\hat{f}(x,y),\hat{g}(x,y))$ and the non-zero entries of the first partition $T_{n-k}(\hat{f}(x,y))$ are given by

$$\hat{a}_{i_1,i_2}\binom{m}{i_1,i_2} \quad \text{for} \quad i_1+i_2=0,\ldots,m.$$

   for each column $j_1+j_2=0,\ldots,n-k$.

2. The second variant of the kth subresultant matrix is given by $D^{-1}_{m+n-k}T_k(\hat{f}(x,y),\hat{g}(x,y))$ and the non-zero entries of the first partition $D^{-1}_{m+n-k}T_{n-k}(\hat{f}(x,y))$ are given by

$$\frac{\hat{a}_{i_1,i_2}\binom{m}{i_1,i_2}}{\binom{m+n-k}{i_1+j_1,i_2+j_2}} \quad \text{for} \quad i_1+i_2=0,\ldots,m; \quad j_1+j_2=0,\ldots,n-k.$$

3. The third variant of the kth subresultant matrix is given by $T_k(\hat{f}(x,y),\hat{g}(x,y))\hat{Q}_k$ and the non-zero entries of the first partition $T_{n-k}(\hat{f}(x,y))Q_{n-k}$ are given by

$$\hat{a}_{i_1,i_2}\binom{m}{i_1,i_2}\binom{n-k}{j_1,j_2} \quad \text{for} \quad i_1+i_2=0,\ldots,m; \quad j_1+j_2=0,\ldots,n-k.$$

4. The fourth subresultant matrix variant is given by $D^{-1}_{m+n-k}T_k(\hat{f}(x,y),\hat{g}(x,y))\hat{Q}_k$ and the non-zero entries of the first partition $D^{-1}_{m+n-k}T_{n-k}(\hat{f}(x,y))Q_{n-k}$ are given by

$$\frac{\hat{a}_{i_1,i_2}\binom{m}{i_1,i_2}\binom{n-k}{j_1,j_2}}{\binom{m+n-k}{i_1+j_1,i_2+j_2}} \quad \text{for} \quad i_1+i_2=0,\ldots,m; \quad j_1+j_2=0,\ldots,n-k.$$

As with the variants of the two-polynomial subresultant matrices for univariate polynomials, the rank of the kth subresultant matrix for each variant is theoretically equal. However, scaling due to the presence of potentially large trinomial terms can cause numerical issues, and the entries of the partitions of the kth subresultant matrix may be of significantly different magnitudes. This can cause spurious results in the analysis of the SVD.

6.3.2 The Three-Polynomial Subresultant Matrices

The partitioned structure of the kth (3×3) subresultant matrix was described in Section 6.2.2 where it was defined as the product of three matrices, $\tilde{D}_k^{-1}$, $\tilde{T}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ and $\tilde{Q}_k$. The four variants of the kth (3×3) partitioned subresultant matrix are therefore given by

(i) $\{\tilde{T}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))\}$

(ii) $\{\tilde{D}_k^{-1}\tilde{T}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))\}$

(iii) $\{\tilde{T}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))\tilde{Q}_k\}$

(iv) $\{\tilde{D}_k^{-1}\tilde{T}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))\tilde{Q}_k\}$

The kth (3×3) partitioned subresultant matrix consists of six non-zero partitions and three zero partitions as defined in (6.12). The entries contained in the six non-zero partitions of these subresultant matrix variants have an equivalent structure to the variants defined for the two-polynomial subresultant matrices in Section 6.3.1.

The (2×3) partitioned subresultant matrices $\hat{S}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ are similarly defined

$$\hat{S}_k\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)=\hat{D}_k^{-1}\hat{T}_k\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)\tilde{Q}_k.$$

The block diagonal matrix $\hat{D}_k^{-1}$ of order $(\binom{m+n-k+2}{2}+\binom{n+o-k+2}{2})$ is given by

$$\hat{D}_k^{-1}=\operatorname{diag}\left[\begin{array}{ccc} D_{m+n-k}^{-1}, & D_{m+o-k}^{-1} \end{array}\right].$$

The matrix $\hat{T}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ consists of Toeplitz matrices and is given by

$$\left[\begin{array}{ccc} T_{n-k}\left(\hat{f}(x,y)\right) & & T_{m-k}\left(\hat{g}(x,y)\right) \\ & T_{o-k}\left(\hat{f}(x,y)\right) & T_{m-k}\left(\hat{h}(x,y)\right) \end{array}\right]$$

and the block diagonal matrix $\tilde{Q}_k$ of order $\binom{n-k+2}{2}+\binom{o-k+2}{2}+\binom{m-k+2}{2}$ is given by

$$\tilde{Q}_k=\operatorname{diag}\left[\begin{array}{ccc} Q_{n-k}, & Q_{o-k}, & Q_{m-k} \end{array}\right].$$

The variants of the (2×3) subresultant matrices are therefore given by

(i) $\{\hat{T}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))\}$

(ii) $\{\hat{D}_k^{-1}\hat{T}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))\}$

(iii) $\{\hat{T}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))\tilde{Q}_k\}$

(iv) $\{\hat{D}_k^{-1}\hat{T}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))\tilde{Q}_k\}$

The entries in the partitions of these variants are of the same structure as those found in Section 6.3.1.

Example 6.3.1. In this example, the singular values of the two-polynomial subresultant matrix variants (i) $\{T_k(\hat{f}(x,y),\hat{g}(x,y))\}$, (ii) $\{D_{m+n-k}^{-1}T_k(\hat{f}(x,y),\hat{g}(x,y))\}$, (iii) $\{T_k(\hat{f}(x,y),\hat{g}(x,y))\hat{Q}_k\}$ and (iv) $\{D_{m+n-k}^{-1}T_k(\hat{f}(x,y),\hat{g}(x,y))\hat{Q}_k\}$ are analysed to determine the optimal variant for use in the computation of the degree of the GCD.

Consider the Bernstein form of the exact polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$, whose fac-

torisations are given by

$$\begin{aligned}\hat{f}(x,y) &= (x-1.126479841321)^5(x-0.8365498798)^3(x+0.145487821)^{10}\\ &\quad (y-0.2564878)^4(x+y-0.16546978321)^2(x+y+1.5679814354)^3\\ &\quad (x^2+y^2-0.46549871232156)\\ \hat{g}(x,y) &= (x-1.126479841321)^5(x-0.8365498798)^3(y-0.45489789123123)\\ &\quad (x+y-0.16546978321)^2(x+y-0.35648979126321)^3(x+y+1.5679814354)^3\\ &\quad (x^2+y^2-0.46549871232156),\end{aligned}$$

and whose GCD $\hat{d}_t(x,y)$ of total degree $t=15$ has the factorisation given by

$$\begin{aligned}\hat{d}_t(x,y) &= (x-1.126479841321)^5(x-0.8365498798)^3(x+y-0.16546978321)^2\\ &\quad (x+y+1.5679814354)^3(x^2+y^2-0.46549871232156).\end{aligned}$$

Noise is added to the coefficients of $\hat{f}(x,y)$ and $\hat{g}(x,y)$ such that the coefficients of the inexact polynomials are given by

$$a_{i_1,i_2} = \hat{a}_{i_1,i_2} + \hat{a}_{i_1,i_2}\left(r_{f,i_1,i_2}\times 10^{-5}\right) \quad \text{and} \quad b_{j_1,j_2} = \hat{b}_{j_1,j_2} + \hat{b}_{j_1,j_2}\left(r_{g,j_1,j_2}\times 10^{-5}\right),$$

where $\{r_{f,i_1,i_2}\}$ and $\{r_{g,j_1,j_2}\}$ are uniformly distributed random variables in the interval $[-1,1]$.

The SVD of the set of subresultant matrices for each of the four variants (i) $\{T_k(f(x,y),g(x,y))\}$, (ii) $\{D^{-1}_{m+n-k}T_k(f(x,y),g(x,y))\}$, (iii) $\{T_k(f(x,y),g(x,y))\hat{Q}_k\}$ and (iv) $\{D^{-1}_{m+n-k}T_k(f(x,y),g(x,y))\hat{Q}_k\}$ are computed, and the singular values $\{\sigma_{k,i}\}$ of each variant are plotted in Figures 6.3i to 6.3iv. The following observations can be made:

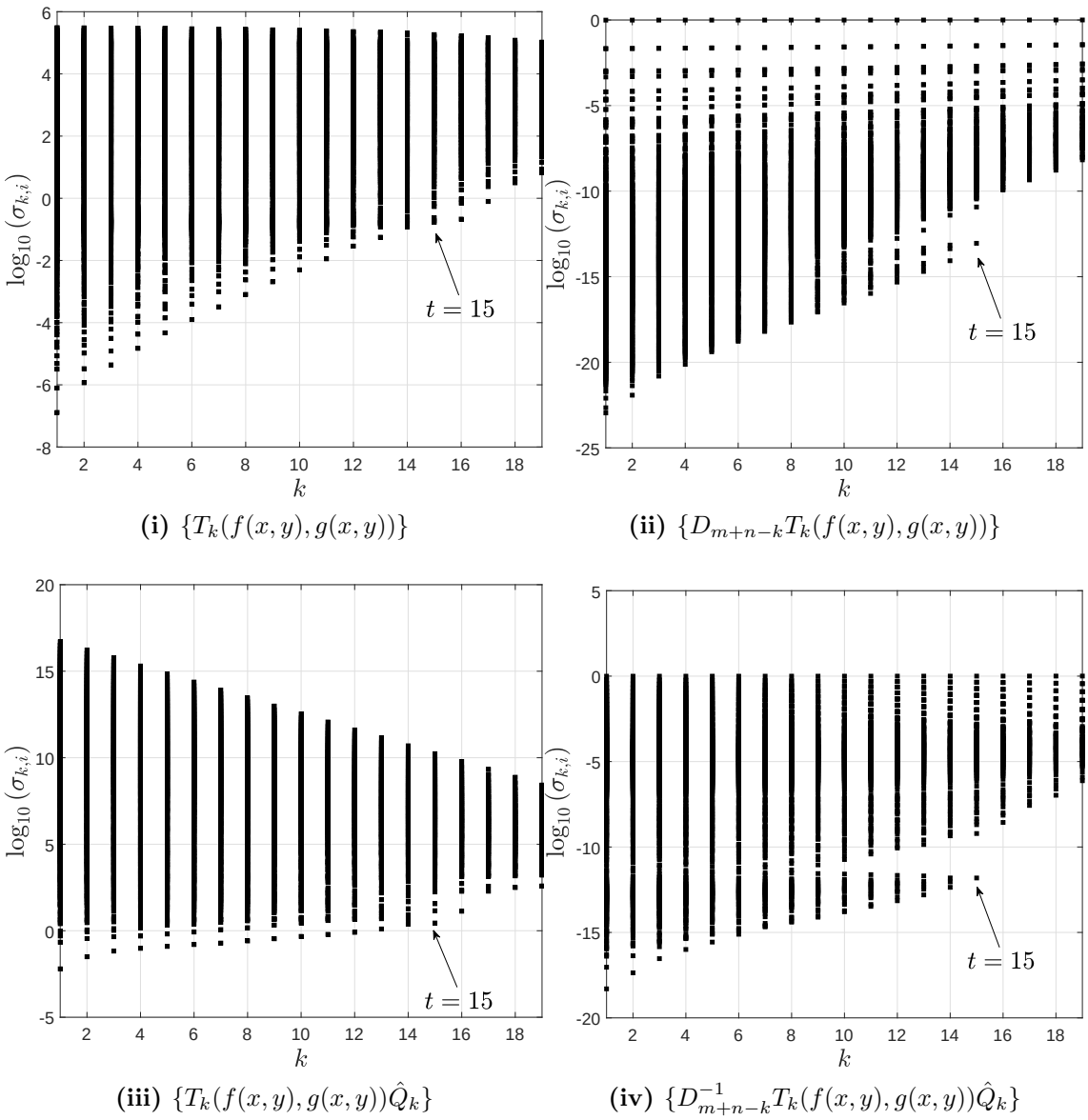
1. In Figures 6.3i and 6.3iii there is no discernible separation between the numerically zero and non-zero singular values of the subresultant matrices $\{T_k(f(x,y),g(x,y))\}$ and $\{T_k(f(x,y),g(x,y))\hat{Q}_k\}$.

2. In Figure 6.3iv there is a significant separation between the numerically zero and non-zero singular values of $S_{10}(f(x,y),g(x,y),\ldots,S_{15}(f(x,y),g(x,y)$. From Figure 6.3iv, the degree of the AGCD is correctly determined to be equal to $t=15$.

3. In Figure 6.3ii a similar pattern emerges amongst the singular values $\{\sigma_{k,i}\}$ of $\{D^{-1}_{m+n-k}T_k(f(x,y),g(x,y))\}$, but fewer of the subresultant matrices in the sequence have the required separation between their numerically zero and non-zero singular values. For subresultant matrices where this separation is present, it is considerably smaller than the separation found between the numerically zero and non-zero singular values of the subresultant matrices $\{D^{-1}_{m+n-k}T_k(f(x,y),g(x,y))\hat{Q}_k\}$.

4. There is no distinct separation between the numerically zero and non-zero singular values of the first subresultant matrix of any of the four variants. Thus, the degree of the AGCD cannot be determined from the rank loss of the first subresultant matrix.

The coefficients of the cofactor polynomials $\hat{u}_t(x,y)$ and $\hat{v}_t(x,y)$ and the GCD $\hat{d}_t(x,y)$ can be approximated by a least squares based method that will be described in Sec-



(i) $\{T_k(f(x,y),g(x,y))\}$ **(ii)** $\{D_{m+n-k}T_k(f(x,y),g(x,y))\}$

(iii) $\{T_k(f(x,y),g(x,y))\hat{Q}_k\}$ **(iv)** $\{D^{-1}_{m+n-k}T_k(f(x,y),g(x,y))\hat{Q}_k\}$

Figure 6.3: The singular values $\{\sigma_{k,i}\}$ of the unprocessed subresultant matrices in Example 6.3.1

tion 6.5. This method is a trivial extension of the method described for the univariate problem. The distances between the exact polynomials and the two sets of approximations obtained by the least squares method using (i) $D^{-1}_{m+n-t}T_t(f(x,y),g(x,y))$ and (ii) $D^{-1}_{m+n-t}T_t(f(x,y),g(x,y))\hat{Q}_t$ are computed and given in Table 6.1. Note that the approximations of $\hat{u}_t$, $\hat{v}_t$ and $\hat{d}_t$ were not computed from the tth subresultant matrix of the form $T_k(f(x,y),g(x,y))$ or $T_k(f(x,y),g(x,y))\hat{Q}_k$ since the degree of the AGCD was not correctly determined from the corresponding sets of subresultant matrices.

The table shows that the sets of approximations return errors which are of the same order of magnitude as the initial noise added to the exact polynomials, but $D^{-1}_{m+n-t}T_t(f(x,y),g(x,y))\hat{Q}_t$ offers slightly better results.

	$T_k(f,g)$	$D^{-1}_{m+n-k}T_k(f,g)$	$T_k(f,g)\hat{Q}_k$	$D^{-1}_{m+n-k}T_k(f,g)\hat{Q}_k$
Error $\hat{u}_t(x,y)$	-	$7.397585e-05$	-	$1.704812e-05$
Error $\hat{v}_t(x,y)$	-	$4.069999e-04$	-	$1.245633e-04$
Error $\hat{d}_t(x,y)$	-	$1.574557e-04$	-	$9.827083e-05$
Average	-	$1.679890e-04$	-	$5.931488e-05$

Table 6.1: Error in the approximations of $\hat{u}_t(x,y)$, $\hat{v}_t(x,y)$ and $\hat{d}_t(x,y)$ in Example 6.3.1

□

In Example 6.3.1 the set of subresultant matrices $\{D^{-1}_{m+n-k}T_k(\hat{f}(x,y),\hat{g}(x,y))\hat{Q}_k\}$ had the largest separation between the numerically zero and non-zero singular values and this result is typical for many examples. Therefore, this is the most appropriate set of subresultant matrices for the computation of the degree of the AGCD. This variant and the variant $D^{-1}_{m+n-k}T_k(\hat{f}(x,y),\hat{g}(x,y))$ also gave the best approximations of the coefficients of cofactor polynomials and the GCD.

As the upper bound of the noise is increased, the separation between the numerically zero and non-zero singular values decreases. However, preprocessing the polynomials in each subresultant matrix typically allows for higher levels of noise to be applied before the separation can no longer be identified. These preprocessing operations are now considered.

6.4 Preprocessing of the Bivariate Subresultant Matrices

In Section 6.3 it was shown that the sequences of subresultant matrices $\{D^{-1}_{m+n-k}T_k(\hat{f}(x,y),\hat{g}(x,y))\hat{Q}_k\}$ and $\{\tilde{D}^{-1}_{m+n-k}\tilde{T}_k(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))\tilde{Q}_k\}$ exhibited the best relative scaling of the entries in their partitions when compared with the other variants of two-polynomial and three-polynomial subresultant matrices. These variant also had maximum separation between their numerically zero and non-zero singular values which were used in the computation of the degree of the GCD. However, the ratio of the entry of maximum magnitude to entry of minimum magnitude in these subresultant matrices may still be large.

As in Section 3.4, preprocessing the polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ will be shown to yield improved results in the computation of the degree and coefficients of their GCD. The three preprocessing operations were defined in Section 3.4 and modifications are now considered.

Normalisation by Arithmetic or Geometric Means in the Two-Polynomial Subresultant Matrices

The geometric means of the non-zero entries in $C_{n-k}(\hat{f}(x,y))$ and $C_{m-k}(\hat{g}(x,y))$ are denoted $\mathcal{G}_{n-k}(\hat{f}(x,y))$ and $\mathcal{G}_{m-k}(\hat{g}(x,y))$ respectively.

The geometric mean of the non-zero entries of $C_{n-k}(\hat{f}(x,y))$ is given by

$$\mathcal{G}_{n-k}\left(\hat{f}(x,y)\right) = \prod_{j_1+j_2=0}^{n-k}\prod_{i_1+i_2=0}^{m}\left(\hat{a}_{i_1,i_2}\frac{\binom{m}{i_1,i_2}\binom{n-k}{j_1,j_2}}{\binom{m+n-k}{i_1+j_1,i_2+j_2}}\right)^{\frac{1}{\binom{m+2}{2}\times\binom{n-k+2}{2}}} \tag{6.21}$$

and a similar expression is given for $\mathcal{G}_{m-k}(\hat{g}(x,y))$.

Methods for the fast computation of the arithmetic and geometric means of the non-zero entries in the $(n-k)$th order convolution matrix, $C_{n-k}(\hat{f}(x))$, are given in Appendix C.2.1 and Appendix C.2.2 respectively and the normalised polynomials $\bar{f}_k(x,y)$ and $\bar{g}_k(x,y)$ are given by

$$\bar{f}_k(x,y) = \sum_{i_1+i_2=0}^{m} \bar{a}_{i_1,i_2} B^m_{i_1,i_2}(x,y) \qquad \text{where} \qquad \bar{a}_{i_1,i_2} = \frac{\hat{a}_{i_1,i_2}}{\mathcal{G}_{n-k}\left(\hat{f}(x,y)\right)}$$

$$\bar{g}_k(x,y) = \sum_{i_1+i_2=0}^{n} \bar{b}_{i_1,i_2} B^n_{i_1,i_2}(x,y) \qquad \text{where} \qquad \bar{b}_{i_1,i_2} = \frac{\hat{b}_{i_1,i_2}}{\mathcal{G}_{m-k}\left(\hat{g}(x,y)\right)}.$$

Normalisation by Geometric Means in the (3×3) Partitioned Subresultant Matrices

Normalisation of the polynomials $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ in the (3×3) partitioned subresultant matrices of the form $\tilde{S}_k(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))$ is given by a simple extension of the method described for the normalisation of two polynomials in the two-polynomial subresultant matrices.

Since each of the three polynomials appear in two partitions, the normalised polynomials are given by

$$\bar{f}_k(x,y) = \frac{\hat{f}(x,y)}{\hat{\mathcal{G}}_k\left(\hat{f}(x,y)\right)}, \quad \bar{g}_k(x,y) = \frac{\hat{g}(x,y)}{\hat{\mathcal{G}}_k\left(\hat{g}(x,y)\right)} \quad \text{and} \quad \bar{h}_k(x,y) = \frac{\hat{h}(x,y)}{\hat{\mathcal{G}}_k\left(\hat{h}(x,y)\right)},$$

where $\hat{\mathcal{G}}_k\left(\hat{f}(x,y)\right)$ is given by

$$\hat{\mathcal{G}}_k\left(\hat{f}(x,y)\right) = \left(\prod_{j_1+j_2=0}^{n-k} \prod_{i_1+i_2=0}^{m} \frac{\hat{a}_{i_1,i_2}\binom{m}{i_1,i_2}\binom{n-k}{j_1,j_2}}{\binom{m+n-k}{i_1+j_1,i_2+j_2}} \prod_{j_1+j_2=0}^{o-k} \prod_{i_1+i_2=0}^{m} \frac{\hat{a}_{i_1,i_2}\binom{m}{i_1,i_2}\binom{o-k}{j_1,j_2}}{\binom{m+o-k}{i_1+j_1,i_2+j_2}} \right)^{\frac{1}{r+c}} \tag{6.22}$$

and

$$r = \binom{m+n-k+2}{2}\binom{n-k+2}{2} \quad \text{and} \quad c = \binom{m+o-k+2}{2}\binom{o-k+2}{2}.$$

Normalisation by Geometric Means in the (2×3) Partitioned Subresultant Matrices

The coefficients of the polynomials $\hat{g}(x,y)$ and $\hat{h}(x,y)$ appear in one partition each of the (2×3) partitioned subresultant matrix $\hat{S}_k(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))$. The geometric means of the non-zero entries containing the coefficients of $\hat{g}(x,y)$ and $\hat{h}(x,y)$ in the kth subresultant matrix are given by $\mathcal{G}_{m-k}(\hat{g}(x,y))$ and $\mathcal{G}_{m-k}(\hat{h}(x,y))$ respectively. These follow from (6.21). However, the coefficients of $\hat{f}(x,y)$ appear in two partitions of $\hat{S}_k(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))$ and therefore $\hat{f}(x,y)$ is normalised by the geometric mean of its entries in both partitions.

The normalised polynomials $\bar{f}_k(x,y)$, $\bar{g}_k(x,y)$ and $\bar{h}_k(x,y)$ are given by

$$\bar{f}_k(x,y) = \frac{\hat{f}(x,y)}{\hat{\mathcal{G}}_k\left(\hat{f}(x,y)\right)}, \quad \bar{g}_k(x,y) = \frac{\hat{g}(x,y)}{\mathcal{G}_{m-k}\left(\hat{g}(x,y)\right)} \quad \text{and} \quad \bar{h}_k(x,y) = \frac{\hat{h}(x,y)}{\mathcal{G}_{m-k}\left(\hat{h}(x,y)\right)},$$

where $\hat{\mathcal{G}}_k(\hat{f}(x,y))$ is defined in (6.22).

The Optimisation Problem for the Two-Polynomial Subresultant Matrix

The GCD of two polynomials is defined to within an arbitrary scalar, so the polynomial $\bar{f}_k(x,y)$ is multiplied by the scalar $\lambda \in \mathbb{R}$, where the optimal value of λ, denoted λ_k, is determined for each preprocessed subresultant matrix S_k such that the magnitude of entries in both partitions is similar.

The two independent variables x and y are replaced by $x = \theta_1\omega_1$ and $y = \theta_2\omega_2$, where ω_1 and ω_2 are the new independent variables and $\theta_1, \theta_2 \in \mathbb{R}$ are optimally chosen to minimise the ratio of entry of maximum magnitude to entry of minimum magnitude in the kth subresultant matrix S_k.

Let $\lambda\ddot{f}(\theta_1,\theta_2,\omega_1,\omega_2)$ and $\ddot{g}(\theta_1,\theta_2,\omega_1,\omega_2)$ denote the unoptimised polynomials given by

$$\lambda\ddot{f}_k(\theta_1,\theta_2,\omega_1,\omega_2) = \lambda \sum_{i_1+i_2=0}^{m} \bar{a}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{m}{i_1,i_2}(1-\theta_1\omega_1-\theta_2\omega_2)^{m-i_1-i_2}\,\omega_1^{i_1}\omega_2^{i_2}, \quad (6.23)$$

$$\ddot{g}_k(\theta_1,\theta_2,\omega_1,\omega_2) = \sum_{i_1+i_2=0}^{n} \bar{b}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{n}{i_1,i_2}(1-\theta_1\omega_1-\theta_2\omega_2)^{n-i_1-i_2}\,\omega_1^{i_1}\omega_2^{i_2}. \quad (6.24)$$

Let the sets of non-zero entries in the first and second partitions of the kth unprocessed subresultant matrix be denoted $\mathscr{P}_{1,k}(\lambda,\theta_1,\theta_2)$ and $\mathscr{P}_{2,k}(\theta_1,\theta_2)$, which are given by

$$\mathscr{P}_{1,k}(\lambda,\theta_1,\theta_2) = \left\{ \frac{\left|\lambda\bar{a}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{m}{i_1,i_2}\binom{n-k}{j_1,j_2}\right|}{\binom{m+n-k}{i_1+j_1,i_2+j_2}} \;\middle|\; i_1+i_2=0,\ldots,m;\ j_1+j_2=0,\ldots,n-k \right\} \quad (6.25)$$

$$\mathscr{P}_{2,k}(\theta_1,\theta_2) = \left\{ \frac{\left|\bar{b}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{n}{i_1,i_2}\binom{m-k}{j_1,j_2}\right|}{\binom{m+n-k}{i_1+j_1,i_2+j_2}} \;\middle|\; i_1+i_2=0,\ldots,n;\ j_1+j_2=0,\ldots,m-k \right\}. \quad (6.26)$$

The optimal values λ, θ_1 and θ_2 are given by solutions of the minimisation problem

$$(\lambda_k,\theta_{1,k},\theta_{2,k}) = \arg\min_{\lambda,\theta_1,\theta_2} \left\{ \frac{\max\{\max\{\mathscr{P}_{1,k}(\lambda,\theta_1,\theta_2)\},\max\{\mathscr{P}_{2,k}(\theta_1,\theta_2)\}\}}{\min\{\min\{\mathscr{P}_{1,k}(\lambda,\theta_1,\theta_2)\},\min\{\mathscr{P}_{2,k}(\theta_1,\theta_2)\}\}} \right\}.$$

This minimisation is solved by a linear programming problem described in Appendix C.2.3. It follows that the subresultant matrices of the preprocessed polynomials, given by $S_k(\lambda_k\tilde{f}_k(\omega_1,\omega_2),\tilde{g}_k(\omega_1,\omega_2)) = D_{m+n-k}^{-1}T_k(\lambda_k\tilde{f}_k(\omega_1,\omega_2),\tilde{g}_k(\omega_1,\omega_2))\hat{Q}_k$ for $k = 1,\ldots,\min(m,n)$, are used to compute the degree of an AGCD of the inexact polyno-

mials, where

$$\lambda_k \tilde{f}_k(\omega_1,\omega_2) = \lambda_k \sum_{i_1+i_2=0}^{m} \bar{a}_{i_1,i_2}\theta_{1,k}^{i_1}\theta_{2,k}^{i_2}\binom{m}{i_1,i_2}(1-\theta_{1,k}\omega_1-\theta_{2,k}\omega_2)^{m-i_1-i_2}\omega_1^{i_1}\omega_2^{i_2}$$

$$\tilde{g}_k(\omega_1,\omega_2) = \sum_{i_1+i_2=0}^{n} \bar{b}_{i_1,i_2}\theta_{1,k}^{i_1}\theta_{2,k}^{i_2}\binom{n}{i_1,i_2}(1-\theta_{1,k}\omega_1-\theta_{2,k}\omega_2)^{n-i_1-i_2}\omega_1^{i_1}\omega_2^{i_2}.$$

The Optimisation Problem for the Three-Polynomial Subresultant Matrix

The second and third preprocessing operations for the (2×3) partitioned subresultant matrix follow directly from the second and third preprocessing operations for the two-polynomial subresultant matrices. A new variable ρ is introduced, which scales the polynomial $\hat{h}(x,y)$. The unoptimised polynomials $\ddot{f}_k(\theta_1,\theta_2,\omega_1,\omega_2)$ and $\ddot{g}_k(\theta_1,\theta_2,\omega_1,\omega_2)$ are defined in (6.23) and (6.24) respectively, and the unoptimised polynomial $\ddot{h}_k(\theta_1,\theta_2,\omega_1,\omega_2)$ is given by

$$\ddot{h}_k(\theta_1,\theta_2,\omega_1,\omega_2) = \sum_{i_1+i_2=0}^{o} \bar{c}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{o}{i_1,i_2}(1-\theta_1\omega_1-\theta_2\omega_2)^{o-i_1-i_2}\omega_1^{i_1}\omega_2^{i_2}.$$

The kth unoptimised subresultant matrix is given by

$$\begin{bmatrix} C_{n-k}\left(\lambda\ddot{f}_k(\theta_1,\theta_2,\omega_1,\omega_2)\right) & & C_{m-k}\left(\ddot{g}_k(\theta_1,\theta_2,\omega_1,\omega_2)\right) \\ & C_{o-k}\left(\lambda\ddot{f}_k(\theta_1,\theta_2,\omega_1,\omega_2)\right) & C_{m-k}\left(\rho\ddot{h}_k(\theta_1,\theta_2,\omega_1,\omega_2)\right) \end{bmatrix}.$$

The optimal values θ_1, θ_2, λ_k and ρ_k minimise the ratio of entry of maximum magnitude to entry of minimum magnitude in the kth subresultant matrix.

The sets of non-zero entries in the first two partitions, denoted $\mathscr{P}_{1,k}(\lambda,\theta_1,\theta_2,)$ and $\mathscr{P}_{2,k}(\theta_1,\theta_2)$, are already defined in (6.25) and (6.26). The sets of non-zero entries in the third and fourth non-zero partitions are given by $\mathscr{P}_{3,k}(\lambda,\theta_1,\theta_2)$ and $\mathscr{P}_{4,k}(\theta_1,\theta_2)$, where

$$\mathscr{P}_{3,k}(\lambda,\theta_1,\theta_2) = \left\{ \frac{\left|\lambda\bar{a}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{m}{i_1,i_2}\binom{o-k}{j_1,j_2}\right|}{\binom{m+o-k}{i_1+j_1,i_2+j_2}} \;\middle|\; i_1+i_2=0,\ldots,m;\ j_1+j_2=0,\ldots,o-k \right\}$$

$$\mathscr{P}_{4,k}(\rho,\theta_1,\theta_2) = \left\{ \frac{\left|\rho\bar{c}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{o}{i_1,i_2}\binom{m-k}{j_1,j_2}\right|}{\binom{m+o-k}{i_1+j_1,i_2+j_2}} \;\middle|\; i_1+i_2=0,\ldots,o;\ j_1+j_2=0,\ldots,m-k \right\}$$

such that the minimisation problem can be written as

$$(\lambda_k,\rho_k,\theta_1,\theta_2) = \arg\min_{\lambda,\rho,\theta_1,\theta_2} \left\{ \frac{\max\left\{\max\{\mathscr{P}_{1,k}(\lambda,\theta_1,\theta_2)\},\max\{\mathscr{P}_{2,k}(\theta_1,\theta_2)\},\right.}{\min\left\{\min\{\mathscr{P}_{1,k}(\lambda,\theta_1,\theta_2)\},\min\{\mathscr{P}_{2,k}(\theta_1,\theta_2)\},\right.} \right.$$
$$\left. \left. \frac{\max\{\mathscr{P}_{3,k}(\lambda,\theta_1,\theta_2)\},\max\{\mathscr{P}_{4,k}(\rho,\theta_1,\theta_2)\}\}}{\min\{\mathscr{P}_{3,k}(\lambda,\theta_1,\theta_2)\},\min\{\mathscr{P}_{4,k}(\rho,\theta_1,\theta_2)\}\}} \right\}. \qquad (6.27)$$

The minimisation problem for determining the optimal values of λ, ρ, θ_1 and θ_2 in the (3×3) partitioned subresultant matrices is a simple extension of the (2×3) partitioned

subresultant matrix problem.

$$\begin{bmatrix} C_{n-k}\left(\lambda\ddot{f}_k(\theta_1,\theta_2,\omega_1,\omega_2)\right) & & C_{m-k}\left(\ddot{g}_k(\theta_1,\theta_2,\omega_1,\omega_2)\right) \\ & C_{o-k}\left(\lambda\ddot{f}_k(\theta_1,\theta_2,\omega_1,\omega_2)\right) & C_{m-k}\left(\rho\ddot{h}_k(\theta_1,\theta_2,\omega_1,\omega_2)\right) \\ C_{n-k}\left(\lambda\ddot{h}_k(\theta_1,\theta_2,\omega_1,\omega_2)\right) & -C_{o-k}\left(\ddot{g}_k(\theta_1,\theta_2,\omega_1,\omega_2)\right) & \end{bmatrix}.$$

The sets of non-zero entries in $C_{n-k}\left(\lambda\ddot{h}_k(\theta_1,\theta_2,\omega_1,\omega_2)\right)$ and $C_{o-k}\left(\ddot{g}_k(\theta_1,\theta_2,\omega_1,\omega_2)\right)$ are given by $\mathscr{P}_5(\rho,\theta_1,\theta_2)$ and $\mathscr{P}_6(\theta_1,\theta_2)$, where

$$\mathscr{P}_{5,k}(\rho,\theta_1,\theta_2) = \left\{ \frac{\left|\rho\bar{c}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{o}{i_1,i_2}\binom{n-k}{j_1,j_2}\right|}{\binom{n+o-k}{i_1+j_1,i_2+j_2}} \;\middle|\; i_1+i_2=0,\ldots,o;\ j_1+j_2=0,\ldots,n-k \right\}$$

$$\mathscr{P}_{6,k}(\theta_1,\theta_2) = \left\{ \frac{\left|\bar{b}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{n}{i_1,i_2}\binom{o-k}{j_1,j_2}\right|}{\binom{n+o-k}{i_1+j_1,i_2+j_2}} \;\middle|\; i_1+i_2=0,\ldots,n;\ j_1+j_2=0,\ldots,o-k \right\}.$$

The minimisation problem (6.27) is extended to

$$(\lambda_k,\rho_k,\theta_1,\theta_2) = \arg\min_{\lambda,\rho,\theta_1,\theta_2} \left\{ \begin{array}{l} \dfrac{\max\{\max\{\mathscr{P}_{1,k}(\lambda,\theta_1,\theta_2)\},\max\{\mathscr{P}_{2,k}(\theta_1,\theta_2)\}\}}{\min\{\min\{\mathscr{P}_{1,k}(\lambda,\theta_1,\theta_2)\},\min\{\mathscr{P}_{2,k}(\theta_1,\theta_2)\}\}}, \\ \dfrac{\max\{\mathscr{P}_{3,k}(\lambda,\theta_1,\theta_2)\},\max\{\mathscr{P}_{4,k}(\rho,\theta_1,\theta_2)\}}{\min\{\mathscr{P}_{3,k}(\lambda,\theta_1,\theta_2)\},\min\{\mathscr{P}_{4,k}(\rho,\theta_1,\theta_2)\}} \\ \dfrac{\max\{\mathscr{P}_{5,k}(\rho,\theta_1,\theta_2)\},\max\{\mathscr{P}_{6,k}(\theta_1,\theta_2)\}\}}{\min\{\mathscr{P}_{5,k}(\rho,\theta_1,\theta_2)\},\min\{\mathscr{P}_{6,k}(\theta_1,\theta_2)\}\}} \end{array} \right\}.$$

This minimisation problem is reduced to a linear programming problem which is described in Appendix C.2.4 and the values λ_k, ρ_k, $\theta_{1,k}$ and $\theta_{2,k}$ are retrieved from the solution of this linear programming problem. The preprocessed polynomials $\lambda_k\tilde{f}_k(\omega_1,\omega_2)$, $\tilde{g}_k(\omega_1,\omega_2)$ and $\rho_k\tilde{h}_k(\omega_1,\omega_2)$ are therefore given by

$$\lambda_k\tilde{f}_k(\omega_1,\omega_2) = \lambda_k \sum_{i_1+i_2=0}^{m} \bar{a}_{i_1,i_2}\theta_{1,k}^{i_1}\theta_{2,k}^{i_2}\binom{m}{i_1,i_2}(1-\theta_{1,k}\omega_1-\theta_{2,k}\omega_2)^{m-i_1-i_2}\omega_1^{i_1}\omega_2^{i_2},$$

$$\tilde{g}_k(\omega_1,\omega_2) = \sum_{i_1+i_2=0}^{n} \bar{b}_{i_1,i_2}\theta_{1,k}^{i_1}\theta_{2,k}^{i_2}\binom{n}{i_1,i_2}(1-\theta_{1,k}\omega_1-\theta_{2,k}\omega_2)^{n-i_1-i_2}\omega_1^{i_1}\omega_2^{i_2},$$

$$\rho_k\tilde{h}_k(\omega_1,\omega_2) = \rho_k \sum_{i_1+i_2=0}^{o} \bar{c}_{i_1,i_2}\theta_{1,k}^{i_1}\theta_{2,k}^{i_2}\binom{o}{i_1,i_2}(1-\theta_{1,k}\omega_1-\theta_{2,k}\omega_2)^{o-i_1-i_2}\omega_1^{i_1}\omega_2^{i_2}.$$

This section has considered preprocessing of the polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ in the two-polynomial problem, and $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ in the three-polynomial problem. The corresponding preprocessed subresultant matrices contain entries where the ratio of entry of maximum magnitude to entry of minimum magnitude is minimised. Preprocessing gives greater separation between the numerically zero and non-zero singular values of the set of preprocessed subresultant matrices when compared with equivalent unprocessed matrices.

As with the univariate GCD problem, improved approximations of the cofactor polynomials are also obtained when preprocessed polynomials are considered. Results will be saved until after the discussion on methods used to approximate the coefficients of the

cofactor polynomials.

6.5 Approximating the Coefficients of the Cofactor Polynomials and the GCD

Computing Coefficients of the Cofactor Polynomials and the AGCD in the Two Polynomial Problem

Given t, the total degree of the AGCD, the coefficients of the cofactor polynomials $\hat{u}_t(x,y)$ and $\hat{v}_t(x,y)$ can be approximated from the tth subresultant matrix and this is similar to the equivalent problem for two univariate polynomials in Section 3.5.1. The tth subresultant matrix is near rank deficient, so

$$S_t(f,g)\mathbf{x}_t \approx 0 \tag{6.28}$$

has a non-zero solution vector $\mathbf{x}_t$. The definitions of f and g are deliberately not specified as this method applies to both (i) the unprocessed inexact polynomials $f(x,y)$ and $g(x,y)$ and (ii) the preprocessed polynomials $\tilde{f}_t(\omega_1,\omega_2)$ and $\alpha_t\tilde{g}_t(\omega_1,\omega_2)$. Since $S_t(f,g)$ has a rank deficiency of one, a column $\mathbf{c}_{t,q}$ of $S_t(f,g)$ nearly lies in the space spanned by the remaining columns. That is,

$$A_{t,q}(f,g)\mathbf{x}_{t,q} \approx \mathbf{c}_{t,q}. \tag{6.29}$$

The vector $\mathbf{x}_{t,q}$ can be computed by the least squares solution of (6.29), and the vector $\mathbf{x}_t$ in (6.28) is given by the insertion of '-1' into the qth position of the vector $\mathbf{x}_{t,q}$. The vector $\mathbf{x}_t$ contains coefficients the of the polynomials u_t and v_t. That is, $\mathbf{x}_t$ is given by

$$\mathbf{x}_t = \begin{bmatrix} \mathbf{v}_t, & -\mathbf{u}_t \end{bmatrix}^T,$$

where $\mathbf{v}_t \in \mathbb{R}^{\binom{n-t+2}{2}}$ and $\mathbf{u}_t \in \mathbb{R}^{\binom{m-t+2}{2}}$ are vectors of the coefficients of the approximations v_t and u_t. An approximation of the coefficients of GCD $\hat{d}(x,y)$ can then be computed as the least squares solution of

$$\begin{bmatrix} C_t(u_t) \\ C_t(v_t) \end{bmatrix} \mathbf{d}_t \approx \begin{bmatrix} \mathbf{f} \\ \mathbf{g} \end{bmatrix}. \tag{6.30}$$

Given that the coefficients of the approximations u_t, v_t and d_t are computed, they can be compared with the coefficients of the exact polynomials $\hat{u}_t(x,y)$, $\hat{v}_t(x,y)$ and $\hat{d}_t(x,y)$. This comparison is dependent on the definition of f, g and h.

Suppose that f and g represent unprocessed polynomials $f(x,y)$ and $g(x,y)$, then u_t, v_t and d_t represent the approximations $u_t(x,y)$, $v_t(x,y)$ and $d_t(x,y)$. The error in these approximations is given by

$$\epsilon_{u_t} = \hat{u}_t(x,y) - u_t(x,y), \quad \epsilon_{v_t} = \hat{v}_t(x,y) - v_t(x,y) \quad \text{and} \quad \epsilon_{d_t} = \hat{d}_t(x,y) - d_t(x,y). \tag{6.31}$$

Suppose now that f and g represent the preprocessed polynomials $\tilde{f}_t(\omega_1, \omega_2)$ and $\alpha_t \tilde{g}_t(\omega_1, \omega_2)$ then the approximations u_t, v_t and d_t represent polynomials $\tilde{u}_t(\omega_1, \omega_2)$, $\tilde{v}_t(\omega_1, \omega_2)$ and $\tilde{d}_t(\omega_1, \omega_2)$. By the substitutions $\omega_1 = x/\theta_1$ and $\omega_2 = y/\theta_2$, the polynomials $\tilde{u}_t(x, y)$, $\tilde{v}_t(x, y)$ and $\tilde{d}_t(x, y)$ are given and the error in these approximations is given by

$$\epsilon_{\tilde{u}_t} = \hat{u}_t(x, y) - \tilde{u}_t(x, y), \quad \epsilon_{\tilde{v}_t} = \hat{v}_t(x, y) - \tilde{v}_t(x, y) \quad \text{and} \quad \epsilon_{\tilde{d}_t} = \hat{d}_t(x, y) - \tilde{d}_t(x, y). \tag{6.32}$$

Computing the Cofactor Polynomials and the GCD by Least Squares in the Three Polynomial Problem

By extension of the method described above, the coefficients of the polynomials $\hat{u}_t(x, y)$, $\hat{v}_t(x, y)$, $\hat{w}_t(x, y)$ and $\hat{d}_t(x, y)$ can be approximated. The three variations of the (2×3) partitioned subresultant matrices and the (3×3) partitioned subresultant matrix are numerically rank deficient, so

$$\tilde{S}_t(f, g, h)\mathbf{x}_t \approx 0 \quad \text{and} \quad \hat{S}_t(f, g, h)\mathbf{x}_t \approx 0 \tag{6.33}$$

have non-trivial solutions. The columns $\tilde{\mathbf{c}}_{t,q}$ and $\hat{\mathbf{c}}_{t,q}$ almost lie in the space spanned by the remaining columns of $\tilde{S}_t(f, g, h)$ and $\hat{S}_t(f, g, h)$ respectively, with a minimal residual, so

$$\tilde{A}_t(f, g, h)\tilde{\mathbf{x}}_{t,q} \approx \tilde{\mathbf{c}}_{t,q} \qquad \text{and} \qquad \hat{A}_t(f, g, h)\hat{\mathbf{x}}_{t,q} \approx \hat{\mathbf{c}}_{t,q},$$

where $\tilde{\mathbf{x}}_{t,q}$ and $\tilde{\mathbf{x}}_{t,q}$ are found by simple least squares. The vectors $\tilde{\mathbf{x}}_t$ and $\hat{\mathbf{x}}_t$ are given by the insertion of '-1' into the qth position of $\tilde{\mathbf{x}}_{t,q}$ and $\hat{\mathbf{x}}_{t,q}$ respectively, and the coefficients of u_t, v_t and w_t are contained within the vectors

$$\tilde{\mathbf{x}}_t = \hat{\mathbf{x}}_t = \begin{bmatrix} \mathbf{v}_t, & \mathbf{w}_t, & -\mathbf{u}_t \end{bmatrix}^T.$$

Given an alternative variation of the (2×3) partitioned subresultant matrix, either $\hat{S}_t(g, f, h)$ or $\hat{S}_t(h, g, f)$, the order of the vectors $\mathbf{v}_t$, $\mathbf{w}_t$ and $\mathbf{u}_t$ would be rearranged according to either (6.18) or (6.19).

Toefficients of the approximation d_t are given by the least squares solution of

$$\begin{bmatrix} C_t(u_t) \\ C_t(v_t) \\ C_t(w_t) \end{bmatrix} \mathbf{d}_t \approx \begin{bmatrix} \mathbf{f} \\ \mathbf{g} \\ \mathbf{h} \end{bmatrix}.$$

The errors in the approximations u_t, v_t, w_t and d_t are computed according to the error measure defined in (6.31) and (6.32).

The method of SNTLN was used in Section 3.5.2 for the computation of the low rank approximation of the tth subresultant matrix of two univariate polynomials in Bernstein form. From the low rank approximation, approximations of the coefficients of cofactor polynomials and the GCD were considerably more accurate than the approximations obtained by the least squares based method. It is expected that an equivalent method would

give better approximations of the cofactor polynomials in the bivariate two or three-polynomial problem. However, the approximations obtained by the least squares based method are sufficient to show how preprocessing the subresultant matrices yields improved results when compared with unprocessed subresultant matrices.

The extension of the SNTLN method would require a significant amount of new work, more suited to future research. The main focus of this chapter was the computation of the degree of the GCD of two or three bivariate polynomials in Bernstein form, with experiments in variations of the subresultant matrices. These extensions are significantly more interesting than extensions of methods for the computation of the coefficients of the GCD.

6.6 Results

Examples of the two-polynomial problem will now be considered, followed by examples for the three-polynomial problem. These results will show that the numerical rank of each of the preprocessed subresultant matrices is better defined than those of the equivalent unprocessed subresultant matrices. The separation between the numerically zero and non-zero singular values of the preprocessed matrices is typically larger than the separation of the singular values of the unprocessed subresultant matrices.

Two-Polynomial Problems

Example 6.6.1. Consider the Bernstein form of the exact polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$, whose factorisations are given by

$$\hat{f}(x,y) = (x-0.46549)^5(x+0.11156)^6(x+0.16551)^4(y-0.24687)^2(x^2+y^2-0.16579)^3$$
$$\hat{g}(x,y) = (x-0.35465)^3(x+0.11156)^6(y-0.46546)(y-0.24687)^2$$

and whose GCD $\hat{d}_t(x,y)$ of degree $t=15$ given by

$$\hat{d}(x,y) = (x-0.46549)^5(x+0.16551)^4(x^2+y^2-0.16579)^3.$$

The coefficients of $\hat{f}(x,y)$ and $\hat{g}(x,y)$, given by $\hat{a}_{i,j}$ and $\hat{b}_{i,j}$, are perturbed to give the inexact polynomials $f(x,y)$ and $g(x,y)$ whose coefficients are

$$a_{i_1,i_2} = \hat{a}_{i_1,i_2} + \hat{a}_{i_1,i_2} r_{f,i_1,i_2} \epsilon_{i_1,i_2} \qquad \text{and} \qquad b_{j_1,j_2} = \hat{b}_{j_1,j_2} + \hat{b}_{j_1,j_2} r_{g,j_1,j_2} \epsilon_{j_1,j_2}, \tag{6.34}$$

where $\{r_{f,i_1,i_2}\}$ and $\{r_{g,j_1,j_2}\}$ are uniformly distributed random variables in the interval $[-1,1]$ and $\{\epsilon_{f,i_1,i_2}\} = \{\epsilon_{g,j_1,j_2}\} = 10^{-6}$.

The inexact polynomials $f(x,y)$ and $g(x,y)$ are preprocessed as described in Section 6.4 to give the polynomials $\lambda_1\tilde{f}_1(\omega_1,\omega_2)$ and $\tilde{g}_1(\omega_1,\omega_2)$. In Figure 6.4 the coefficients of $f(x,y)$ and $g(x,y)$ span approximately 15 orders of magnitude, while coefficients of the preprocessed polynomials $\lambda_1\tilde{f}_1(\omega_1,\omega_2)$ and $\tilde{g}_1(\omega_1,\omega_2)$ span approximately 5 orders of magnitude. Also note that the sets of coefficients of $\lambda_1\tilde{f}_1(\omega_1,\omega_2)$ and $\tilde{g}_1(\omega_1,\omega_2)$ are of similar magnitude.

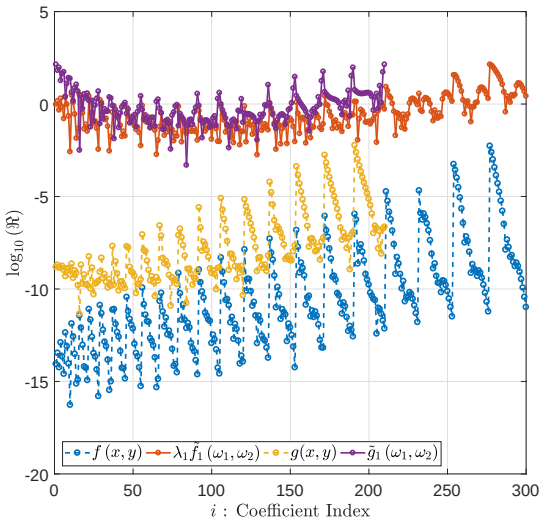


Figure 6.4: The coefficients of both the unprocessed polynomials $f(x, y)$ and $g(x, y)$ and the preprocessed polynomials $\lambda_1\tilde{f}_1(\omega_1, \omega_2)$ and $\tilde{g}_1(\omega_1, \omega_2)$ in Example 6.6.1

The DC1 and DC2 methods can be used to compute the degree of the AGCD using the singular values of unprocessed and preprocessed subresultant matrices.

The sets of singular values of the (i) unprocessed and (ii) preprocessed subresultant matrices are plotted in Figure 6.5i and Figure 6.5ii respectively and the following observations are made:

1. The degree of the AGCD can be computed using the minimum singular values of the unprocessed subresultant matrices (by DC2), which are plotted in Figure 6.5i. Let $\delta\dot{\rho}_i$ be defined as in (3.16), then $\delta\dot{\rho}_{15} = \dot{\rho}_{16} - \dot{\rho}_{15}$ is maximal in the set $\{\delta\dot{\rho}_i \mid i = 1, \ldots, \min(m, n)\}$. The degree of the AGCD is computed as $t = 15$.

2. The maximal change $\delta\dot{\rho}_{15}$ amongst the minimum singular values of the unprocessed subresultant matrices is significantly smaller than the maximal change $\delta\dot{\rho}_{15}$ amongst the minimum singular values of the preprocessed subresultant matrices.

3. When considering the complete set of singular values, there is no clear separation between the numerically zero and non-zero singular values $\{\sigma_{k,i}\}$ of the unprocessed subresultant matrices $\{S_k(f(x, y), g(x, y))\}$. However, there is a significant separation between the complete set of numerically zero and non-zero singular values $\{\sigma_{k,i}\}$ of the preprocessed subresultant matrices $\{S_k(\lambda_k\tilde{f}_k(\omega_1, \omega_2), \tilde{g}_k(\omega_1, \omega_2))\}$ in Figure 6.5ii, and by observation of the complete set, the degree of the AGCD is determined to be given by $t = 15$.

Given the degree of the AGCD, the coefficients of the cofactor polynomials and the GCD are approximated as described in (6.29) and (6.30) respectively. The unprocessed subresultant matrix $S_t(f(x, y), g(x, y))$ yields the approximations of the GCD triple $(u_t(x, y), v_t(x, y), d_t(x, y))$ and the preprocessed subresultant $S_t(\lambda_t\tilde{f}_t(\omega_1, \omega_2), \tilde{g}_t(\omega_1, \omega_2))$ yields the approximations $(\tilde{u}_t(\omega_1, \omega_2), \tilde{v}_t(\omega_1, \omega_2), \tilde{d}_t(\omega_1, \omega_2))$. The respective errors are measured as per (6.31) and (6.32) and plotted in Table 6.2, where it can be seen that

there is significantly less error in the approximations obtained by first preprocessing the polynomials.

	Without Preprocessing $u_t(x,y)$, $v_t(x,y)$ and $d_t(x,y)$	With Preprocessing $\tilde{u}_t(x,y)$, $\tilde{v}_t(x,y)$ and $\tilde{d}_t(x,y)$
Error $\hat{u}_t(x,y)$	$9.999988e-01$	$2.259597e-05$
Error $\hat{v}_t(x,y)$	$9.080576e-01$	$3.505760e-05$
Error $\hat{d}_t(x,y)$	$1.309569e+01$	$2.289066e-05$
Average	$2.282515e+00$	$2.627184e-05$

Table 6.2: Error in the approximations of $\hat{u}_t(x,y)$, $\hat{v}_t(x,y)$ and $\hat{d}_t(x,y)$, where $\{\epsilon_{f,i_1,i_2}\}$ and $\{\epsilon_{g,j_1,j_2}\}$ are set at 10^{-6} in Example 6.6.1

The level of additive noise is reduced such that $\{\epsilon_{f,i_1,i_2}\}$ and $\{\epsilon_{g,j_1,j_2}\}$ are uniformly distributed random variables in the interval $[1e-10, 1e-8]$ and the approximation errors are given in Table 6.3.

	Without Preprocessing $u_t(x,y)$, $v_t(x,y)$ and $d_t(x,y)$	With Preprocessing $\tilde{u}_t(x,y)$, $\tilde{v}_t(x,y)$ and $\tilde{d}_t(x,y)$
Error $\hat{u}_t(x,y)$	$1.003175e+00$	$2.938467e-07$
Error $\hat{v}_t(x,y)$	$4.176566e-02$	$2.996180e-07$
Error $\hat{d}_t(x,y)$	$4.499127e-02$	$3.076126e-07$
Average	$1.235307e-01$	$3.003062e-07$

Table 6.3: Error in the approximations of $\hat{u}_t(x,y)$, $\hat{v}_t(x,y)$ and $\hat{d}_t(x,y)$, where $\{\epsilon_{f,i_1,i_2}\}$ and $\{\epsilon_{g,j_1,j_2}\}$ are in the interval $[10^{-10}, 10^{-8}]$ in Example 6.6.1

At both noise levels, the approximations obtained from the tth preprocessed subresultant matrix are significantly better than those from the tth unprocessed subresultant matrix.

□

Example 6.6.2. Consider the Bernstein form of the exact polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$, whose factorisations are given by

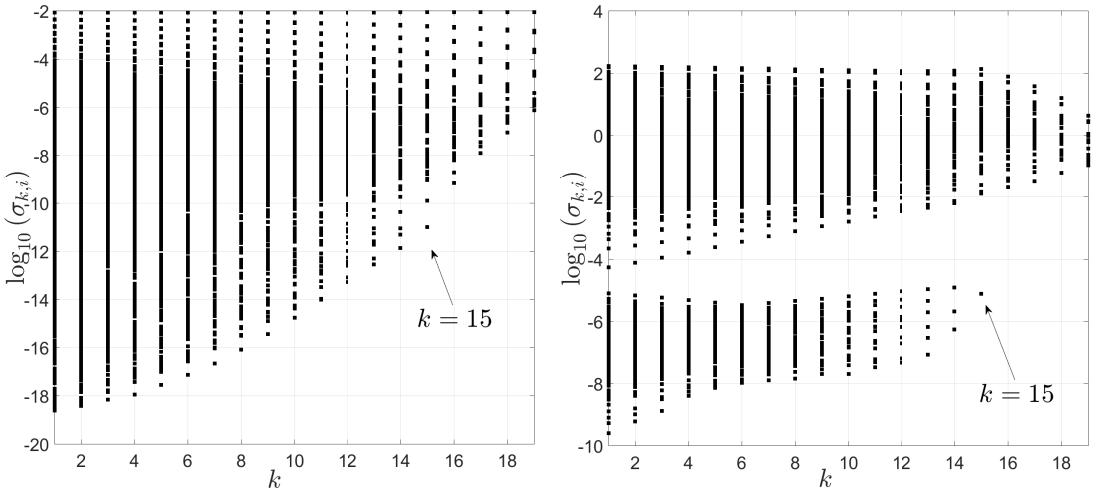
$$\hat{f}(x,y) = (x+0.56)\left(x^2+y^2+0.51\right)^2(x+y+1.12)^3(x+y+0.0124)^6$$
$$\hat{g}(x,y) = (x+0.56)\left(x^2+y^2+0.51\right)^2(x+y+1.12)^3(x+y+0.4512)^3$$

and whose GCD $\hat{d}_t(x,y)$ of degree $t=8$ is given by

$$\hat{d}_t(x,y) = (x+0.56)\left(x^2+y^2+0.51\right)^2(x+y+1.12)^3.$$

Noise is added to the coefficients of $\hat{f}(x,y)$ and $\hat{g}(x,y)$ such that the inexact polynomials $f(x,y)$ and $g(x,y)$ have coefficients given by

$$a_{i_1,i_2} = \hat{a}_{i_1,i_2} + \hat{a}_{i_1,i_2}(r_{i_1,i_2}\epsilon_f) \qquad \text{and} \qquad b_{j_1,j_2} = \hat{b}_{j_1,j_2} + \hat{b}_{j_1,j_2}(r_{g,j_1,j_2}\epsilon_g),$$



(i) The singular values $\{\sigma_{k,i}\}$ of the unprocessed subresultant matrices $\{S_k(f(x,y), g(x,y))\}$

(ii) The singular values $\{\sigma_{k,i}\}$ of the preprocessed subresultant matrices $\{S_k(\lambda_k \tilde{f}_k(\omega_1,\omega_2), \tilde{g}_k(\omega_1,\omega_2))\}$

Figure 6.5: The singular values $\{\sigma_{k,i}\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 6.6.1

where $\{r_{f,i_1,i_2}\}$ and $\{r_{g,j_1,j_2}\}$ are uniformly distributed random variables in the interval $[-1,1]$ and $\epsilon_f = \epsilon_g = 10^{-10}$.

From Figure 6.6, it can be seen that the coefficients of $f(x,y)$ span approximately 14 orders of magnitude, whereas the coefficients of the preprocessed polynomial $\lambda_1\tilde{f}_1(\omega_1,\omega_2)$ span only 4 orders of magnitude. Both $g(x,y)$ and $\tilde{g}_1(\omega_1,\omega_2)$ span approximately 4 orders of magnitude, but the coefficients of $\tilde{g}_1(\omega_1,\omega_2)$ are of the same order of magnitude as the coefficients of $\lambda_1\tilde{f}_1(\omega_1,\omega_2)$.

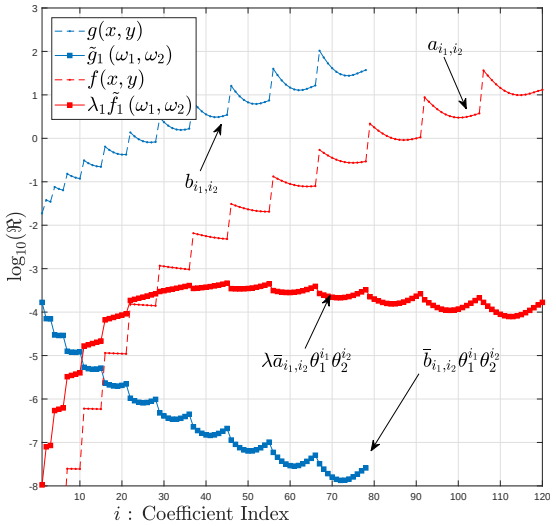
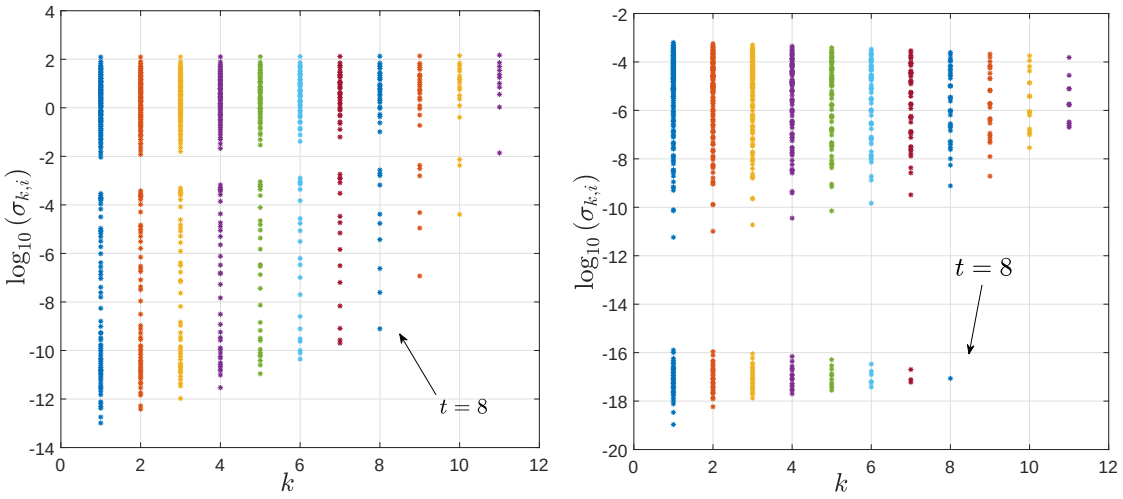


Figure 6.6: The coefficients of both the unprocessed polynomials $f(x,y)$ and $g(x,y)$ and the preprocessed polynomials $\lambda_1\tilde{f}_1(\omega_1,\omega_2)$ and $\tilde{g}_1(\omega_1,\omega_2)$ in Example 6.6.2

The singular values of the subresultant matrices of the unprocessed and preprocessed polynomials are plotted in Figure 6.7i and Figure 6.7ii respectively. There is no clear separation between the numerically zero and non-zero singular values of the unprocessed subresultant matrices, however the separation between the two sets of singular values is clearly defined for the preprocessed subresultant matrices.

Given that the degree of the AGCD is computed, approximations of the coefficients of cofactor polynomials and the GCD are computed and their errors are given in Table 6.4. Note that approximations were not computed from the unprocessed subresultant matrices since the degree of the AGCD was not correctly determined.



(i) The singular values $\{\sigma_{k,i}\}$ of the unprocessed subresultant matrices $\{S_k(f(x,y), g(x,y))\}$

(ii) The singular values $\{\sigma_{k,i}\}$ of the preprocessed subresultant matrices $\{S_k(\lambda_k \tilde{f}_k(\omega_1,\omega_2), \tilde{g}_k(\omega_1,\omega_2))\}$

Figure 6.7: The singular values $\{\sigma_{k,i}\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 6.6.2

	Without Preprocessing $u_t(x,y)$, $v_t(x,y)$ and $d_t(x,y)$	With Preprocessing $\tilde{u}_t(x,y)$, $\tilde{v}_t(x,y)$ and $\tilde{d}_t(x,y)$
Error $\hat{u}_t(x,y)$	-	$9.005377e-10$
Error $\hat{v}_t(x,y)$	-	$7.410582e-10$
Error $\hat{d}_t(x,y)$	-	$3.090752e-10$
Average	-	$5.908439e-10$

Table 6.4: Error in the approximations of $\hat{u}_t(x,y)$, $\hat{v}_t(x,y)$ and $\hat{d}_t(x,y)$ in Example 6.6.2

The amount of noise is reduced such that $\epsilon_{f,i_1,i_2} = \epsilon_{g,j_1,j_2} = 10^{-14}$, and the degree of the AGCD is correctly determined by both the sets of unprocessed subresultant matrices and preprocessed subresultant matrices. Given the tth unprocessed and preprocessed subresultant matrices, the approximations of $\hat{u}_t(x,y)$, $\hat{g}_t(x,y)$ and $\hat{d}_t(x,y)$ are computed and the respective errors are given in Table 6.5. It can be seen that approximations obtained from the tth preprocessed subresultant matrix are significantly better than the approximations obtained from the tth unprocessed subresultant matrix.

	Without Preprocessing $u_t(x,y)$, $v_t(x,y)$ and $d_t(x,y)$	With Preprocessing $\tilde{u}_t(x,y)$, $\tilde{v}_t(x,y)$ and $\tilde{d}_t(x,y)$
Error $\hat{u}_t(x,y)$	$5.582028e-09$	$9.895789e-14$
Error $\hat{v}_t(x,y)$	$4.480417e-09$	$8.115812e-14$
Error $\hat{d}_t(x,y)$	$4.293959e-09$	$3.681717e-14$
Average	$4.753237e-09$	$6.662097e-14$

Table 6.5: Error in the approximations of $\hat{u}_t(x,y)$, $\hat{v}_t(x,y)$ and $\hat{d}_t(x,y)$ with reduced upper bound of noise in Example 6.6.2

□

Three-Polynomial Problems

Example 6.6.3. In this example, the polynomials from Example 6.2.1 are reconsidered. The SVD of the sets of preprocessed subresultant matrices

(i) $\{\tilde{S}_k(\lambda_k\tilde{f}_k(\omega_1,\omega_2),\tilde{g}_k(\omega_1,\omega_2),\rho_k\tilde{h}_k(\omega_1,\omega_2))\}$

(ii) $\{\hat{S}_k(\lambda_k\tilde{f}_k(\omega_1,\omega_2),\tilde{g}_k(\omega_1,\omega_2),\rho_k\tilde{h}_k(\omega_1,\omega_2))\}$

(iii) $\{\hat{S}_k(\mu_k\tilde{g}_k(\omega_1,\omega_2),\tilde{f}_k(\omega_1,\omega_2),\rho_k\tilde{h}_k(\omega_1,\omega_2))\}$

(iv) $\{\hat{S}_k(\rho_k\tilde{h}_k(\omega_1,\omega_2),\tilde{g}_k(\omega_1,\omega_2),\lambda_k\tilde{f}_k(\omega_1,\omega_2))\}$

are computed and plotted in Figure 6.8. There is a clear separation between the non-zero and numerically zero singular values for all four sets of subresultant matrices. This is in contrast to the singular values of the unprocessed subresultant matrices in Example 6.2.1 (Figure 6.2). From this example it can be seen that the ordering of the polynomials, which gives rise to the three variations of the (2×3) subresultant matrices, is irrelevant when polynomials are first preprocessed.

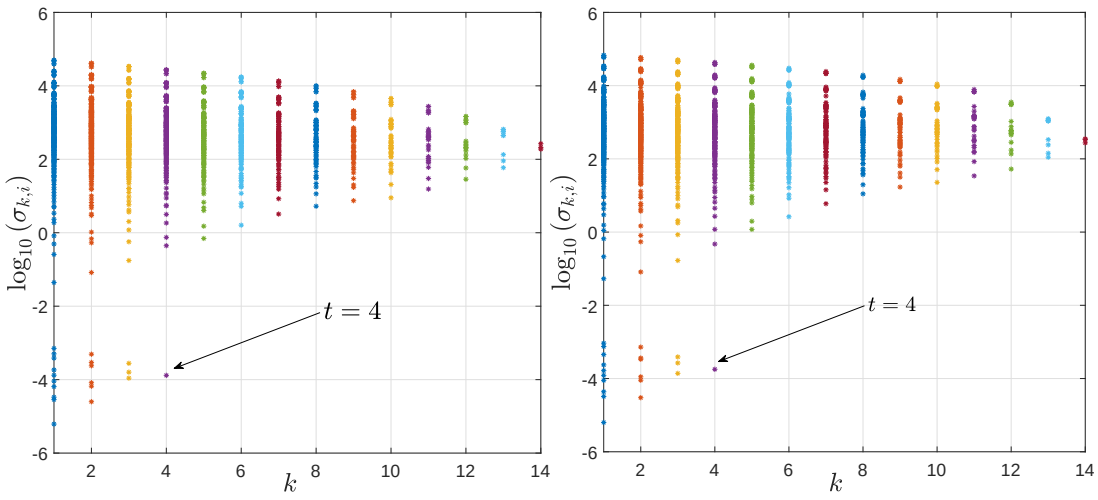
□

Example 6.6.4. Consider the Bernstein form of the exact polynomials $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ of degrees $m = 23$, $n = 11$ and $o = 10$, whose factorisations are given by

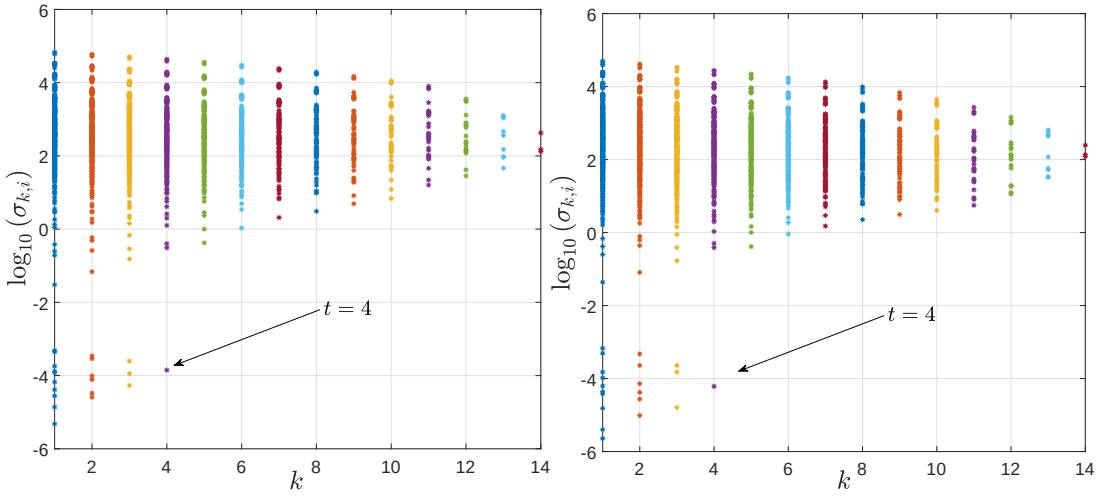
$$\begin{aligned}
\hat{f}(x,y) &= (x+2.21657951321)(x^2+y^2+0.5679814324687)^2\times \\
&\quad (x+y+42.46578784351654)^3(x+y+0.0124)^6(x-0.554687987932164654)^3 \\
\hat{g}(x,y) &= (x+2.21657951321)(x^2+y^2+0.5679814324687)^2\times \\
&\quad (x+y+42.46578784351654)^3(x+y+0.4512)^3 \\
\hat{h}(x,y) &= (x+2.21657951321)(x^2+y^2+0.5679814324687)^2\times \\
&\quad (x+y+42.46578784351654)^3(12x^2+y^2-52.34)
\end{aligned}$$

and whose GCD $\hat{d}_t(x,y)$ of degree $t = 8$ in factorised form is given by

$$\begin{aligned}
\hat{d}_t(x,y) &= (x+2.21657951321)(x^2+y^2+0.5679814324687)^2\times \\
&\quad (x+y+42.46578784351654)^3.
\end{aligned}$$



(i) $\tilde{S}(\lambda_k \tilde{f}_k(\omega_1, \omega_2), \tilde{g}_k(\omega_1, \omega_2), \rho_k \tilde{h}_k(\omega_1, \omega_2))$ **(ii)** $\hat{S}_k(\lambda_k \tilde{f}_k(\omega_1, \omega_2), \tilde{g}_k(\omega_1, \omega_2), \rho_k \tilde{h}_k(\omega_1, \omega_2))$

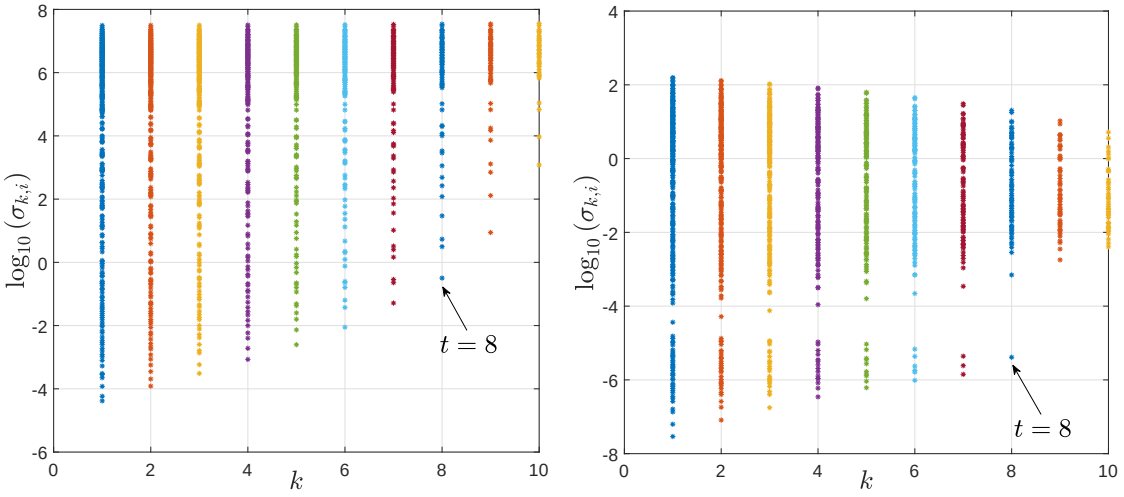


(iii) $\hat{S}_k(\mu_k \tilde{g}_k(\omega_1, \omega_2), \tilde{f}_k(\omega_1, \omega_2), \rho_k \tilde{h}_k(\omega_1, \omega_2))$ **(iv)** $\hat{S}_k(\rho_k \tilde{h}_k(\omega_1, \omega_2), \tilde{g}_k(\omega_1, \omega_2), \lambda_k \tilde{f}_k(\omega_1, \omega_2))$

Figure 6.8: The singular values $\{\sigma_{k,i}\}$ of the preprocessed subresultant matrices in Example 6.6.3

Noise is added to the coefficients of the exact polynomials $\hat{f}(x, y)$, $\hat{g}(x, y)$ and $\hat{h}(x, y)$, where the coefficients of the inexact polynomials are given by (6.20) with $\{r_{f,i_1,i_2}\}$, $\{r_{g,j_1,j_2}\}$ and $\{r_{h,p_1,p_2}\}$ being uniformly distributed random variables in the interval $[-1, 1]$ and $\{\epsilon_{f,i_1,i_2}\}$, $\{\epsilon_{g,i_1,i_2}\}$ and $\{\epsilon_{h,i_1,i_2}\}$ being uniformly distributed random variables in the interval $\left[10^{-6}, 10^{-4}\right]$.

The sets of singular values of the unprocessed subresultant matrices $\{S_k(f(x, y), g(x, y), h(x, y)) \mid k = 1, \ldots, \min(m, n, o)\}$ are plotted in Figure 6.9i, but the degree of the AGCD cannot be computed from these values as there is no separation into numerically zero and non-zero singular values. However, the degree of the AGCD can be computed from the singular values of the preprocessed subresultant matrices $\{S_k(\lambda_k \tilde{f}_k(\omega_1, \omega_2), \tilde{g}_k(\omega_1, \omega_2), \rho_k \tilde{h}_k(\omega_1, \omega_2))\}$, which are plotted in Figure 6.9ii, and the degree of the AGCD is correctly determined to be $t = 8$.



(i) The singular values $\{\sigma_{k,i}\}$ of the unprocessed subresultant matrices $\{\hat{S}_k(f(x,y), g(x,y), h(x,y))\}$

(ii) The singular values $\{\sigma_{k,i}\}$ of the preprocessed subresultant matrices $\{\hat{S}_k(\alpha\tilde{f}(\omega_1,\omega_2), \beta\tilde{g}(\omega_1,\omega_2), \gamma\tilde{h}(\omega_1,\omega_2))\}$

Figure 6.9: The singular values $\{\sigma_{k,i}\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 6.6.4

Approximations of the cofactor polynomials and coefficients of the GCD are computed where possible. The SVD of the subresultant matrices of the inexact unprocessed polynomials $f(x,y)$, $g(x,y)$ and $h(x,y)$ cannot be used to determine the degree of the AGCD and so the coefficients of the cofactor polynomials and GCD cannot be approximated. Errors in the approximations of $\hat{u}_t(x,y)$, $\hat{v}_t(x,y)$, $\hat{w}_t(x,y)$ and $\hat{d}_t(x,y)$ are given in Table 6.6.

Method	Without Preprocessing $u_t(x,y)$, $v_t(x,y)$, $w_t(x,y)$ and $d_t(x,y)$	With Preprocessing $\tilde{u}_t(x,y)$, $\tilde{v}_t(x,y)$, $\tilde{w}_t(x,y)$ and $\tilde{d}_t(x,y)$
Error $\hat{u}_t(x,y)$	-	$5.890669e-04$
Error $\hat{v}_t(x,y)$	-	$6.053376e-04$
Error $\hat{w}_t(x,y)$	-	$4.344286e-04$
Error $\hat{d}_t(x,y)$	-	$5.718709e-04$
Average	-	$5.5018e-04$

Table 6.6: Error in the approximations of $\hat{u}_t(x,y)$, $\hat{v}_t(x,y)$, $\hat{w}_t(x,y)$ and $\hat{d}_t(x,y)$, where $\{\epsilon_{f,i}\}$ and $\{\epsilon_{g,j}\}$ are in the interval $[10^{-6}, 10^{-4}]$ in Example 6.6.4

The noise level is reduced such that ϵ_{f,i_1,i_2}, ϵ_{g,j_1,j_2} and ϵ_{h,k_1,k_2} are uniformly distributed random variables in the interval $[10^{-8}, 10^{-6}]$, at which point the degree of the AGCD is correctly determined from the SVD of both the unprocessed and preprocessed subresultant matrices. The errors in the approximations of the coefficients of the cofactor polynomials and the GCD are given in Table 6.7

Method	Without Preprocessing $u_t(x,y)$, $v_t(x,y)$, $w_t(x,y)$ and $d_t(x,y)$	With Preprocessing $\tilde{u}_t(x,y)$, $\tilde{v}_t(x,y)$, $\tilde{w}_t(x,y)$ and $\tilde{d}_t(x,y)$
Error $\hat{u}_t(x,y)$	$1.682191e-04$	$5.375237e-06$
Error $\hat{v}_t(x,y)$	$1.477191e-04$	$5.579333e-06$
Error $\hat{w}_t(x,y)$	$1.268452e-04$	$3.991345e-06$
Error $\hat{d}_t(x,y)$	$1.582510e-04$	$5.214778e-06$
Average	$1.494456e-04$	$4.998432e-06$

Table 6.7: Error in the approximations of $\hat{u}_t(x,y)$, $\hat{v}_t(x,y)$, $\hat{w}_t(x,y)$ and $\hat{d}_t(x,y)$, where $\{\epsilon_{f,i}\}$ and $\{\epsilon_{g,j}\}$ are in the interval $[10^{-8}, 10^{-6}]$ in Example 6.6.4

□

6.7 Conclusion

This chapter has considered the computation of the degree and coefficients of the GCD of two or three bivariate polynomials in Bernstein form. The main findings of this chapter are outlined below:

The Bivariate Subresultant Matrix : This chapter has extended the definition of the Sylvester matrix and the set of subresultant matrices. These were defined for the two-polynomial and three-polynomial GCD finding problems where the bivariate polynomials are in Bernstein form defined over a triangular domain.

Variations of the Subresultant Matrices : Several variants of the (2×3) and (3×3) subresultant matrices were defined. It was shown that the computation of the degree of the GCD of poorly scaled polynomials can return erroneous results due to unbalanced row-partitions in these subresultant matrices. In particular, when two of the three polynomials, $\hat{f}(x,y)$ and $\hat{g}(x,y)$, have a GCD $\hat{d}_a(x,y)$ of degree greater than the degree of the GCD of all three polynomials, poor scaling can result in the GCD of $\hat{f}(x,y)$ and $\hat{g}(x,y)$ being incorrectly identified as the GCD of all three polynomials.

Preprocessing : The poor scaling was addressed by adapting the preprocessing operations described in Section 3.4. Examples have shown that improved results are obtained from preprocessed polynomials. Example 6.6.1 and Example 6.6.2 have shown that the degree of the GCD can be computed by analysis of the singular values of the set of preprocessed subresultant matrices, where analysis of unprocessed subresultant matrices would otherwise fail.

Coefficient Computation : It has also been shown that the approximations of both the cofactor polynomials $\hat{u}_t(x,y)$, $\hat{v}_t(x,y)$ and $\hat{w}_t(x,y)$ and the coefficients of the GCD $\hat{d}_t(x,y)$ obtained from the tth preprocessed subresultant matrix are considerably better than the approximations obtained by the same method applied to the tth unprocessed subresultant matrix.

Having extended the GCD finding method to solve the two-polynomial and three-polynomial problem for bivariate polynomials defined over a triangular domain, the more difficult extension to the two-polynomial and three-polynomial problem for polynomials defined over a rectangular domain is described in the next chapter.

Chapter 7

GCDs of Bivariate Polynomials over a Rectangular Domain

The previous chapter considered the computation of the degree and coefficients of the GCD (or AGCD) of two or three bivariate polynomials in Bernstein form defined over a triangular domain. This chapter considers the computation of the GCD of two or three bivariate polynomials defined over a rectangular domain, which is a problem that arises in the computation of intersection points involving tensor-product Bézier surfaces.

The bivariate polynomial defined over a rectangular domain is given in terms of its relative degree with respect to x and y, and the computation of the degree (t_1, t_2) of the two-polynomial or three-polynomial GCD reduces to computing the numerical rank of a two-dimensional array of two-polynomial or three-polynomial subresultant matrices.

The first of two methods developed in this chapter, described as the BVGCD method, is a simple extension of the UGCD method, but has considerable computational complexity. A one-dimensional search for the degree t in an interval $[1, \min(m, n)]$ is replaced by a two-dimensional search for (t_1, t_2) in an array of (k_1, k_2) pairs for $k_1 = 0, \ldots, \min(m_1, n_1)$ and $k_2 = 0, \ldots, \min(m_2, n_2)$.

Suppose that $m_1 \approx m_2 \approx n_1 \approx n_2$, then there are approximately m^2 subresultant matrices which must be evaluated. Each matrix in the m^2 array must be preprocessed and its SVD must also be computed. The computation of the degree of the GCD is the most expensive component of this method.

The second method presented in this chapter, BVDRGCD, overcomes the cost associated with the computation of the degree of the GCD. The developed method is faster, has similar intermediate results, and the same outputs as the BVGCD method.

Section 7.1 This section describes the first approach to the method of computing the degree of the GCD of two or three bivariate polynomials in Bernstein form defined over a rectangular domain. It will be shown that the problem reduces to the determination of the index (k_1, k_2) of the last numerically rank deficient subresultant matrix. The two-polynomial and three-polynomial subresultant matrices are defined and their respective structures are discussed.

Section 7.2 Variants of the two-polynomial and three-polynomial subresultant matrices are considered for bivariate polynomials defined over a rectangular domain. The

binomial terms included in the entries of some of these variants are shown to cause poor scaling amongst the matrix partitions.

Section 7.3 Preprocessing of two-polynomial and three-polynomial subresultant matrices has been considered for univariate and bivariate polynomials over a triangular domain. This section extends the set of preprocessing operations defined in earlier chapters so that the subresultant matrices of two or three bivariate polynomials defined over a rectangular domain can be similarly preprocessed.

Section 7.4 This section develops two methods for the computation of the degree of the GCD of two or three bivariate polynomials in Bernstein form. The first method, BVGCD, is a simple but computationally expensive extension of univariate GCD (UGCD). The second method, bivariate dimension reducing GCD (BVDRGCD), aims to reduce the problem to a one-dimensional set of subresultant matrices by a sequence of degree elevations such that the degree of the GCD is computed in two stages but using a significantly faster algorithm.

Section 7.5 Approximations of the coefficients of the cofactor polynomials and the GCD are computed by a least squares based method for the two-polynomial and three-polynomial problem. Examples will show that the best approximations are given when polynomials are first preprocessed.

Section 7.6 This section considers a set of examples of the computation of the degree and coefficients of the GCD of two bivariate polynomials. Both the methods of BVGCD and BVDRGCD will be considered in these examples.

7.1 The GCD of Two or Three Bivariate Polynomials in Bernstein Form Defined over a Rectangular Domain

Consider the two polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ of degrees (m_1,m_2) and (n_1,n_2) respectively, which are given by

$$\hat{f}(x,y)=\sum_{i_2=0}^{m_2}\sum_{i_1=0}^{m_1}\hat{a}_{i_1,i_2}B_{i_1}^{m_1}(x)B_{i_2}^{m_2}(y) \qquad \text{and} \qquad \hat{g}(x,y)=\sum_{i_2=0}^{n_2}\sum_{i_1=0}^{n_1}\hat{b}_{i_1,i_2}B_{i_1}^{n_1}(x)B_{i_2}^{n_2}(y).$$

Suppose they have a GCD $\hat{d}_{t_1,t_2}(x,y)$ of degree (t_1,t_2) which is given by

$$\hat{d}_{t_1,t_2}(x,y)=\sum_{i_1=0}^{t_1}\sum_{i_2=0}^{t_2}\hat{d}_{i_1,i_2}B_{i_1}^{t_1}(x)B_{i_2}^{t_2}(y),$$

then there exists a set of cofactor polynomials $\hat{u}_{k_1,k_2}(x,y)$ and $\hat{v}_{k_1,k_2}(x,y)$ such that

$$\hat{f}(x,y)\hat{v}_{k_1,k_2}(x,y)-\hat{g}(x,y)\hat{u}_{k_1,k_2}(x,y)=0 \tag{7.1}$$

holds for $k_1 = 0, \ldots, t_1;\ k_2 = 0, \ldots, t_2$. The cofactor polynomials $\hat{u}_{k_1,k_2}(x,y)$ and $\hat{v}_{k_1,k_2}(x,y)$ are given by

$$\hat{v}_{k_1,k_2}(x,y) = \sum_{i_2=0}^{n_2-k_2} \sum_{i_1=0}^{n_1-k_1} \hat{v}_{i,j} B_{i_1}^{n_1-k_1}(x) B_{i_2}^{n_2-k_2}(y),$$
$$\hat{u}_{k_1,k_2}(x,y) = \sum_{i_2=0}^{m_2-k_2} \sum_{i_1=0}^{m_1-k_1} \hat{u}_{i_1,i_2} B_{i_1}^{m_1-k_1}(x) B_{i_2}^{m_2-k_2}(y).$$

Equation (7.1) can be written in matrix form as

$$S_{k_1,k_2}\left(\hat{f}(x,y), \hat{g}(x,y)\right) \mathbf{x}_{k_1,k_2} = \mathbf{0}, \tag{7.2}$$

which has non-trivial solutions for $k_1 = 0, \ldots, t_1;\ k_2 = 0, \ldots, t_2$. The solution vector $\mathbf{x}_{k_1,k_2}$ contains the coefficients of the polynomials $\hat{u}_{k_1,k_2}(x,y)$ and $\hat{v}_{k_1,k_2}(x,y)$ and is given by

$$\mathbf{x}_{k_1,k_2} = \begin{bmatrix} \hat{\mathbf{v}}_{k_1,k_2}, & -\hat{\mathbf{u}}_{k_1,k_2} \end{bmatrix}^T,$$

where the partitioned vectors $\hat{\mathbf{v}}_{k_1,k_2}$ and $\hat{\mathbf{u}}_{k_1,k_2}$ are given by

$$\hat{\mathbf{v}}_{k_1,k_2} = \begin{bmatrix} \hat{\mathbf{v}}_1, & \hat{\mathbf{v}}_2, & \ldots, & \hat{\mathbf{v}}_{n_2-k_2} \end{bmatrix}^T \quad \text{and} \quad \hat{\mathbf{u}}_{k_1,k_2} = \begin{bmatrix} \hat{\mathbf{u}}_1, & \hat{\mathbf{u}}_2, & \ldots, & \hat{\mathbf{u}}_{m_2-k_2} \end{bmatrix}^T,$$

the partitions of which, $\{\hat{\mathbf{v}}_i\}$ and $\{\hat{\mathbf{u}}_i\}$, are given by

$$\hat{\mathbf{v}}_i = \begin{bmatrix} \hat{v}_{0,i}, & \hat{v}_{1,i}, & \ldots, & \hat{v}_{m_1-k_1,i} \end{bmatrix}^T \quad \text{and} \quad \hat{\mathbf{u}}_i = \begin{bmatrix} \hat{u}_{0,i} & \hat{u}_{1,i} & \ldots, & \hat{u}_{n_1-k_1,i} \end{bmatrix}^T.$$

The matrix $S_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y))$ is the (k_1,k_2)th two-polynomial subresultant matrix, where the number of rows r and columns c are given by

$$r = (m_1 + n_1 - k_1 + 1)(m_2 + n_2 - k_2 + 1)$$
$$c = (m_1 - k_1 + 1)(m_2 - k_2 + 1) + (n_1 - k_1 + 1)(n_2 - k_2 + 1).$$

In a similar manner to other versions of the subresultant matrices, the (k_1,k_2)th subresultant matrix $S_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y))$ can be defined as

$$S_{k_1,k_2} = \left[\ C_{n_1-k_1,n_2-k_2}\left(\hat{f}(x,y)\right) \ \middle|\ C_{m_1-k_1,m_2-k_2}\left(\hat{g}(x,y)\right)\ \right],$$

where the first partition $C_{n_1-k_1,n_2-k_2}(\hat{f}(x,y))$ is a bivariate convolution matrix of the same

structure as the matrix defined in (2.19). The matrix $C_{n_1-k_1,n_2-k_2}(\hat{f}(x,y))$ is given by

$$\begin{bmatrix} \frac{C_{n_1-k_1}\left(\hat{f}_0(x)\right)\binom{m_2}{0}\binom{n_2-k_2}{0}}{\binom{m_2+n_2}{0}} & & & \\ \frac{C_{n_1-k_1}\left(\hat{f}_1(x)\right)\binom{m_2}{1}\binom{n_2-k_2}{0}}{\binom{m_2+n_2-k_2}{1}} & \frac{C_{n_1-k_1}\left(\hat{f}_0(x)\right)\binom{m_2}{0}\binom{n_2-k_2}{1}}{\binom{m_2+n_2-k_2}{1}} & & \\ \vdots & \frac{C_{n_1-k_1}\left(\hat{f}_1(x)\right)\binom{m_2}{1}\binom{n_2-k_2}{1}}{\binom{m_2+n_2-k_2}{3}} & \ddots & \\ \vdots & \vdots & \ddots & \frac{C_{n_1-k_1}\left(\hat{f}_0(x)\right)\binom{m_2}{0}\binom{n_2-k_2}{n_2-k_2}}{\binom{m_2+n_2-k_2}{n_2-k_2}} \\ \frac{C_{n_1-k_1}\left(\hat{f}_{m_2}(x)\right)\binom{m_2}{m_2}\binom{n_2-k_2}{0}}{\binom{m_2+n_2-k_2}{m_2}} & \vdots & & \frac{C_{n_1-k_1}\left(\hat{f}_1(x)\right)\binom{m_2}{1}\binom{n_2-k_2}{n_2-k_2}}{\binom{m_2+n_2-k_2}{n_2-k_2+1}} \\ & \frac{C_{n_1-k_1}\left(\hat{f}_{m_2}(x)\right)\binom{m_2}{m_2}\binom{n_2-k_2}{1}}{\binom{m_2+n_2-k_2}{m_2+1}} & & \vdots \\ & & \ddots & \vdots \\ & & & \frac{C_{n_1-k_1}\left(\hat{f}_{m_2}(x)\right)\binom{m_2}{m_2}\binom{n_2-k_2}{n_2-k_2}}{\binom{m_2+n_2-k_2}{m_2+n_2-k_2}} \end{bmatrix}.$$

Alternatively, the (k_1,k_2)th subresultant matrix can be written as

$$S_{k_1,k_2}\left(\hat{f}(x,y),\hat{g}(x,y)\right) = D^{-1}_{m_1+n_1-k_1,m_2+n_2-k_2} T_{k_1,k_2}\left(\hat{f}(x,y),\hat{g}(x,y)\right)\hat{Q}_{k_1,k_2}.$$

The block diagonal matrix $D^{-1}_{m_1+n_1-k_1,m_2+n_2-k_2}$ of order $(m_1+n_1-k_1+1)(m_2+n_2-k_2+1)$ is given by

$$\operatorname{diag}\left[\; D^{-1}_{m_1+n_1-k_1}\tfrac{1}{\binom{m_2+n_2-k_2}{0}},\quad D^{-1}_{m_1+n_1-k_1}\tfrac{1}{\binom{m_2+n_2-k_2}{1}},\quad \ldots,\quad D^{-1}_{m_1+n_1-k_1}\tfrac{1}{\binom{m_2+n_2-k_2}{m_2+n_2-k_2}}\;\right], \tag{7.3}$$

where the diagonal matrix $D^{-1}_{m_1+n_1-k_1}$ of order $(m_1+n_1-k_1+1)$ is defined in (3.6). The matrix $T_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y))$ is given by

$$\left[\; T_{n_1-k_1,n_2-k_2}\left(\hat{f}(x,y)\right) \;\middle|\; T_{m_1-k_1,m_2-k_2}\left(\hat{g}(x,y)\right)\;\right],$$

where the partitions are of the form (2.20). The block diagonal matrix $\hat{Q}_{k_1,k_2}$ is given by

$$\hat{Q}_{k_1,k_2} = \operatorname{diag}\left[\; Q_{n_1-k_1,n_2-k_2},\quad Q_{m_1-k_1,m_2-k_2}\;\right],$$

where both of the matrices $Q_{n_1-k_1,n_2-k_2}$ and $Q_{m_1-k_1,m_2-k_2}$ are themselves block diagonal matrices given by

$$\begin{aligned} Q_{n_1-k_1,n_2-k_2} &= \operatorname{diag}\left[\; Q_{n_1-k_1}\tbinom{n_2-k_2}{0},\quad Q_{n_1-k_1}\tbinom{n_2-k_2}{1},\quad \ldots,\quad Q_{n_1-k_1}\tbinom{n_2-k_2}{n_2-k_2}\;\right] \\ Q_{m_1-k_1,m_2-k_2} &= \operatorname{diag}\left[\; Q_{m_1-k_1}\tbinom{m_2-k_2}{0},\quad Q_{m_1-k_1}\tbinom{m_2-k_2}{1},\quad \ldots,\quad Q_{m_1-k_1}\tbinom{m_2-k_2}{m_2-k_2}\;\right] \end{aligned}$$

and $Q_{n_1-k_1} \in \mathbb{R}^{(n_1-k_1+1)\times(n_1-k_1+1)}$ and $Q_{m_1-k_1} \in \mathbb{R}^{(m_1-k_1+1)\times(m_1-k_1+1)}$ are of the same structure as the matrix partitions shown in (2.12).

From equation (7.2) the (k_1,k_2)th subresultant matrix of two bivariate polynomials in Bernstein form is numerically rank deficient for $k_1 = 0,\ldots,t_1$ and $k_2 = 0,\ldots,t_2$, while

the remaining subresultant matrices are of full rank

$$\begin{aligned}\operatorname{rank}(S_{k_1,k_2}) &< (m_1+n_1-k_1+1)(m_2+n_2-k_2+1)\\ &\quad \text{for} \quad k_1=0,\ldots,t_1;\ k_2=0,\ldots,t_2\\ \operatorname{rank}(S_{k_1,k_2}) &= (m_1+n_1-k_1+1)(m_2+n_2-k_2+1)\\ &\quad \text{for} \quad k_1=(t_1+1),\ldots,\min(m_1,n_1);\ k_2=(t_2+1),\ldots,\min(m_2,n_2).\end{aligned}$$

The computation of the degree of the GCD therefore reduces to the computation of the last numerically rank deficient subresultant matrix $S_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y))$, where $S_{k_1+1,k_2}(\hat{f}(x,y),\hat{g}(x,y))$ and $S_{k_1,k_2+1}(\hat{f}(x,y),\hat{g}(x,y))$ are both of full rank. Having reduced the two-polynomial problem to the determination of the last numerically rank deficient subresultant matrix, the three-polynomial problem is considered.

7.1.1 The GCD of Three Bivariate Polynomials in Bernstein Form over a Rectangular Domain

Suppose $\hat{f}(x,y)$, $\hat{g}(x,y)$ and a third polynomial $\hat{h}(x,y)$ have a GCD of degree (t_1,t_2), then there exist common divisors of degree (k_1,k_2) such that

$$\frac{\hat{f}(x,y)}{\hat{u}_{k_1,k_2}(x,y)}=\hat{d}_{k_1,k_2}(x,y),\quad \frac{\hat{g}(x,y)}{\hat{v}_{k_1,k_2}(x,y)}=\hat{d}_{k_1,k_2}(x,y)\quad\text{and}\quad \frac{\hat{h}(x,y)}{\hat{w}_{k_1,k_2(x,y)}}=\hat{d}_{k_1,k_2}(x,y)$$

for $k_1=0,\ldots,t_1$ and $k_2=0,\ldots,t_2$. This gives rise to three equations, the first of which is defined in (7.1) and the remaining two equations are given by

$$\hat{f}(x,y)\hat{w}_{k_1,k_2}(x,y)-\hat{h}(x,y)\hat{u}_{k_1,k_2}(x,y)=0 \tag{7.4}$$

$$\hat{h}(x,y)\hat{v}_{k_1,k_2}(x,y)-\hat{g}(x,y)\hat{w}_{k_1,k_2}(x,y)=0. \tag{7.5}$$

The three equations (7.1, 7.4, 7.5) must be simultaneously satisfied and can be written in matrix form as

$$\tilde{S}_{k_1,k_2}\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)\mathbf{x}_{k_1,k_2}=\mathbf{0}, \tag{7.6}$$

which has a non-trivial solution for $k_1=0,\ldots,t_1$; and $k_2=0,\ldots,t_2$ and the solution vector is given by

$$\mathbf{x}_{k_1,k_2}=\left[\begin{array}{ccc}\hat{\mathbf{v}}_{k_1,k_2}, & \hat{\mathbf{w}}_{k_1,k_2}, & -\hat{\mathbf{u}}_{k_1,k_2}\end{array}\right]^T, \tag{7.7}$$

where $\hat{\mathbf{v}}_{k_1,k_2}$, $\hat{\mathbf{w}}_{k_1,k_2}$ and $\hat{\mathbf{u}}_{k_1,k_2}$ are the vectors of the coefficients of polynomials $\hat{u}_{k_1,k_2}(x,y)$, $\hat{v}_{k_1,k_2}(x,y)$ and $\hat{w}_{k_1,k_2}(x,y)$.

The matrix $\tilde{S}_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ in (7.6) is the (k_1,k_2)th (3×3) partitioned subresultant matrix of three bivariate polynomials. It can also be written as

$$\tilde{S}_{k_1,k_2}\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)=\tilde{D}^{-1}_{k_1,k_2}\tilde{T}_{k_1,k_2}\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)\tilde{Q}_k. \tag{7.8}$$

The block diagonal matrix $\tilde{D}^{-1}_{k_1,k_2}$ is given by

$$\tilde{D}^{-1}_{k_1,k_2} = \text{diag}\left[\ D^{-1}_{m_1+n_1-k_1,m_2+n_2-k_2},\quad D^{-1}_{m_1+o_1-k_1,m_2+o_2-k_2},\quad D^{-1}_{n_1+o_1-k_1,n_2+o_2-k_2}\ \right],$$

where the matrix partitions are of the same form as the matrix $D^{-1}_{m_1+n_1-k_1,m_2+n_2-k_2}$ in (7.3). The partitioned matrix $\tilde{T}_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ is given by

$$\begin{bmatrix} T_{n_1-k_1,n_2-k_2}\Big(\hat{f}(x,y)\Big) & & T_{m_1-k_1,m_2-k_2}\Big(\hat{g}(x,y)\Big) \\ & T_{o_1-k_1,o_2-k_2}\Big(\hat{f}(x,y)\Big) & T_{m_1-k_1,m_2-k_2}\Big(\hat{h}(x,y)\Big) \\ T_{n_1-k_1,n_2-k_2}\Big(\hat{h}(x,y)\Big) & -T_{o_1-k_1,o_2-k_2}\Big(\hat{g}(x,y)\Big) & \end{bmatrix},$$

where the partitions $T_{n_1-k_1,n_2-k_2}(\hat{f}(x,y))$ are of the same form as (2.20). The block diagonal matrix $\tilde{Q}_{k_1,k_2}$ is given by

$$\tilde{Q}_{k_1,k_2} = \text{diag}\left[\ Q_{n_1-k_1,n_2-k_2},\quad Q_{o_1-k_1,o_2-k_2},\quad Q_{m_1-k_1,m_2-k_2}\ \right]. \tag{7.9}$$

The matrix $\tilde{S}_{k_1,k_2}\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)$ can be defined in terms of row-partitions

$$\begin{bmatrix} \mathcal{R}_{a,k_1,k_2} \\ \mathcal{R}_{b,k_1,k_2} \\ \mathcal{R}_{c,k_1,k_2} \end{bmatrix} = \begin{bmatrix} C_{n_1-k_1,n_2-k_2}\Big(\hat{f}(x,y)\Big) & & C_{m_1-k_1,m_2-k_2}\Big(\hat{g}(x,y)\Big) \\ & C_{o_1-k_1,o_2-k_2}\Big(\hat{f}(x,y)\Big) & C_{m_1-k_1,m_2-k_2}\Big(\hat{h}(x,y)\Big) \\ C_{n_1-k_1,n_2-k_2}\Big(\hat{h}(x,y)\Big) & -C_{o_1-k_1,o_2-k_2}\Big(\hat{g}(x,y)\Big) & \end{bmatrix},$$

where

$$\mathcal{R}_{a,k_1,k_2} = \left[\ C_{n_1-k_1,n_2-k_2}\Big(\hat{f}(x,y)\Big)\quad 0_a\quad C_{m_1-k_1,m_2-k_2}\Big(\hat{g}(x,y)\Big)\ \right], \tag{7.10}$$

$$\mathcal{R}_{b,k_1,k_2} = \left[\ 0_b\quad C_{o_1-k_1,o_2-k_2}\Big(\hat{f}(x,y)\Big)\quad C_{m_1-k_1,m_2-k_2}\Big(\hat{h}(x,y)\Big)\ \right], \tag{7.11}$$

$$\mathcal{R}_{c,k_1,k_2} = \left[\ C_{n_1-k_1,n_2-k_2}\Big(\hat{h}(x,y)\Big)\quad -C_{o_1-k_1,o_2-k_2}\Big(\hat{g}(x,y)\Big)\quad 0_c\ \right]. \tag{7.12}$$

The matrices 0_a, 0_b and 0_c in (7.10, 7.11, 7.12) are appropriately sized zero matrices, where 0_a is a zero matrix of size $(m_1+n_1-k_1)(m_2+n_2-k_2)\times(o_1-k_1+1)(o_2-k_2+1)$, the matrix 0_b is of size $(m_1+o_1-k_1)(m_2+o_2-k_2)\times(m_1-k_1+1)(m_2-k_2+1)$ and the matrix 0_c is of size $(n_1+o_1-k_1)(n_2+o_2-k_2)\times(m_1-k_1+1)(m_2-k_2+1)$.

An alternative definition of the three-polynomial subresultant matrix has a (2×3) partitioned structure. Two of the three equations (7.1, 7.4, 7.5) are sufficient to describe the system, which gives rise to the (2×3) partitioned subresultant matrices whose row-partitions are a subset of the row-partitions of the (3×3) partitioned subresultant matrix.

The subresultant matrices of this form have already been described for use in the three-polynomial GCD problem where polynomials are either univariate or bivariate and defined over a triangular domain. In this section only one variation of the (2×3) subresultant matrix is considered and the other two variations are easily derived.

The equations (7.1) and (7.4) can be written in matrix form as

$$\hat{S}_{k_1,k_2}\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)\mathbf{x}_{k_1,k_2} = \mathbf{0},$$

which has non-trivial solutions for $k_1 = 0, \ldots, t_1$; and $k_2 = 0, \ldots, t_2$, and the vector $\mathbf{x}_{k_1,k_2}$ is given by (7.7).

The matrix $\hat{S}_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))$ is the (2×3) partitioned subresultant matrix and is given by

$$\begin{bmatrix} \mathcal{R}_{a,k_1,k_2} \\ \mathcal{R}_{b,k_1,k_2} \end{bmatrix} = \begin{bmatrix} C_{n_1-k_1,n_2-k_2}\left(\hat{f}(x,y)\right) & & C_{m_1-k_1,m_2-k_2}\left(\hat{g}(x,y)\right) \\ & C_{o_1-k_1,o_2-k_2}\left(\hat{f}(x,y)\right) & C_{m_1-k_1,m_2-k_2}\left(\hat{h}(x,y)\right) \end{bmatrix},$$

where $\mathcal{R}_{a,k_1,k_2}$ and $\mathcal{R}_{b,k_1,k_2}$ are row-partitions as defined in (7.10) and (7.11).

7.2 Variants of the Subresultant Matrices

The Two-Polynomial Subresultant Matrices

As with the other forms of subresultant matrix of two polynomials in Bernstein form, the sequence of subresultant matrices of two bivariate polynomials have several variants given by

(i) $\{T_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y))\}$

(ii) $\{D^{-1}_{k_1,k_2} T_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y))\}$

(iii) $\{T_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y))\hat{Q}_{k_1,k_2}\}$

(iv) $\{D^{-1}_{k_1,k_2} T_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y))\hat{Q}_{k_1,k_2}\}$

for $k_1 = 0, \ldots, \min(m_1, n_1)$; $k_2 = 0, \ldots, \min(m_2, n_2)$.

1. The first variant of the (k_1, k_2)th subresultant matrix is given by $T_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y))$, and entries in the first partition are of the form

$$\hat{a}_{i_1,i_2}\binom{m_1}{i_1}\binom{m_2}{i_2} \quad \begin{matrix} i_1 = 0, \ldots, m_1; \\ i_2 = 0, \ldots, m_2. \end{matrix}$$

   The entries of this subresultant variant can span many orders of magnitude when m_1 or m_2 is large. Each coefficient of $\hat{f}(x,y)$ appears in each of the $(n_1 - k_1 + 1) \times (n_2 - k_2 + 1)$ columns of the first partition.

2. The second variant of the (k_1, k_2)th subresultant matrix is given by $D^{-1}_{k_1,k_2} T_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y))$ and entries in the first partition are of the form

$$\frac{\hat{a}_{i_1,i_2}\binom{m_1}{i_1}\binom{m_2}{i_2}}{\binom{m_1+n_1-k_1}{i_1+j_1}\binom{m_2+n_2-k_2}{i_2+j_2}} \quad \begin{matrix} i_1 = 0, \ldots, m_1; \quad j_1 = 0, \ldots, n_1 - k_1; \\ i_2 = 0, \ldots, m_2; \quad j_2 = 0, \ldots, n_2 - k_2. \end{matrix}$$

3. The third subresultant matrix variant is given by $T_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y))\hat{Q}_{k_1,k_2}$ and entries in the first partition are of the form

$$\hat{a}_{i_1,i_2}\binom{m_1}{i_1}\binom{m_2}{i_2}\binom{n_1-k_1}{j_1}\binom{n_2-k_2}{j_2} \quad \begin{matrix} i_1 = 0, \ldots, m_1; \quad j_1 = 0, \ldots, n_1 - k_1; \\ i_2 = 0, \ldots, m_2; \quad j_2 = 0, \ldots, n_2 - k_2. \end{matrix}$$

4. The fourth variant of the (k_1, k_2)th subresultant matrix is given by $D^{-1}_{k_1,k_2} T_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y))\hat{Q}_{k_1,k_2}$ and the first partition contains entries of the form

$$\frac{\hat{a}_{i_1,i_2}\binom{m_1}{i_1}\binom{m_2}{i_2}\binom{n_1-k_1}{j_1}\binom{n_2-k_2}{j_2}}{\binom{m_1+n_1-k_1}{i_1+j_1}\binom{m_2+n_2-k_2}{i_2+j_2}} \qquad \begin{matrix} i_1 = 0,\ldots,m_1; & j_1 = 0,\ldots,n_1-k_1; \\ i_2 = 0,\ldots,m_2; & j_2 = 0,\ldots,n_2-k_2. \end{matrix}$$

The Three-Polynomial Subresultant Matrices

The (k_1, k_2)th (3×3) partitioned subresultant matrix was defined in (7.8) and the variants of the (3×3) partitioned subresultant matrices are given by

(i) $\{\tilde{T}_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))\}$

(ii) $\{\tilde{D}^{-1}_{k_1,k_2}\tilde{T}_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))\}$

(iii) $\{\tilde{T}_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))\tilde{Q}_{k_1,k_2}\}$

(iv) $\{\tilde{D}^{-1}_{k_1,k_2}\tilde{T}_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))\tilde{Q}_{k_1,k_2}\}$

for $k_1 = 0,\ldots,\min(m_1, n_1, o_1)$; $k_2 = 0,\ldots,\min(m_2, n_2, o_2)$.

The variants of the (k_1, k_2)th subresultant matrix contain partitions of the same form as the partitions of the two-polynomial subresultant matrices described in (7.2), but the second row-partition is now included, which contains two additional non-zero partitions.

The (k_1, k_2)th (2×3) partitioned subresultant matrix is given by

$$\hat{S}_{k_1,k_2}\left(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y)\right) = \hat{D}^{-1}_{k_1,k_2}\hat{T}_{k_1,k_2}\left(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y)\right)\tilde{Q}_{k_1,k_2}.$$

The matrix $\hat{D}^{-1}_{k_1,k_2}$ is given by

$$\hat{D}^{-1}_{k_1,k_2} = \text{diag}\left[\ D^{-1}_{m_1+n_1-k_1,m_2+n_2-k_2},\quad D^{-1}_{m_1+o_1-k_1,m_2+o_2-k_2}\ \right],$$

where the matrices on the diagonal are of the same form as the matrix defined in (7.3). The matrix $\hat{T}_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))$ is given by

$$\begin{bmatrix} T_{n_1-k_1,n_2-k_2}\left(\hat{f}(x,y)\right) & & T_{m_1-k_1,m_2-k_2}\left(\hat{g}(x,y)\right) \\ & T_{o_1-k_1,o_2-k_2}\left(\hat{f}(x,y)\right) & T_{m_1-k_1,m_2-k_2}\left(\hat{h}(x,y)\right) \end{bmatrix},$$

where the partitions are of the form (2.20), and $\tilde{Q}_{k_1,k_2}$ is already defined in (7.9).

The variants of the (2×3) partitioned subresultant matrices are therefore given by:

(i) $\{\hat{T}_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))\}$

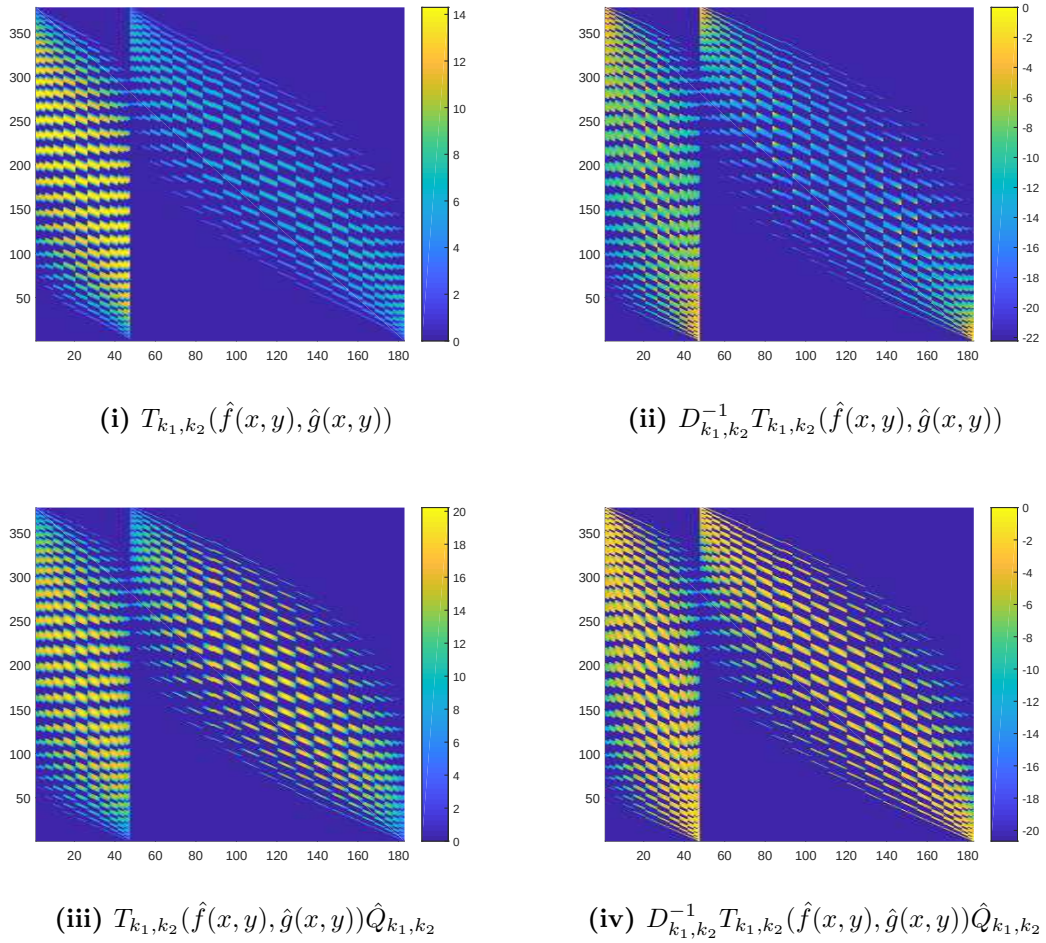
(ii) $\{\hat{D}^{-1}_{k_1,k_2}\hat{T}_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))\}$

(iii) $\{\hat{T}_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))\tilde{Q}_{k_1,k_2}\}$

(iv) $\{\hat{D}^{-1}_{k_1,k_2}\hat{T}_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y), \hat{h}(x,y))\tilde{Q}_{k_1,k_2}\}$

for $k_1 = 0,\ldots,\min(m_1, n_1, o_1)$; $k_2 = 0,\ldots,\min(m_2, n_2, o_2)$.

Again, the partitions of these matrices follow from the definitions of the two-polynomial subresultant matrix variants but six non-zero partitions are contained rather than two.



(i) $T_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y))$ **(ii)** $D^{-1}_{k_1,k_2}T_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y))$

(iii) $T_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y))\hat{Q}_{k_1,k_2}$ **(iv)** $D^{-1}_{k_1,k_2}T_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y))\hat{Q}_{k_1,k_2}$

Figure 7.1: Heat map of the coefficient multipliers in the subresultant matrix variants where $k_1 = 1$ and $k_2 = 1$ in Example 7.2.1

Example 7.2.1. Consider the polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ with degree structures $(m_1,m_2) = (10,15)$ and $(n_1,n_2) = (9,6)$. Figure 7.1 shows heat maps of the scaling effect of the coefficient multipliers in each entry of the first subresultant matrix for each of the four the subresultant variants (i) $T_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y))$, (ii) $D^{-1}_{k_1,k_2}T_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y))$, (iii) $T_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y))\hat{Q}_{k_1,k_2}$ and (iv) $D^{-1}_{k_1,k_2}T_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y))\hat{Q}_{k_1,k_2}$, where $k_1 = 1$ and $k_2 = 1$.

In Figure 7.1i the entries in the first partition are significantly larger than the entries in the second due to the presence of the binomial terms $\binom{m_1}{i_1}\binom{m_2}{i_2}$, which are significantly larger than the terms $\binom{n_1}{i_1}\binom{n_2}{i_2}$. The largest of the coefficient multipliers in the first partition is equal to $\binom{10}{5}\binom{15}{7} = 1621620$, while the largest in the second partition is equal to $\binom{9}{4}\binom{6}{3} = 2520$. In Figure 7.1ii the entries in the second partition are still significantly smaller than entries in the first partition. In Figure 7.1iii entries in the second partition are much larger due to the presence of the binomial terms $\binom{m_1-k_1}{j_1}\binom{m_2-k_2}{j_2}$, which are much larger than the binomial terms $\binom{n_1-k_1}{j_1}\binom{n_2-k_2}{j_2}$ in the coefficient multipliers of the first partition.

As with the univariate subresultant matrices, optimally scaled entries are given in the variant of the form $D^{-1}_{k_1,k_2}T_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y))\hat{Q}_{k_1,k_2}$, which can be seen in Figure 7.1iv.

Example 7.2.2. Consider the Bernstein form of the exact polynomials $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ of degrees $(17,13)$, $(20,19)$ and $(10,13)$ respectively, whose factorised forms

are given by

$$\hat{f}(x,y) = (x - 0.554687987932164654)^3(x + 0.21657951321)(x + y - 0.46578784351654)^3 \\ (x + y + 0.0124)^6(x^2 + y^2 + 0.5679814324687)^2 \quad (7.13)$$

$$\hat{g}(x,y) = (x + 0.21657951321)(x + y - 0.46578784351654)^3(x + y + 0.4512)^6 \\ (x^2 + y^2 - 0.00104751807)^3(x^2 + y^2 + 0.5679814324687)^2 \quad (7.14)$$

$$\hat{h}(x,y) = (x + 0.21657951321)(y - 0.2465879841351465498)^4(x + y - 0.46578784351654)^3 \\ (12x^2 + y^2 - 0.348798)(x^2 + y^2 + 0.5679814324687)^2. \quad (7.15)$$

The polynomials $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ have a GCD $\hat{d}_{t_1,t_2}(x,y)$ of degree $(8,7)$, which is given by

$$\hat{d}(x,y) = (x + 0.21657951321)(x^2 + y^2 + 0.5679814324687)^2(x + y - 0.46578784351654)^3.$$

Noise is added to the coefficients of $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ such that the coefficients of the inexact polynomials $f(x,y)$, $g(x,y)$ and $h(x,y)$ are given by

$$\begin{aligned} a_{i_1,i_2} &= \hat{a}_{i_1,i_2} + \epsilon_{f,i_1,i_2}\hat{a}_{i_1,i_2}r_{f,i_1,i_2} \quad \text{for} \quad i_1 = 0,\ldots,m_1; \quad i_2 = 0,\ldots,m_2, \\ b_{j_1,j_2} &= \hat{b}_{i_1,i_2} + \epsilon_{g,j_1,j_2}\hat{b}_{j_1,j_2}r_{g,j_1,j_2} \quad \text{for} \quad i_1 = 0,\ldots,n_1; \quad i_2 = 0,\ldots,n_2, \\ c_{p_1,p_2} &= \hat{c}_{p_1,p_2} + \epsilon_{h,p_1,p_2}\hat{c}_{p_1,p_2}r_{h,p_1,p_2} \quad \text{for} \quad i_1 = 0,\ldots,o_1; \quad i_2 = 0,\ldots,o_2, \end{aligned} \quad (7.16)$$

where $\{\epsilon_{f,i_1,i_2}\}$, $\{\epsilon_{g,j_1,j_2}\}$ and $\{\epsilon_{h,p_1,p_2}\}$ are uniformly distributed random variables in the interval $[10^{-12}, 10^{-10}]$ and $\{r_{f,i_1,i_2}\}$, $\{r_{g,j_1,j_2}\}$ and $\{r_{h,p_1,p_2}\}$ are uniformly distributed random variables in the interval $[-1,1]$. The coefficients of $f(x,y)$, $g(x,y)$ and $h(x,y)$ are plotted in Figure 7.2, where it can be seen that the coefficients of $f(x,y)$ and $g(x,y)$ span many more orders of magnitude than those of $h(x,y)$.

Heat maps of the coefficient multipliers in the variants of the subresultant matrices are plotted in Figure 7.3 and the following observations are made:

1. Figure 7.3i plots the magnitude of the coefficient multipliers in the entries of $\hat{T}_{k_1,k_2}(f(x,y), g(x,y), h(x,y))$. It can be seen that the top right partition, whose entries are of the form $b_{i_1,i_2}\binom{n_1}{i_1}\binom{n_2}{i_2}$, has entries of significantly larger magnitude than those in the remaining three non-zero partitions.

2. Figure 7.3iii plots the magnitude of the coefficient multipliers in $\hat{T}_{k_1,k_2}(f(x,y), g(x,y), h(x,y))\tilde{Q}_{k_1,k_2}$. The first row-partition of this matrix, $\mathcal{R}_{a,k_1,k_2}$, contains coefficient multipliers which are significantly larger than the coefficient multipliers of the second row-partition $\mathcal{R}_{b,k_1,k_2}$.

3. Figure 7.3iv plots the magnitude of the coefficient multipliers in $\hat{D}^{-1}_{k_1,k_2}\hat{T}_{k_1,k_2}(f(x,y), g(x,y), h(x,y))\tilde{Q}_{k_1,k_2}$, which appears to have the optimal form of scaling.

The minimum singular values of the subresultant matrices

(i) $\{\hat{T}_{k_1,k_2}(f(x,y), g(x,y), h(x,y))\}$

(ii) $\{\hat{D}^{-1}_{k_1,k_2}\hat{T}_{k_1,k_2}(f(x,y), g(x,y), h(x,y))\}$

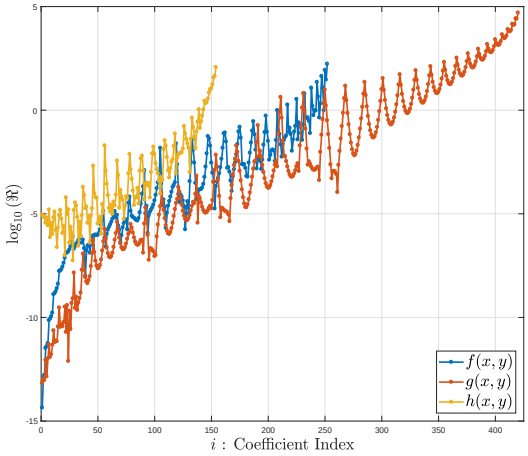


Figure 7.2: The coefficients of the polynomials $f(x,y)$, $g(x,y)$ and $h(x,y)$ in Example 7.2.2

(iii) $\{\hat{T}_{k_1,k_2}(f(x,y),g(x,y),h(x,y))\tilde{Q}_{k_1,k_2}\}$

(iv) $\{\hat{D}^{-1}_{k_1,k_2}\hat{T}_{k_1,k_2}(f(x,y),g(x,y),h(x,y))\tilde{Q}_{k_1,k_2}\}$

for $k_1 = 1,\ldots,\min(m_1,n_1,o_1)$; $k_2 = 1,\ldots,\min(m_2,n_2,o_2)$ are computed and plotted in Figures 7.4i to 7.4iv. From these sets of minimum singular values, it is clear that $\{\hat{D}^{-1}_{k_1,k_2}\hat{T}_{k_1,k_2}(f(x,y),g(x,y),h(x,y))\tilde{Q}_{k_1,k_2}\}$ is the optimal subresultant matrix sequence for the computation of the degree of the AGCD since the separation between the numerically zero and non-zero minimum singular values is more pronounced.

The results in this example can be further improved by preprocessing the three-polynomial subresultant matrices and this will be considered in Section 7.3. This example is extended in Example 7.6.4 to consider higher levels of noise and the effect of preprocessing on the computation of the degree of the AGCD.

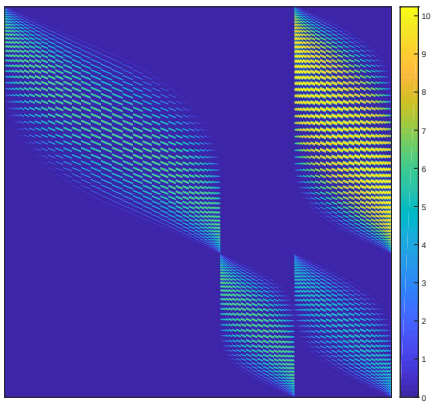
□

The results in this section are consistent with earlier results for the two-polynomial and three-polynomial problems for univariate and bivariate polynomials defined over a rectangular domain. Despite $D^{-1}_{k_1,k_2}T_{k_1,k_2}Q_{k_1,k_2}$ being the optimal variant of the four subresultant matrix variants, the ratio of entry of maximum magnitude to entry of minimum magnitude is still likely to be large and again preprocessing the three polynomials polynomials is considered.

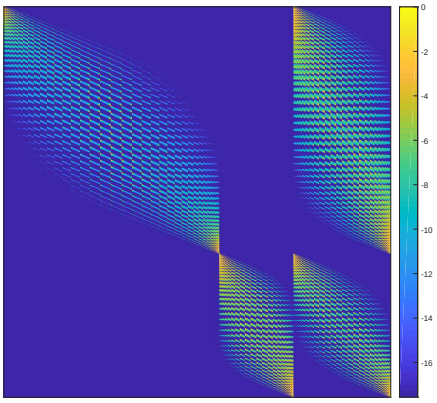
7.3 Preprocessing

The three preprocessing operations have been discussed for the two-polynomial and three-polynomial subresultant matrices of both univariate and bivariate polynomials where the bivariate polynomials were defined over a triangular domain.

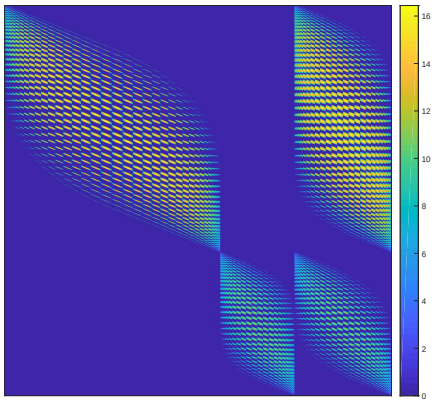
The equivalent three preprocessing operations are now considered for the two-polynomial and three-polynomial subresultant matrices for bivariate polynomials defined over a rectangular domain. In this section the two-polynomial and three-polynomial preprocessing stages will be considered simultaneously, since the extension from preprocess-



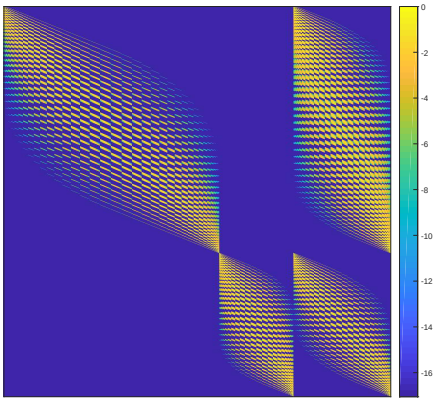
(i) $\hat{T}_{k_1,k_2}(f(x,y),g(x,y),h(x,y))$



(ii) $\hat{D}^{-1}_{k_1,k_2}\hat{T}_{k_1,k_2}(f(x,y),g(x,y),h(x,y))$



(iii) $\hat{T}_{k_1,k_2}(f(x,y),g(x,y),h(x,y))\tilde{Q}_{k_1,k_2}$



(iv) $\hat{D}^{-1}_{k_1,k_2}\hat{T}_{k_1,k_2}(f(x,y),g(x,y),h(x,y))\tilde{Q}_{k_1,k_2}$

Figure 7.3: Heat map of the coefficient multipliers (using logarithmic scale) in the variants of the subresultant matrices in Example 7.2.2.

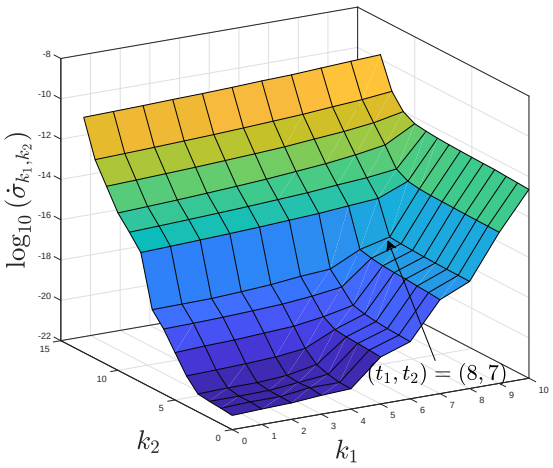
ing the two-polynomial subresultant matrix to the three-polynomial subresultant matrix is easily derived.

The three preprocessing stages developed in Section 3.4 are adapted for the computation of preprocessed two-polynomial and three-polynomial subresultant matrices of bivariate polynomials in Bernstein form.

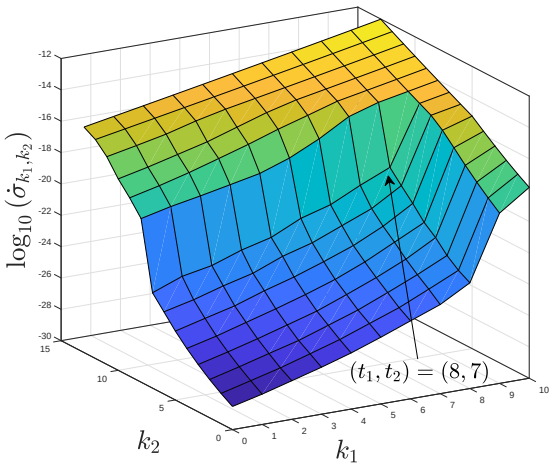
Normalisation by Geometric Means in the Two-Polynomial Subresultant Matrices

Polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ are normalised by the geometric mean of their non-zero entries in the matrix partitions $C_{n_1-k_1,n_2-k_2}(\hat{f}(x,y))$ and $C_{m_1-k_1,m_2-k_2}(\hat{g}(x,y))$ respectively. The geometric mean of the non-zero entries in $C_{n_1-k_1,n_2-k_2}(\hat{f}(x,y))$, denoted $\mathcal{G}_{n_1-k_1,n_2-k_2}(\hat{f}(x,y))$, is given by

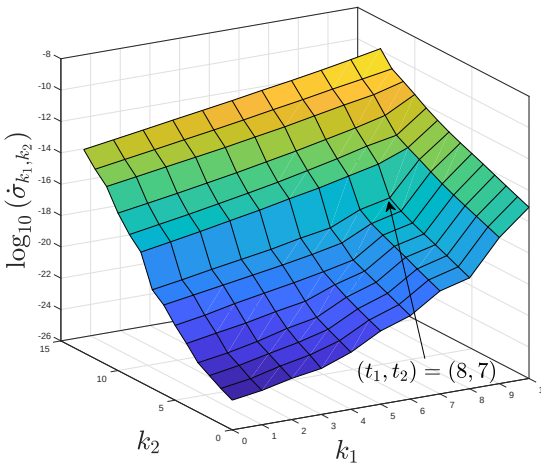
$$\prod_{j_2=0}^{n_2-k_2}\prod_{j_1=0}^{n_1-k_1}\prod_{i_2=0}^{m_2}\prod_{i_1=0}^{m_1}\left(\hat{a}_{i_1,i_2}\frac{\binom{i_1+j_1}{i_1}\binom{m_1+n_1-k_1-i_1}{n_1-k_1-i_1-j_1}}{\binom{m_1+n_1-k_1}{n_1-k_1}}\frac{\binom{i_2+j_2}{i_2}\binom{m_2+n_2-k_2-i_2}{n_2-k_2-i_2-j_2}}{\binom{m_2+n_2-k_2}{n_2-k_2}}\right)^{\frac{1}{r\times c}}, \tag{7.17}$$



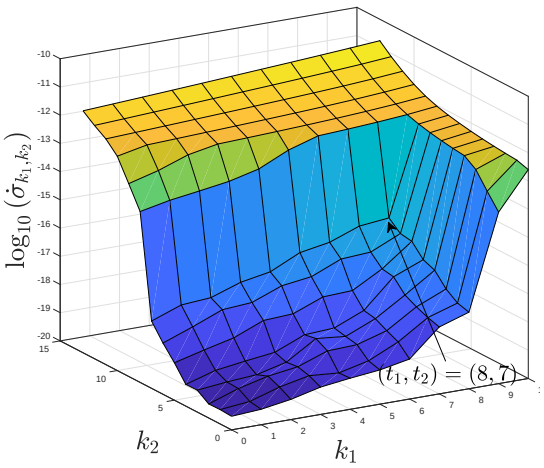
(i) $\{\hat{T}_{k_1,k_2}(f(x,y), g(x,y), h(x,y))\}$



(ii) $\{\hat{D}^{-1}_{k_1,k_2}\hat{T}_{k_1,k_2}(f(x,y), g(x,y), h(x,y))\}$



(iii) $\{\hat{T}_{k_1,k_2}(f(x,y), g(x,y), h(x,y))\tilde{Q}_{k_1,k_2}\}$



(iv)
$\{\hat{D}^{-1}_{k_1,k_2}\hat{T}_{k_1,k_2}(f(x,y), g(x,y), h(x,y))\tilde{Q}_{k_1,k_2}\}$

Figure 7.4: The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of each subresultant matrix for each of the four subresultant matrix variants in Example 7.2.2.

where

$$r = (m_1 + 1)(m_2 + 1) \qquad \text{and} \qquad c = (n_1 - k_1 + 1)(n_2 - k_2 + 1)$$

and a similar expression for $\mathcal{G}_{m_1-k_1,m_2-k_2}(\hat{g}(x,y))$ can be derived such that the normalised polynomials $\bar{f}_{k_1,k_2}(x,y)$ and $\bar{g}_{k_1,k_2}(x,y)$ are given by

$$\bar{f}_{k_1,k_2}(x,y) = \frac{\hat{f}(x,y)}{\mathcal{G}_{n_1-k_1,n_2-k_2}\left(\hat{f}(x,y)\right)},$$
$$\bar{g}_{k_1,k_2}(x,y) = \frac{\hat{g}(x,y)}{\mathcal{G}_{m_1-k_1,m_2-k_2}\left(\hat{g}(x,y)\right)}.$$

An efficient method for the computation of the geometric mean is given in Appendix C.3.1 which extends the method seen in Section 3.4.

Normalisation by Geometric Means in the (2×3) Partitioned Subresultant Matrices

The polynomials $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ are normalised by the geometric mean of their entries in the (k_1,k_2)th subresultant matrix $\hat{S}_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$, where the coefficients of $\hat{f}(x,y)$ appear in two partitions, $C_{n_1-k_1,n_2-k_2}(\hat{f}(x,y))$ and $C_{o_1-k_1,o_2-k_2}(\hat{f}(x,y))$, and the coefficients of $\hat{g}(x,y)$ and $\hat{h}(x,y)$ only appear in one partition each. These are $C_{m_1-k_1,m_2-k_2}(\hat{g}(x,y))$ and $C_{m_1-k_1,m_2-k_2}(\hat{h}(x,y))$ respectively. The normalised polynomials are therefore given by

$$\bar{f}_{k_1,k_2}(x,y) = \frac{\hat{f}(x,y)}{\hat{\mathcal{G}}_{k_1,k_2}\left(\hat{f}(x,y)\right)},$$
$$\bar{g}_{k_1,k_2}(x,y) = \frac{\hat{g}(x,y)}{\mathcal{G}_{m_1-k_1,m_2-k_2}\left(\hat{g}(x,y)\right)},$$
$$\bar{h}_{k_1,k_2}(x,y) = \frac{\hat{h}(x,y)}{\mathcal{G}_{m_1-k_1,m_2-k_2}\left(\hat{h}(x,y)\right)},$$

where $\mathcal{G}_{m_1-k_1,m_2-k_2}(\hat{g}(x,y))$ and $\mathcal{G}_{m_1-k_1,m_2-k_2}(\hat{h}(x,y))$ are given by (7.17), and $\hat{\mathcal{G}}_{k_1,k_2}(\hat{f}(x,y))$ is the geometric mean of $\hat{f}(x,y)$ in two partitions of the (k_1,k_2)th subresultant matrix.

Normalisation by Geometric Means in the (2×3) Partitioned Subresultant Matrices

In the (3×3) subresultant matrix, all three of the polynomials $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ appear in two partitions each, and so the normalised polynomials are given by

$$\bar{f}_{k_1,k_2}(x,y) = \frac{\hat{f}(x,y)}{\hat{\mathcal{G}}_{k_1,k_2}\left(\hat{f}(x,y)\right)},$$
$$\bar{g}_{k_1,k_2}(x,y) = \frac{\hat{g}(x,y)}{\hat{\mathcal{G}}_{k_1,k_2}\left(\hat{g}(x,y)\right)},$$
$$\bar{h}_{k_1,k_2}(x,y) = \frac{\hat{h}(x,y)}{\hat{\mathcal{G}}_{k_1,k_2}\left(\hat{h}(x,y)\right)},$$

where $\hat{\mathcal{G}}_{k_1,k_2}(\hat{f}(x,y))$ is the geometric mean of $\hat{f}(x,y)$ in two partitions of the (k_1,k_2)th subresultant matrix.

The Two-Polynomial Subresultant Matrix Optimisation Problem

To scale the two partitions of the two-polynomial subresultant matrix, the second partition is scaled by λ and the independent variables x and y are replaced such that $x = \theta_1\omega_1$ and $y = \theta_2\omega_2$. The optimal values of λ, θ_1 and θ_2 are computed such that the ratio of entry of maximum magnitude to entry of minimum magnitude in the (k_1,k_2)th subresultant

matrix, given by

$$\left[\; C_{n_1-k_1,n_2-k_2}\left(\lambda\ddot{f}(\theta_1,\theta_2,\omega_1,\omega_2)\right),\quad C_{m_1-k_1,m_2-k_2}\left(\ddot{g}(\theta_1,\theta_2,\omega_1,\omega_2)\right)\;\right],$$

is minimised. The unoptimised polynomials $\lambda\ddot{f}(\theta_1,\theta_2,\omega_1,\omega_2)$ and $\ddot{g}(\theta_1,\theta_2,\omega_1,\omega_2)$ are given by

$$\lambda\ddot{f}(\theta_1,\theta_2,\omega_1,\omega_2)=\lambda\sum_{i_2=0}^{m_2}\sum_{i_1=0}^{m_1}a_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{m_1}{i_1}\binom{m_2}{i_2}(1-\theta_1\omega_1)^{m_1-i_1}(1-\theta_2\omega_2)^{m_2-i_2}\omega_1^{i_1}\omega_2^{i_2}$$

$$\ddot{g}(\theta_1,\theta_2,\omega_1,\omega_2)=\sum_{i_2=0}^{n_2}\sum_{i_1=0}^{n_1}\bar{b}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{n_1}{i_1}\binom{n_2}{i_2}(1-\theta_1\omega_1)^{n_1-i_1}(1-\theta_2\omega_2)^{n_2-i_2}\omega_1^{i_1}\omega_2^{i_2}.$$

Let the sets of all non-zero entries in the first and second partitions of the (k_1,k_2)th subresultant matrix be denoted $\mathscr{P}_{1,k_1,k_2}(\lambda,\theta_1,\theta_2)$ and $\mathscr{P}_{2,k_1,k_2}(\theta_1,\theta_2)$ respectively, where

$$\mathscr{P}_{1,k_1,k_2}=\left\{\frac{\left|\lambda\bar{a}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{m_1}{i_1}\binom{m_2}{i_2}\binom{n_1-k_1}{j_1}\binom{n_2-k_2}{j_2}\right|}{\binom{m_1+n_1-k_1}{i_1+j_1}\binom{m_2+n_2-k_2}{i_2+j_2}}\right\}\quad\begin{matrix}i_1=0,\ldots,m_1;\quad j_1=0,\ldots,n_1-k_1,\\ i_2=0,\ldots,m_2;\quad j_2=0,\ldots,n_2-k_2,\end{matrix}\tag{7.18}$$

$$\mathscr{P}_{2,k_1,k_2}=\left\{\frac{\left|\bar{b}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{n_1}{i_1}\binom{n_2}{i_2}\binom{m_1-k_1}{j_1}\binom{m_2-k_2}{j_2}\right|}{\binom{m_1+n_1-k_1}{i_1+j_1}\binom{m_2+n_2-k_2}{i_2+j_2}}\right\}\quad\begin{matrix}i_1=0,\ldots,n_1;\quad j_1=0,\ldots,m_1-k_1,\\ i_2=0,\ldots,n_2\quad\; j_2=0,\ldots,m_2-k_2.\end{matrix}\tag{7.19}$$

The optimal values of λ, θ_1 and θ_2 are given when the ratio of the entry of maximum magnitude to entry of minimum magnitude is minimised. The minimisation problem can be written as

$$(\lambda_{k_1,k_2},\theta_{1,k_1,k_2},\theta_{2,k_1,k_2})=\arg\min_{\lambda,\theta_1,\theta_2}\left\{\frac{\max\{\max\{\mathscr{P}_{1,k_1,k_2}(\lambda,\theta_1,\theta_2)\},\max\{\mathscr{P}_{2,k_1,k_2}(\theta_1,\theta_2)\}\}}{\min\{\min\{\mathscr{P}_{1,k_1,k_2}(\lambda,\theta_1,\theta_2)\},\min\{\mathscr{P}_{2,k_1,k_2}(\theta_1,\theta_2)\}\}}\right\}.\tag{7.20}$$

The minimisation problem is detailed in Appendix C.3.2 and the optimal values λ_{k_1,k_2}, θ_{1,k_1,k_2} and θ_{2,k_1,k_2} are given as the solution of a linear programming problem such that the preprocessed polynomials $\lambda_{k_1,k_2}\tilde{f}_{k_1,k_2}(\omega_1,\omega_2)$ and $\tilde{g}_{k_1,k_2}(\omega_1,\omega_2)$ are given by

$$\begin{aligned}\lambda_{k_1,k_2}\tilde{f}_{k_1,k_2}(\omega_1,\omega_2)=\lambda_{k_1,k_2}&\sum_{i_2=0}^{m_2}\sum_{i_1=0}^{m_1}\bar{a}_{i_1,i_2}\theta_{1,k_1,k_2}^{i_1}\theta_{2,k_1,k_2}^{i_2}\\ &(1-\theta_{1,k_1,k_2}\omega_1)^{m_1-i_1}(1-\theta_{2,k_1,k_2}\omega_2)^{m_2-i_2}\omega_1^{i_1}\omega_2^{i_2}\\ \tilde{g}_{k_1,k_2}(\omega_1,\omega_2)=&\sum_{i_2=0}^{n_2}\sum_{i_1=0}^{n_1}\bar{b}_{i_1,i_2}\theta_{1,k_1,k_2}^{i_1}\theta_{2,k_1,k_2}^{i_2}(1-\theta_{1,k_1,k_2})^{n_1-i_1}(1-\theta_{2,k_1,k_2})^{n_2-i_2}\omega_1^{i_1}\omega_2^{i_2}.\end{aligned}$$

The Three-Polynomial Subresultant Matrix Optimisation Problem

The (2×3) partitioned subresultant matrices have the two additional non-zero partitions $C_{o_1-k_1,o_2-k_2}(\hat{f}(x,y))$ and $C_{m_1-k_1,m_2-k_2}(\hat{h}(x,y))$. The sets $\mathscr{P}_{1,k_1,k_2}(\lambda,\theta_1,\theta_2)$ and $\mathscr{P}_{2,k_1,k_2}(\theta_1,\theta_2)$ are already defined in (7.18) and (7.19), and the sets of entries in the

additional partitions denoted $\mathscr{P}_{3,k_1,k_2}(\lambda,\theta_1,\theta_2)$ and $\mathscr{P}_{4,k_1,k_2}(\rho,\theta_1,\theta_2)$ are given by

$$\mathscr{P}_{3,k_1,k_2}(\lambda,\theta_1,\theta_2)=\left\{\frac{\lambda\bar{a}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{m_1}{i_1}\binom{m_2}{i_2}\binom{o_1-k_1}{j_1}\binom{o_2-k_2}{j_2}}{\binom{m_1+o_1-k_1}{i_1+j_1}\binom{m_2+o_2-k_2}{i_2+j_2}}\right\}\begin{array}{l}i_1=0,\ldots,m_1;\quad j_1=0,\ldots,o_1-k_1,\\ i_2=0,\ldots,m_2;\quad j_2=0,\ldots,o_2-k_2,\end{array}\tag{7.21}$$

$$\mathscr{P}_{4,k_1,k_2}(\rho,\theta_1,\theta_2)=\left\{\frac{\rho\bar{c}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{o_1}{i_1}\binom{o_2}{i_2}\binom{m_1-k_1}{j_1}\binom{m_2-k_2}{j_2}}{\binom{m_1+o_1-k_1}{i_1+j_1}\binom{m_2+o_2-k_2}{i_2+j_2}}\right\}\begin{array}{l}i_1=0,\ldots,o_1;\quad j_1=0.\ldots,m_1-k_1,\\ i_2=0,\ldots,o_2;\quad j_2=0,\ldots,m_2-k_2.\end{array}\tag{7.22}$$

The minimisation problem for the (2×3) partitioned subresultant matrix can be written as an extension of (7.20) and is given by

$$\begin{aligned}(\lambda_{k_1,k_2},\rho_{k_1,k_2},\theta_{1,k_1,k_2},\theta_{2,k_1,k_2})=\quad \arg\min_{\lambda,\rho,\theta_1,\theta_2}\Bigg\{&\frac{\max\{\max\{P_{1,k_1,k_2}(\lambda,\theta_1,\theta_2)\},\max\{P_{2,k_1,k_2}(\theta_1,\theta_2)\},}{\min\{\min\{P_{1,k_1,k_2}(\lambda,\theta_1,\theta_2)\},\min\{P_{2,k_1,k_2}(\theta_1,\theta_2)\},}\\ &\frac{\max\{\mathscr{P}_{3,k_1,k_2}(\lambda,\theta_1,\theta_2)\},\max\{\mathscr{P}_{4,k_1,k_2}(\rho,\theta_1,\theta_2)\}\}}{\min\{\mathscr{P}_{3,k_1,k_2}(\lambda,\theta_1,\theta_2)\},\min\{\mathscr{P}_{4,k_1,k_2}(\rho,\theta_1,\theta_2)\}\}}\Bigg\},\end{aligned}$$

where the linear programming problem for determining the optimal values of λ, ρ, θ_1 and θ_2 is detailed in Appendix C.3.3.

The method of minimising the (2×3) subresultant matrices can be extended to minimise the (3×3) partitioned subresultant matrix, and values λ, ρ, θ_1 and θ_2 are given by the minimisation of

$$\begin{aligned}(\lambda_{k_1,k_2},\rho_{k_1,k_2},\theta_{1,k_1,k_2},\theta_{2,k_1,k_2})=\arg\min_{\lambda,\rho,\theta_1,\theta_2}\Bigg\{&\frac{\max\{\max\{\mathscr{P}_{1,k_1,k_2}(\lambda,\theta_1,\theta_2)\},\max\{\mathscr{P}_{2,k_1,k_2}(\theta_1,\theta_2)\},}{\min\{\min\{\mathscr{P}_{1,k_1,k_2}(\lambda,\theta_1,\theta_2)\},\min\{\mathscr{P}_{2,k_1,k_2}(\theta_1,\theta_2)\},}\\ &\frac{\max\{\mathscr{P}_{3,k_1,k_2}(\lambda,\theta_1,\theta_2)\},\max\{\mathscr{P}_{4,k_1,k_2}(\rho,\theta_1,\theta_2)\}}{\min\{\mathscr{P}_{3,k_1,k_2}(\lambda,\theta_1,\theta_2)\},\min\{\mathscr{P}_{4,k_1,k_2}(\rho,\theta_1,\theta_2)\}}\\ &\frac{\max\{\mathscr{P}_{5,k_1,k_2}(\rho,\theta_1,\theta_2)\},\max\{\mathscr{P}_{6,k_1,k_2}(\theta_1,\theta_2)\}\}}{\min\{\mathscr{P}_{5,k_1,k_2}(\rho,\theta_1,\theta_2)\},\min\{\mathscr{P}_{6,k_1,k_2}(\theta_1,\theta_2)\}\}}\Bigg\},\end{aligned}$$

where

$$\mathscr{P}_{5,k_1,k_2}(\rho,\theta_1,\theta_2)=\left\{\frac{\rho\bar{c}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{o_1}{i_1}\binom{o_2}{i_2}\binom{n_1-k_1}{j_1}\binom{n_2-k_2}{j_2}}{\binom{n_1+o_1-k_1}{i_1+j_1}\binom{n_2+o_2-k_2}{i_2+j_2}}\right\}\begin{array}{l}i_1=0,\ldots,o_1;\quad j_1=0.\ldots,n_1-k_1,\\ i_2=0,\ldots,o_2;\quad j_2=0,\ldots,n_2-k_2,\end{array}\tag{7.23}$$

$$\mathscr{P}_{6,k_1,k_2}(\theta_1,\theta_2)=\left\{\frac{\left|\bar{b}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{n_1}{i_1}\binom{n_2}{i_2}\binom{o_1-k_1}{j_1}\binom{o_2-k_2}{j_2}\right|}{\binom{n_1+o_1-k_1}{i_1+j_1}\binom{n_2+o_2-k_2}{i_2+j_2}}\right\}\begin{array}{l}i_1=0,\ldots,n_1;\quad j_1=0,\ldots,o_1-k_1;\\ i_2=0,\ldots,n_2;\quad j_2=0,\ldots,o_2-k_2.\end{array}\tag{7.24}$$

The preprocessed polynomials $\lambda_{k_1,k_2}\tilde{f}_{k_1,k_2}(\omega_1,\omega_2)$, $\tilde{g}_{k_1,k_2}(\omega_1,\omega_2)$ and $\rho_{k_1,k_2}\tilde{h}(\omega_1,\omega_2)$ are

given by

$$\lambda_{k_1,k_2}\tilde{f}_{k_1,k_2}(\omega_1,\omega_2) = \lambda_{k_1,k_2}\sum_{i_2=0}^{m_2}\sum_{i_1=0}^{m_1}\bar{a}_{i_1,i_2}\theta_{1,k_1,k_2}^{i_1}\theta_{2,k_1,k_2}^{i_2}\binom{m_1}{i_1}\binom{m_2}{i_2}\times$$
$$(1-\theta_{1,k_1,k_2}\omega_1)^{m_1-i_1}(1-\theta_{2,k_1,k_2}\omega_2)^{m_2-i_2}\omega_1^{i_1}\omega_2^{i_2}$$
$$\tilde{g}_{k_1,k_2}(\omega_1,\omega_2) = \sum_{i_2=0}^{n_2}\sum_{i_1=0}^{n_1}\bar{b}_{i_1,i_2}\theta_{1,k_1,k_2}^{i_1}\theta_{2,k_1,k_2}^{i_2}\binom{n_1}{i_1}\binom{n_2}{i_2}\times$$
$$(1-\theta_{1,k_1,k_2}\omega_1)^{n_1-i_1}(1-\theta_{2,k_1,k_2}\omega_2)^{n_2-i_2}\omega_1^{i_1}\omega_2^{i_2}$$
$$\rho_{k_1,k_2}\tilde{h}_{k_1,k_2}(\omega_1,\omega_2) = \rho_{k_1,k_2}\sum_{i_2=0}^{o_2}\sum_{i_1=0}^{o_1}\bar{c}_{i_1,i_2}\theta_{1,k_1,k_2}^{i_1}\theta_{2,k_1,k_2}^{i_2}\binom{o_1}{i_1}\binom{o_2}{i_2}\times$$
$$(1-\theta_{1,k_1,k_2}\omega_1)^{o_1-i_1}(1-\theta_{2,k_1,k_2}\omega_2)^{o_2-i_2}\omega_1^{i_1}\omega_2^{i_2}.$$

This section has presented the extensions necessary to preprocess the subresultant matrices of two or three bivariate polynomials in Bernstein form. It will be shown in Section 7.6 that preprocessing the polynomials in the two-polynomial and three-polynomial subresultant matrices yields improved results for the computation of the degree of the GCD and approximations of its coefficients.

Preprocessing the subresultant matrices has a significant cost due to the repeated use of linear programming. In the two-polynomial problem the solution of a linear programming problem gives the optimal values (θ_1, θ_2, λ). Similarly, in the three-polynomial problem, the solution of a linear programming problem gives the optimal values (θ_1, θ_2, λ, ρ). A new linear programming problem must be considered for each subresultant matrix in a two-dimensional array.

Methods have been considered for the efficient computation of the geometric means of polynomials $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ in the sequence of subresultant matrices (Appendix C.3.1). Despite offering a clean and efficient expression for these geometric means, the reduction in complexity of the complete algorithm is minimal.

The cost of the algorithm is still significant due to the repeated use of the singular value decomposition for each of the subresultant matrices in a two-dimensional array. The next section considers a new method for the computation of the degree of the GCD of two bivariate polynomials in Bernstein form. This method still makes use of a sequence of subresultant matrices, but reduces the computational complexity by reducing the two-dimensional array of subresultant matrices to a one-dimensional problem.

7.4 Methods for the Computation of the Degree of the GCD

In Section 3.2 the degree of the GCD of two or three univariate polynomials was reduced to the determination of the last singular matrix in the subresultant matrix sequence. The method was extended to the computation of the degree of the GCD of two or three bivariate polynomials defined over a triangular domain, where the numerical rank of a sequence of subresultant matrices $S_k(\hat{f}(x,y),\hat{g}(x,y))$ for $k=1,\ldots,\min(m,n)$ was considered.

Bivariate polynomials defined over a triangular domain are defined in terms of their total degree, and the subresultant matrix sequences associated with the two-polynomial

and three-polynomial problem in this form are one-dimensional. However, bivariate polynomials over a rectangular domain are defined in terms of their relative degree structure with respect to x and y. By extension, the subresultant matrix sequences associated with the two-polynomial or three-polynomial GCD problems form two-dimensional arrays.

This section develops two methods for the computation of the degree of the GCD of two or three bivariate polynomials in Bernstein form. The methods are described for two polynomial problems and the extension to the three-polynomial problem is trivial.

The first method, BVGCD, is a simple extension of the univariate GCD (UGCD) method, while the second method, BVDRGCD, uses degree elevated polynomials in the sequence of subresultant matrices. The BVGCD method follows directly from earlier work and is considered first.

BVGCD

The set of subresultant matrices of two bivariate polynomials defined over a rectangular domain forms the two-dimensional array $\{S_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y)) \mid k_1 = 0,\ldots,\min(m_1,n_1);\ k_2 = 0,\ldots,\min(m_2,n_2)\}$. As with the univariate subresultant matrices, a measure of the rank of the (k_1,k_2)th subresultant matrix is denoted $\dot{\rho}_{k_1,k_2}$. For the remainder of this thesis only the minimum singular value of each subresultant matrix is considered. That is, $\dot{\rho}_{k_1,k_2} = \log_{10}(\dot{\sigma}_{k_1,k_2})$. Since $\{\,\dot{\rho}_{i_1,i_2} \mid i_1 = 0,\ldots,\min(m_1,n_1);\ i_2 = 0,\ldots,\min(m_2,n_2)\,\}$ is a two-dimensional array, the degree of the GCD is determined by the maximum change $\delta\dot{\rho}_{i_1,i_2}$, where

$$\delta\dot{\rho}_{i_1,i_2} = \dot{\rho}_{i_1+1,i_2+1} - \dot{\rho}_{i_1,i_2} \quad \text{for} \quad \begin{array}{l} i_1 = 0,\ldots,\min(m_1,n_1)-1; \\ i_2 = 0,\ldots,\min(m_2,n_2)-1. \end{array}$$

Therefore, the degree of the GCD is given by

$$(t_1,t_2) = \arg_{i_1,i_2}\max\{\delta\dot{\rho}_{i_1,i_2}\}.$$

This comes with the constraint that $\delta\dot{\rho}_{i_1,i_2}$ is only defined for $i_1 = 0,\ldots,\min(m_1,n_1)-1$ and $i_2 = 0,\ldots,\min(m_2,n_2)-1$, so the degree of the GCD is only determined when $0 \le t_1 \le (\min(m_1,n_1)-1)$ $0 \le t_2 \le (\min(m_2,n_2)-1)$.

Despite offering interesting insights, the computation of the degree of the GCD of two bivariate polynomials using the described method is computationally expensive. Constructing, preprocessing and analysing the singular values of $(\min(m_1,n_1)+1) \times (\min(m_2,n_2)+1)$ subresultant matrices is inefficient, and instead methods for reducing this to a one-dimensional problem should now be considered.

BVDRGCD

The BVDRGCD method reduces the computation of the degree structure (t_1,t_2) from a two-dimensional problem to a one-dimensional problem with two stages. The first stage determines a value t^* such that either t_1 or t_2 can be deduced. Then, given either t_1 or t_2, the second stage computes t_2 or t_1 from the set of subresultant matrices $\{\,S_{t_1,k_2} \mid k_2 = 0,\ldots,\min(m_2,n_2)\,\}$ or $\{\,S_{k_1,t_2} \mid k_1 = 0,\ldots,\min(m_1,n_1)\,\}$.

The first stage of the algorithm requires the degree elevation of the polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ and methods for degree elevation are found in [29]. The effect of degree elevation on the computation of the degree of the GCD of the two univariate polynomials $\hat{f}(x)$ and $\hat{g}(x)$ is considered first. This is then extended to the computation of the GCD of two bivariate polynomials which have been arbitrarily degree elevated, before finally considering the degree elevation necessary to reduce the computation of the degree of the GCD to a one-dimensional problem.

Example 7.4.1. Consider the univariate polynomials $\hat{f}(x)$ and $\hat{g}(x)$ of degrees m and n respectively, which have a GCD $\hat{d}_t(x)$ of degree t. Then

$$\hat{f}(x) = \hat{u}_t(x)\hat{d}_t(x) \quad \text{and} \quad \hat{g}(x) = \hat{v}_t(x)\hat{d}_t(x),$$

where $\hat{u}_t(x)$ and $\hat{v}_t(x)$ are of degrees $(m-t)$ and $(n-t)$ respectively. By elimination of $\hat{d}_t(x)$, the two equations can be written as

$$\hat{f}(x)\hat{v}_t(x) - \hat{g}(x)\hat{u}_t(x) = 0,$$

which can be written in matrix form as

$$S_t\left(\hat{f}(x), \hat{g}(x)\right)\mathbf{x}_t = 0. \tag{7.25}$$

This has a unique solution defined to within a scalar multiplier.

The pair of polynomials $\hat{f}(x)$ and $\hat{g}(x)$ are degree elevated to degrees m^* and n^* respectively, where $m^* = m + p$ and $n^* = n + p$. The degree elevated polynomial $\hat{f}^*(x)$ can be written as

$$\hat{f}^*(x) = \hat{u}^*_{t,i}(x)\hat{d}^*_{t,p-i}(x) \quad \text{for} \quad i = 0, \ldots, p$$

for a set of $(p+1)$ possible polynomial pairs $\hat{u}^*_{t,i}(x)$ and $\hat{d}^*_{t,p-i}(x)$ for $i = 0, \ldots, p$. The first subscript of these polynomials is indicative of the degree of the GCD and is consistent with notation used so far in this thesis. The second subscript is indicative of the number of degree elevations of the polynomial. For instance, the polynomials $\{\hat{u}^*_{t,i}(x)\}$ and $\{\hat{d}^*_{t,p-i}(x)\}$ are sets of degree elevated forms of $\hat{u}_t(x)$ and $\hat{d}_t(x)$ respectively, where

$$\begin{aligned} \deg\left(\hat{u}^*_{t,i}(x)\right) &= m - t + i \\ \deg\left(\hat{d}^*_{t,p-i}(x)\right) &= t + p - i \qquad \text{for} \quad i = 0, \ldots p. \end{aligned}$$

Therefore, the polynomial $\hat{f}^*(x)$ has a divisor $\hat{d}^*_{t,p}(x)$ of degree $(t+p)$, which is a degree elevated form of $\hat{d}_t(x)$. Similarly, the polynomial $\hat{g}^*(x)$ can be written as the product

$$\hat{g}^*(x) = \hat{v}^*_{t,i}(x)\hat{d}^*_{t,q-i}(x) \quad \text{for} \quad i = 0, \ldots, q.$$

The $(q+1)$ possible polynomial pairs $\hat{v}^*_{t,i}(x)$ and $\hat{d}^*_{t,q-i}(x)$ are degree elevated forms of

$\hat{v}_t(x)$ and $\hat{d}_t(x)$ respectively and

$$\begin{aligned}\deg\left(\hat{v}^*_{t,i}(x)\right) &= n-t+i \\ \deg\left(\hat{d}^*_{t,q-i}(x)\right) &= t+q-i \qquad \text{for} \quad i=0,\ldots q.\end{aligned}$$

So $\hat{g}^*(x)$ has a divisor $\hat{d}^*_{t,q}(x)$ of degree $(t+q)$, which is a degree elevated form of $\hat{d}_t(x)$. Therefore, polynomials $\hat{f}^*(x)$ and $\hat{g}^*(x)$ have a GCD $\hat{d}_{t,\min(p,q)}(x)$ of degree $(t+\min(p,q))$, which is a degree elevated form of $\hat{d}_t(x)$, and the modified form of (7.25) is given by

$$S_k(\hat{f}^*(x),\hat{g}^*(x))\mathbf{x}_k = 0,$$

which has non-trivial solutions for $k=1,\ldots,(t+\min(p,q))$ and has only zero solutions for $k=(t+\min(p,q)+1),\ldots,\min(m^*,n^*)$.

□

This is now extended to consider the effect of degree elevation on the degree of the GCD of the two arbitrarily degree elevated bivariate polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$, given that the degree of the GCD of the polynomials without degree elevation is known.

Example 7.4.2. Consider now the two bivariate polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ of degree (m_1,m_2) and (n_1,n_2) respectively, with a common divisor $\hat{d}_{t_1,t_2}(x,y)$ of degree (t_1,t_2). Suppose $\hat{f}(x,y)$ and $\hat{g}(x,y)$ are degree elevated by (p_1,p_2) and (q_1,q_2) respectively.

The polynomial $\hat{f}^*(x,y)$ can be written as the product

$$\hat{f}^*(x,y) = \hat{u}_{t_1,t_2,i_1,i_2}(x,y)\hat{d}_{t_1,t_2,p_1-i_1,p_2-i_2} \quad \text{for} \quad i_1=0,\ldots,p_1;\ i_2=0,\ldots,p_2,$$

where the polynomial $\hat{u}_{t_1,t_2,i_1,i_2}$ is a degree elevated form of $\hat{u}_{t_1,t_2}$ which has been degree elevated by (i_1,i_2), and the polynomial $\hat{d}_{t_1,t_2,p_1-i_1,p_2-i_2}$ is a degree elevated form of $\hat{d}_{t_1,t_2}$ which has been degree elevated by (p_1-i_1,p_2-i_2)

$$\begin{aligned}\deg\left(\hat{u}_{t_1,t_2,i_1,i_2}(x,y)\right) &= (m_1-t_1+i_1,m_2-t_2+i_2) \\ \deg\left(\hat{d}_{t_1,t_2,p_1-i_1,p_2-i_2}(x,y)\right) &= (t_1-p_1-i_1,t_2-p_2-i_2)\,.\end{aligned}$$

The polynomial $\hat{f}^*(x,y)$ has a divisor of degree (t_1+p_1,t_2+p_2), which is the degree elevated form of $\hat{d}_{t_1,t_2}(x,y)$. Similarly,

$$\hat{g}^*(x,y) = \hat{v}_{t_1,t_2,i_1,i_2}(x,y)\hat{d}_{t_1,t_2,q_1-i_1,q_2-i_2}(x,y) \quad \text{for} \quad i_1=0,\ldots,q_1;\ i_2=0,\ldots,q_2.$$

The GCD of $\hat{f}(x,y)$ and $\hat{g}(x,y)$ is therefore given by $\hat{d}_{t_1,t_2,\min(p_1,q_1)\,\min(p_2,q_2)}(x,y)$ of degree $(t_1+\min(p_1,q_1),t_2+\min(p_2,q_2))$, which is a degree elevated form of $\hat{d}_{t_1,t_2}(x,y)$. The third and fourth subscripts denote the number of degree elevations of the original $\hat{d}_{t_1,t_2}(x,y)$, and

$$S_{k_1,k_2}\left(\hat{f}^*(x,y),\hat{g}^*(x,y)\right)\mathbf{x}_{k_1,k_2} = 0$$

has non-trivial solutions for $k_1=0,\ldots,(t_1+\min(p_1,q_1))\ ;\ k_2=0,\ldots,(t_2+\min(p_2,q_2))$

and only trivial solutions for $k_1 = (t_1 + \min(p_1, q_1) + 1), \ldots, (\min(m^*, n^*))$; $k_2 = (t_2 + \min(p_2, q_2) + 1), \ldots, (\min(m^*, n^*))$.

□

Suppose now that $\hat{f}(x, y)$ and $\hat{g}^*(x, y)$ are degree elevated such that $\hat{f}^*(x, y)$ is of degree (m^*, m^*) and $\hat{g}^*(x, y)$ is of degree (n^*, n^*), where $m^* = \max(m_1, m_2)$ and $n^* = \max(n_1, n_2)$.

Consider now the bivariate subresultant matrix $S_{k,k}(\hat{f}^*(x, y), \hat{g}^*(x, y))$. If $S_{k,k}(\hat{f}^*(x, y), \hat{g}^*(x, y))$ is rank deficient, then the following conditions hold:

$$k \leq t_1 + \min(p_1, q_1) \qquad \text{and} \qquad k \leq t_2 + \min(p_2, q_2). \tag{7.26}$$

Suppose t is defined such that $S_{t,t}(\hat{f}^*(x, y), \hat{g}^*(x, y))$ is rank deficient but $S_{t+1,t+1}(\hat{f}^*(x, y), \hat{g}^*(x, y))$ is of full rank, then one of the conditions given above in (7.26) no longer holds. Therefore, one or both of the following hold:

$$t = t_1 + \min(p_1, q_1) \tag{7.27}$$

$$\text{or} \qquad t = t_2 + \min(p_2, q_2). \tag{7.28}$$

Three possible scenarios must now be considered:

1. The first equation (7.27) holds and t_1 is therefore given by

   $$t_1 = t - \min(p_1, q_1)$$

   and t_2 must be determined. The set of equations

   $$S_{t_1,k_2}\left(\hat{f}(x, y), \hat{g}(x, y)\right) \mathbf{x}_{t_1,k_2} = \mathbf{0}$$

   have non-trivial solutions for $k_2 = 0, \ldots, t_2$ and only trivial solutions for $k_2 = t_2 + 1, \ldots, \min(m_2, n_2)$, and t_2 can therefore be determined by analysis of the numerical rank of the set of subresultant matrices $\{S_{t_1,k_2}(\hat{f}(x, y), \hat{g}(x, y)) \mid k_2 = 0, \ldots, \min(m_2, n_2)\}$.

2. The second equation (7.28) holds, and t_2 is therefore given by

   $$t_2 = t - \min(p_2, q_2).$$

   Then t_1 must be determined by analysis of the set of subresultant matrices

   $$S_{k_1,t_2}\left(\hat{f}(x, y), \hat{g}(x, y)\right) \quad k_1 = 1, \ldots, \min(m_1, n_1).$$

3. Both equations hold, in which case

   $$t_1 = t - \min(p_1, q_1) \qquad \text{and} \qquad t_2 = t - \min(p_2, q_2).$$

Given t, it is not known which of the two equations (7.27, 7.28) holds, so both sets of sub-

resultant matrices $\{S_{t_1,k_2}(\hat{f}(x,y),\hat{g}(x,y))\}$ and $\{S_{k_1,t_2}(\hat{f}(x,y),\hat{g}(x,y))\}$ are constructed and two possible candidate pairs (t_1,t_2) are computed. The degree of the GCD is the maximum of these two pairs.

The following worked example aims to highlight the differences between the BVGCD and BVDRGCD methods described in this section.

Example 7.4.3. Consider the polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ of degrees $(m_1,m_2) = (16,12)$ and $(n_1,n_2) = (14,10)$ respectively

$$\begin{aligned}\hat{f}(x,y) &= (x+y+0.0124)^5(x+0.56)^4(x^2+y^2+0.51)^2(x+y+1.12)^3\\ \hat{g}(x,y) &= (x+y+0.4512)^3(x+0.56)^4(x^2+y^2+0.51)^2(x+y+1.12)^3,\end{aligned}$$

whose GCD in factorised form is given by

$$\hat{d}(x,y) = (x+0.56)^4(x^2+y^2+0.51)^2(x+y+1.12)^3.$$

Noise is added to the coefficients of $\hat{f}(x,y)$ and $\hat{g}(x,y)$ such that the coefficients of the inexact polynomials $f(x,y)$ and $g(x,y)$ are given by

$$\begin{aligned}a_{i_1,i_2} &= \hat{a}_{i_1,i_2} + r_{f,i_1,i_2}\hat{a}_{i_1,i_2}\epsilon_{f,i_1,i_2} \quad \text{for} \quad i_1 = 0,\ldots,m_1;\ i_2 = 0,\ldots,m_2,\\ b_{i_1,i_2} &= \hat{b}_{i_1,i_2} + r_{g,j_1,j_2}\hat{b}_{i_1,i_2}\epsilon_{g,i_1,i_2} \quad \text{for} \quad j_1 = 0,\ldots,n_1;\ j_2 = 0,\ldots,n_2,\end{aligned} \tag{7.29}$$

where $\{\epsilon_{f,i_1,i_2}\} = \{\epsilon_{g,j_1,j_2}\} = 10^{-8}$.

The bivariate GCD (BVGCD) algorithm proceeds by preprocessing the polynomials $f(x,y)$ and $g(x,y)$ for each subresultant matrix $\{S_{k_1,k_2} \mid k_1 = 0,\ldots,\min(m_1,n_1) = 14;\ k_2 = 0,\ldots,\min(m_2,n_2) = 10\}$ and computing the minimum singular value of each subresultant matrix $\{S_{k_1,k_2}(\tilde{f}_{k_1,k_2}(\omega_1,\omega_2),\alpha_{k_1,k_2}\tilde{g}_{k_1,k_2}(\omega_1,\omega_2))\}$. The two-dimensional set of minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ are plotted in Figure 7.5 from which the degree of the AGCD is correctly identified as $(11,7)$, since the maximum of the set $\{\delta\dot{\rho}_{i_1,i_2}\}$ is given by $\delta\dot{\rho}_{11,7}$.

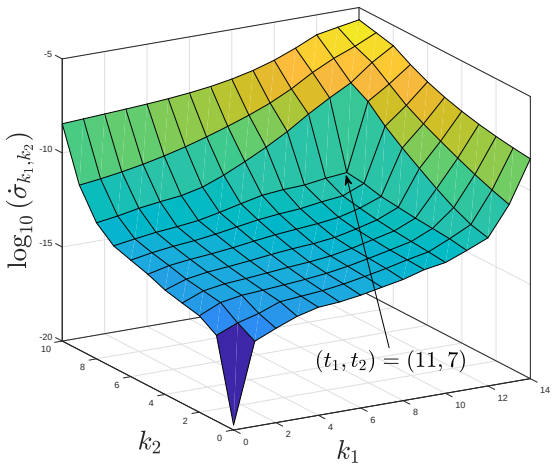


Figure 7.5: The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the preprocessed subresultant matrices $\{S_{k_1,k_2}(\lambda_{k_1,k_2}\tilde{f}_{k_1,k_2}(\omega_1,\omega_2),\tilde{g}_{k_1,k_2}(\omega_1,\omega_2))\}$ in Example 7.4.3

The BVDRGCD method is now considered. In the first stage of the BVDRGCD

method, the polynomials $f(x,y)$ and $g(x,y)$ are degree elevated to (m^*,m^*) and (n^*,n^*) respectively. That is, $f(x,y)$ and $g(x,y)$ are degree elevated by $(p_1,p_2)=(0,4)$ and $(q_1,q_2)=(0,4)$ respectively.

The minimum singular values of the set of subresultant matrices

$$S_{k^*,k^*}(\lambda_{k^*,k^*}\tilde{f}^*_{k,k}(\omega_1,\omega_2),\tilde{g}^*_{k,k}(\omega_1,\omega_2)) \quad \text{for} \quad k=1,\ldots,\min(m^*,n^*) \tag{7.30}$$

are computed and plotted in Figure 7.6i, from which, $t=11$. Given $t=11$, either

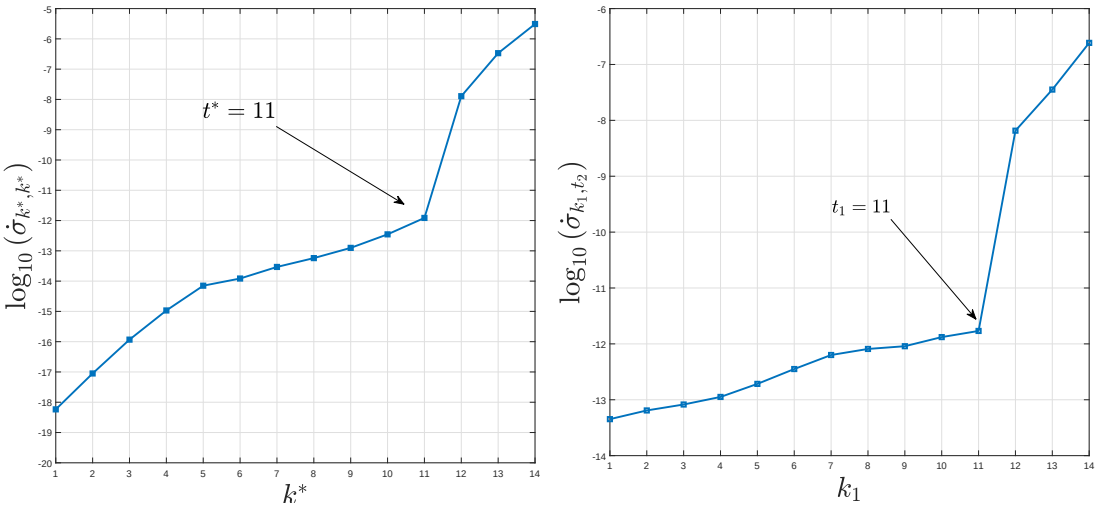
$$\begin{aligned} & t_1 = t-\min(p_1,q_1) = 11 \\ \text{or} \quad & t_2 = t-\min(p_2,q_2) = 11-4 = 7. \end{aligned}$$

In this instance, both t_1 and t_2 are correctly identified. However, assume that only t_2 is correctly determined, then the degree of the AGCD with respect to x, that is t_1, is computed from the minimum singular values of the set of subresultant matrices

$$S_{k_1,t_2}\left(\lambda_{k_1,t_2}\tilde{f}_{k_1,t_2}(\omega_1,\omega_2),\tilde{g}_{k_1,t_2}(\omega_1,\omega_2)\right) \quad \text{for} \quad k_1=1,\ldots,\min(m_1,n_1),$$

which are plotted in Figure 7.6ii. From the set of minimum singular values $\{\dot{\sigma}_{k_1,t_2} \mid k_1=1,\ldots,\min(m_1,n_1)\}$, the degree of the AGCD with respect to x is determined to be equal to $t_1=11$.

In this example, the bivariate GCD (BVGCD) algorithm required the evaluation of 140 subresultant matrices, whereas the bivariate dimension reducing GCD (BVDRGCD) method required the preprocessing and evaluation of only 28 subresultant matrices.



(i) BVDRGCD Stage 1 : The minimum singular values $\{\dot{\sigma}_{k,k}\}$ of $\{S_{k,k}(\lambda_{k,k}\tilde{f}_{k,k}(\omega_1,\omega_2),\tilde{g}_{k,k}(\omega_1,\omega_2))\}$

(ii) BVDRGCD Stage 2a : The minimum singular values $\{\dot{\sigma}_{k_1,t_1}\}$ of $\{S_{k_1,t_2}(\lambda_{k_1,t_2}\tilde{f}_{k_1,t_2}(\omega_1,\omega_2),\tilde{g}_{k_1,t_2}(\omega_1,\omega_2))\}$

Figure 7.6: The minimum singular values of the preprocessed subresultant matrices in the BVDRGCD algorithm in Example 7.4.3

□

In this section, a method (BVDRGCD) has been developed which allows for the fast computation of the degree of the GCD of two bivariate polynomials in Bernstein form

which are defined over a rectangular domain.

Other methods were considered for dimension reduction. One proposed method proceeds as follows :

1. Fix k_1 to an arbitrary value k_1^* which is known to be smaller than t_1, the degree of the GCD with respect to x.

2. Determine the numerical rank of all subresultant matrices $\{ S_{k_1^*,k_2}(\hat{f}(x,y), \hat{g}(x,y)) \mid k_2 = 0, \ldots, \min(m_2, n_2) \}$ and t_2 is given by the index k_2 of the last subresultant matrix S_{t_1,k_2} which is numerically rank deficient.

3. Having determined t_2, compute t_1 from the set of subresultant matrices $\{S_{k_1,t_2}(\hat{f}(x,y), \hat{g}(x,y)) \mid k_1 = 1, \ldots, \min(m_1, n_1)\}$.

Though this method is theoretically correct, since $S_{k_1,k_2}(\hat{f}(x,y), \hat{g}(x,y))$ is rank deficient for all $k_1 = 0, \ldots, \min(m_1, n_1)$; $k_2 = 0, \ldots, \min(m_2, n_2)$, numerical results suggest it should not be utilised. Not all choices of $k_1^* < t_1$ are 'good' choices. Consider Example 7.4.3 and the minimum singular values plotted in Figure 7.5. A choice of k_1^* from the interval $[0, 6]$ fails to reveal the degree of the GCD with respect to y, since there is no maximal change in the minimum singular values of $\{S_{k_1^*,k_2}(\hat{f}(x,y), \hat{g}(x,y)) \mid k_2 = 0, \ldots, \min(m, n)\}$.

In this method, only a subset of possible (k_1, k_2) pairs are considered, however the BVDRGCD method considers all possible (k_1, k_2) pairs in degree elevated forms.

Another similarly proposed method proceeds as follows :

1. Compute the minimum singular values of the set of subresultant matrices $\{S_{k,k}(\hat{f}(x), \hat{g}(x)) \mid \ k = 1, \ldots, \min(\min(m_1, n_1), \min(m_2, n_2))\}$.

2. Determine the value t given by the index of the last rank deficient subresultant matrix.

3. Since $S_{t,t}(\hat{f}(x,y), \hat{g}(x,y))$ is rank deficient and $S_{t+1,t+1}(\hat{f}(x,y), \hat{g}(x,y))$ is of full rank, then either $t_1 = t$ or $t_2 = t$.

4. Suppose that t_1 is correctly identified, then t_2 is given by the index k_2 of the last numerically singular matrix in the set $\{S_{t_1,k_2} \mid k_2 = 0, \ldots, \min(m_2, n_2)\}$ to give the candidate pair (t_1, t_2).

5. Suppose that t_2 is correctly identified, then t_1 is given by the index k_1 of the last numerically singular matrix in the set $\{S_{k_1,t_2} \mid k_1 = 0, \ldots, \min(m_1, n_1)\}$ to give the candidate pair (t_1, t_2).

6. Given the two candidate pairs, determine the degree of the GCD.

However, in Example 7.4.3 the set of minimum singular values of all subresultant matrices S_{k_1,k_2} are plotted in Figure 7.5, from which it can be seen that the maximum change in singular values between $\dot{\sigma}_{11,7}$ and $\dot{\sigma}_{12,8}$, that is, $\delta\dot{\sigma}_{11,7}$, is maximal in the set $\{\delta\dot{\sigma}_{k_1,k_2}\}$. This result is not included when only considering the subresultant matrices $S_{k,k}(\hat{f}(x,y), \hat{g}(x,y))$ and only $\{\delta\dot{\sigma}_{i,i} \mid i = 1, \ldots, \min(m_1, n_1, m_2, n_2)\}$ are considered.

Suppose $m_1 \approx m_2 \approx m$ and $n_1 \approx n_2 \approx n$, then the standard method BVGCD requires the construction of a two-dimensional array of n^2 subresultant matrices, but BVDRGCD reduces this to approximately $3n$ subresultant matrices.

Only the two-polynomial GCD problem has been considered in this section. The necessary theoretical development to extend the BVDRGCD method for the computation of the degree of the GCD of three-polynomial problem is a trivial extension. Examples are given in (7.6.2).

The next section considers the computation of the coefficients of the cofactor polynomials given t_1 and t_2, and this least squares based method is independent of whether BVGCD or BVDRGCD is used in the computation of the degree of the GCD. The coefficients are always computed using the (t_1, t_2)th subresultant matrix. More results comparing BVGCD and BVDRGCD are given in Section 7.6.

7.5 Approximating the Coefficients of the Cofactor Polynomials and the GCD

The Two-Polynomial Problem

This section considers the approximation of the coefficients of the GCD triple $(\hat{u}_{t_1,t_2}, \hat{v}_{t_1,t_2}, \hat{d}_{t_1,t_2})$. Approximations can be computed from the (t_1, t_2)th subresultant matrix of (i) the unprocessed polynomials $f(x, y)$ and $g(x, y)$ or (ii) the preprocessed polynomials $\lambda_{t_1,t_2}\tilde{f}_{t_1,t_2}(\omega_1, \omega_2)$ and $\tilde{g}(\omega_1, \omega_2)$. The respective approximations are denoted $(u_{t_1,t_2}(x, y), v_{t_1,t_2}(x, y), d_{t_1,t_2}(x, y))$ and $(\tilde{u}_{t_1,t_2}(x, y), \tilde{v}_{t_1,t_2}(x, y), \tilde{d}_{t_1,t_2}(x, y))$.

Given that the degree of the GCD has been computed and is given by the pair (t_1, t_2), then

$$S_{t_1,t_2}(f, g)\mathbf{x}_{t_1,t_2} \approx \mathbf{0}$$

has a non-trivial solution and a column $\mathbf{c}_{t_1,t_2,q}$ of $S_{t_1,t_2}(f, g)$ lies in the space spanned by the remaining columns $A_{t_1,t_2,q}$ such that

$$A_{t_1,t_2,q}(f, g)\mathbf{x}_{t_1,t_2,q} \approx \mathbf{c}_{t_1,t_2,q}.$$

The vector $\mathbf{x}_{t_1,t_2}$ is given by the insertion of '-1' into the qth position of $\mathbf{x}_{t_1,t_2,q}$.

The coefficients of the approximations of the polynomials $\hat{u}_{t_1,t_2}(x, y)$ and $\hat{v}_{t_1,t_2}(x, y)$ are given by the vector $\bar{\mathbf{x}}_{t_1,t_2}$, where the first $(n_1 - t_1 + 1) \times (n_2 - t_2 + 1)$ entries are coefficients of the approximation of $\hat{v}_{t_1,t_2}(x, y)$ and the remaining $(m_1 - t_1 + 1) \times (m_2 - t_2 + 1)$ entries are the coefficients of the approximation of $\hat{u}_{t_1,t_2}(x, y)$.

The Three-Polynomial Problem

The (2×3) and (3×3) subresultant matrices $\hat{S}_{k_1,k_2}$ and $\tilde{S}_{k_1,k_2}$ are numerically singular for $k_1 = 0, \ldots, \min(m_1, n_1, o_1)$; $k_2 = 0, \ldots, \min(m_2, n_2, o_2)$ so

$$\tilde{S}_{k_1,k_2}(f, g, h)\,\mathbf{x}_{k_1,k_2} \approx 0 \qquad \text{and} \qquad \hat{S}_{k_1,k_2}(f, g, h)\,\mathbf{x}_{k_1,k_2} \approx 0 \tag{7.31}$$

have non-trivial solutions. A column of $\tilde{S}_{t_1,t_2}(f,g,h)$ and $\hat{S}_{t_1,t_2}(f,g,h)$ lies in the space spanned by the remaining columns with minimal residual, so

$$\hat{A}_{t_1,t_2}(f,g,h)\,\hat{\mathbf{x}}_{t_1,t_2,q} \approx \hat{\mathbf{c}}_{t_1,t_2,q} \qquad \text{and} \qquad \tilde{A}_{t_1,t_2}(f,g,h)\,\tilde{\mathbf{x}}_{t_1,t_2,q} \approx \tilde{\mathbf{c}}_{t_1,t_2,q}.$$

The vectors $\tilde{\mathbf{x}}_{t_1,t_2}$ and $\hat{\mathbf{x}}_{t_1,t_2}$ in (7.31) are given by the insertion of '-1' into the vectors $\tilde{\mathbf{x}}_{t_1,t_2,q}$ and $\hat{\mathbf{x}}_{t_1,t_2,q}$, and contain approximations of the coefficients of the polynomials $\hat{u}_{t_1,t_2}(x,y)$, $\hat{v}_{t_1,t_2}(x,y)$ and $\hat{w}_{t_1,t_2}(x,y)$.

Methods for computing the structured low rank approximation of the (t_1,t_2)th subresultant matrix could be used to compute more accurate approximations of the cofactor polynomials and the GCD. However, this theoretically small extension would require a significant amount of work to implement, and this chapter focuses more on methods for the computation of the degree of the GCD rather than its coefficients.

7.6 Results

This section presents results of the two developed methods in this chapter. Examples will consider the computation of the GCD of two bivariate polynomials using the standard BVGCD method, where the minimum singular values of unprocessed and preprocessed subresultant matrices are compared. The following examples also consider the computation of the degree of the GCD using the bivariate dimension reducing GCD (BVDRGCD) method.

7.6.1 Examples of the Two-Polynomial Problem

Example 7.6.1. Consider the Bernstein form of the exact polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ of degrees $(17,10)$ and $(17,11)$ respectively, whose factorisations are given by

$$\begin{aligned}\hat{f}(x,y) &= (x-1.39872512)^3(x-0.5354788154)^2(x-0.44455421)^{10}\times\\ &\qquad (x+0.268721020)^2(y-0.96543321)^6(y+5.45492341)^4\\ \hat{g}(x,y) &= (x-1.39872512)^3(x-0.5354788154)^2(x-0.155224776)^{10}\\ &\qquad (x+0.268721020)^2(y-0.96543321)^6(y-0.22341321)^5.\end{aligned}$$

The polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ have a GCD of degree $(7,6)$ which is given by

$$\hat{d}(x,y) = (x-1.39872512)^3(x-0.5354788154)^2(x+0.268721020)^2(y-0.96543321)^6.$$

Noise is added to the coefficients of $\hat{f}(x,y)$ and $\hat{g}(x,y)$ such that the coefficients of the inexact polynomials $f(x,y)$ and $g(x,y)$ are given by

$$a_{i_1,i_2} = \hat{a}_{i_1,i_2} + r_{f,i_1,i_2}\hat{a}_{i_1,i_2}\epsilon_{f,i_1,i_2} \quad \text{and} \quad b_{j_1,j_2} = \hat{b}_{j_1,j_2} + r_{g,j_1,j_2}\hat{b}_{j_1,j_2}\epsilon_{g,j_1,j_2}, \tag{7.32}$$

where $\{\, r_{f,i_1,i_2} \mid i_1 = 0,\ldots,m_1;\ i_2 = 0,\ldots,m_2 \,\}$ and $\{\, r_{g,j_1,j_2} \mid j_1 = 0,\ldots,n_1;\ j_2 = 0,\ldots,n_2 \,\}$ are uniformly distributed random variables in the interval $[-1,1]$, while $\{\, \epsilon_{f,i_1,i_2} \mid i_1 = 0,\ldots,m_1;\ i_2 = 0,\ldots,m_2 \,\}$ and $\{\, \epsilon_{g,j_1,j_2} \mid j_1 = 0,\ldots,n_1;\ j_2 = 0,\ldots,n_2 \,\}$ are uniformly distributed random variables in the interval $\left[10^{-12},10^{-10}\right]$.

The inexact polynomials $f(x,y)$ and $g(x,y)$ are preprocessed, and the coefficients of the (i) unprocessed and (ii) preprocessed forms are plotted in Figure 7.7. The coefficients of $f(x,y)$ and $g(x,y)$ span approximately 20 orders of magnitude, while the coefficients of the preprocessed polynomials $\lambda_{1,1}\tilde{f}_{1,1}(\omega_1,\omega_2)$ and $\tilde{g}_{1,1}(\omega_1,\omega_2)$ span approximately 10 orders of magnitude.

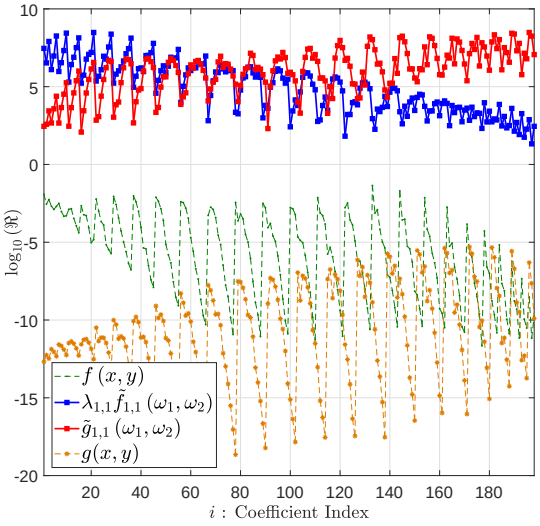


Figure 7.7: The coefficients of the unprocessed polynomials $f(x,y)$ and $g(x,y)$ and the preprocessed polynomials $\lambda_{1,1}\tilde{f}_{1,1}(\omega_1,\omega_2)$ and $\tilde{g}_{1,1}(\omega_1,\omega_2)$ in Example 7.6.1

Using the BVGCD method, the SVD of each of the (i) unprocessed or (ii) preprocessed subresultant matrices are computed, and the minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the unprocessed and preprocessed subresultant matrices are plotted in Figure 7.8i and Figure 7.8ii respectively.

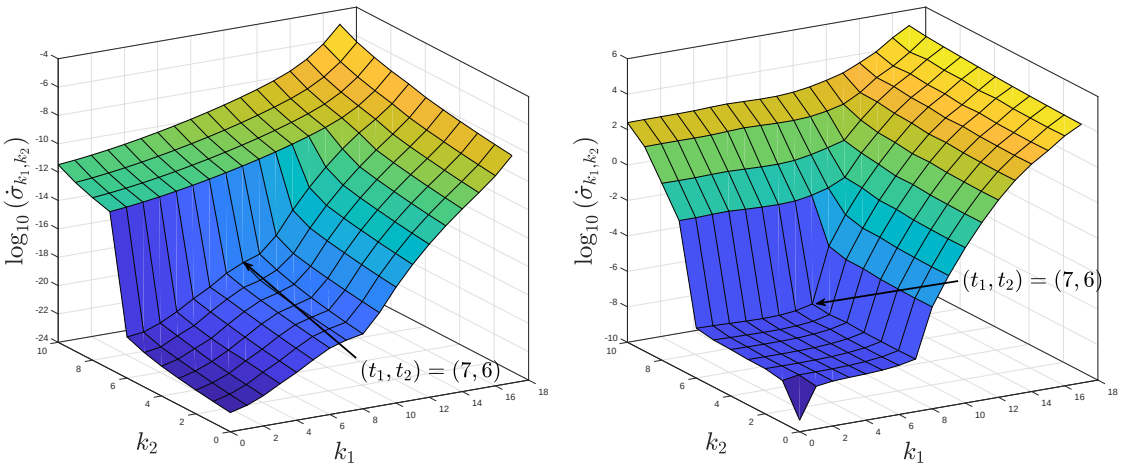
The degree of the AGCD cannot be determined from the minimum singular values of the subresultant matrices of unprocessed polynomials $f(x,y)$ and $g(x,y)$ (Figure 7.8i). However, by observation, it is clear that the degree of the AGCD with respect to y is given by $t_2 = 6$, but the degree with respect to x in unknown.

The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the subresultant matrices $\{S_{k_1,k_2}(\lambda_{k_1,k_2}\tilde{f}_{k_1,k_2}(\omega_1,\omega_2),\tilde{g}_{k_1,k_2}(\omega_1,\omega_2))\}$ of the preprocessed polynomials $\lambda_{k_1,k_2}\tilde{f}_{k_1,k_2}(\omega_1,\omega_2)$ and $\tilde{g}_{k_1,k_2}(\omega_1,\omega_2)$ are plotted in Figure 7.8ii. From this, it can be seen that the of the AGCD is correctly identified as $(t_1,t_2) = (7,6)$.

The BVDRGCD method is now considered. The polynomials $f(x,y)$ and $g(x,y)$ are degree elevated by $(p_1,p_2) = (0,7)$ and $(q_1,q_2) = (0,6)$ respectively to obtain the degree elevated polynomials $f^*(x,y)$ and $g^*(x,y)$ of degrees $m^* = 17$ and $n^* = 13$ respectively. The minimum singular values $\{\dot{\sigma}_{k,k} \mid k = 1,\ldots,17\}$ of the subresultant matrices $\{S_{k\cdot k}(\lambda_{k,k}\tilde{f}^*_{k,k}(\omega_1,\omega_2),\tilde{g}^*_{k,k}(\omega_1,\omega_2))\}$ are plotted in Figure 7.9, from which the degree of the AGCD is given by $t = 7$. Given t, either

$$t_1 = t - \min(p_1,q_1) = 7 \qquad \text{or} \qquad t_2 = t - \min(p_2,q_2) = 7 - 6 = 1.$$

The second stage of the BVDRGCD requires the computation of the sets of the minimum singular values of the preprocessed subresultant matrices $\{S_{t_1,k_2}(\lambda_{t_1,k_2}\tilde{f}_{t_1,k_2}(\omega_1,\omega_2),\tilde{g}_{t_1,k_2}(\omega_1,\omega_2)) \mid k_2 = 0,\ldots,\min(m_2,n_2)\}$ and the pre-



(i) The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the unprocessed subresultant matrices $\{S_{k_1,k_2}(f(x,y),g(x,y))\}$

(ii) The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the preprocessed subresultant matrices $\{S_{k_1,k_2}(\lambda_{k_1,k_2}\tilde{f}_{k_1,k_2}(\omega_1,\omega_2),\tilde{g}_{k_1,k_2}(\omega_1,\omega_2))\}$

Figure 7.8: The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 7.6.1

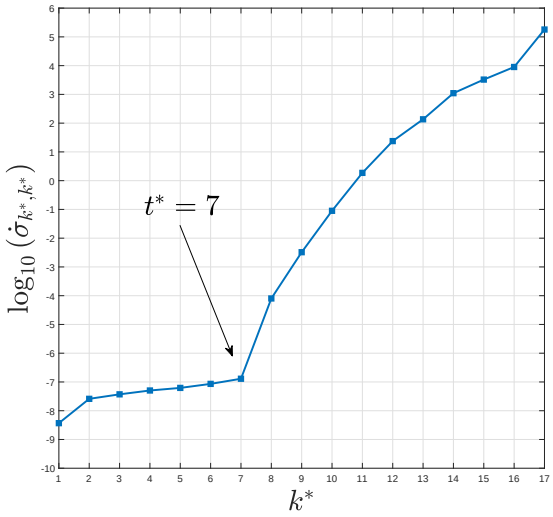


Figure 7.9: BVDRGCD Stage 1 :
The minimum singular values of the preprocessed subresultant matrices $\{S_{k,k}(\lambda_{k,k}\tilde{f}^*_{k,k}(\omega_1,\omega_2),\tilde{g}^*_{k,k}(\omega_1,\omega_2))\}$ in Example 7.6.1

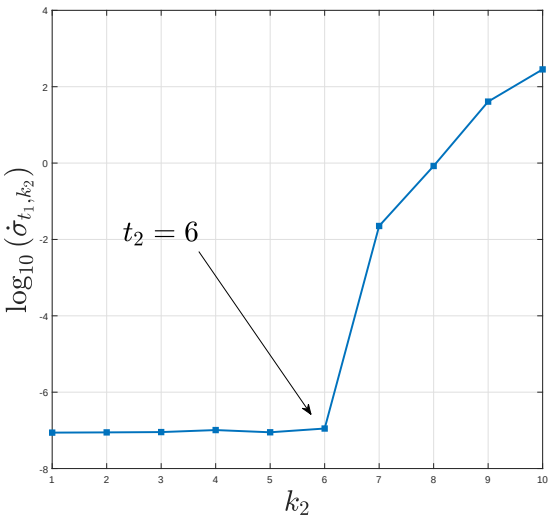
processed subresultant matrices $\{S_{k_1,t_2}(\lambda_{k_1,t_2}\tilde{f}_{k_1,t_2}(\omega_1,\omega_2),\tilde{g}_{k_1,t_2}(\omega_1,\omega_2)) \mid k_1 = 0,\ldots,\min(m_1,n_1)\}$, which are plotted in Figure 7.10i and Figure 7.10ii respectively.

It can be seen that if the candidate $t_1 = 7$ is correct, then t_2 is given by the index of the last numerically rank deficient subresultant matrix in the set $\{S_{t_1,k_2}(\lambda_{t_1,k_2}\tilde{f}_{t_1,k_2}(\omega_1,\omega_2),\tilde{g}_{t_1,k_2}(\omega_1,\omega_2)) \mid k_2 = 0,\ldots,\min(m_2,n_2)\}$. The minimum singular values of these subresultant matrices are given by $\{\dot{\sigma}_{t_1,k_2} \mid k_2 = 1,\ldots,\min(m_2,n_2)\}$ and these values are plotted in Figure 7.10i. The computed degree of the AGCD with respect to x is given by $t_2 = 6$ The candidate pair for the degree of the AGCD is therefore given by $(t_1,t_2) = (7,6)$.

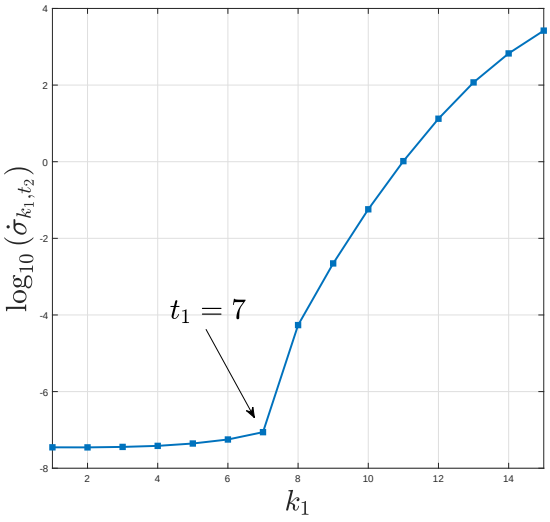
Alternatively, if the candidate $t_2 = 1$ is correct, then t_1 is given by the index of the last numerically rank deficient subresultant matrix in the set $\{S_{k_1,t_2}(\lambda_{k_1,t_2}\tilde{f}_{k_1,t_2}(\omega_1,\omega_2),\tilde{g}_{k_1,t_2}(\omega_1,\omega_2)) \mid k_1 = 0,\ldots,\min(m_1,n_1)\}$. The minimum singu-

lar values of the set of subresultant matrices are denoted $\{\dot{\sigma}_{k_1,t_2} \mid k_1 = 1, \ldots, \min(m_1, n_1)\}$ and are plotted in Figure 7.10ii. The computed degree of the AGCD with respect to x is given by $t_1 = 7$ and the candidate pair is therefore given by $(t_1, t_2) = (7, 1)$.

Since the candidate pair $(7, 6)$ is greater than $(7, 1)$, the degree of the AGCD is given by $(t_1, t_2) = (7, 6)$.



(i) BVDRGCD Stage 2a : The minimum singular values of the preprocessed subresultant matrices $\{S_{t_1,k_2} \mid k_2 = 1, \ldots, \min(m_2, n_2)\}$



(ii) BVDRGCD Stage 2b : The minimum singular values of the preprocessed subresultant matrices $\{S_{k_1,t_2} \mid k_1 = 1, \ldots, \min(m_1, n_1)\}$

Figure 7.10: BVDRGCD Stage 2 : The minimum singular values of the preprocessed subresultant matrices.

In this example the BVGCD method required the evaluation of[(i)]180 subresultant matrices while BVDRGCD required the evaluation of only 32 subresultant matrices.

□

Example 7.6.2. Consider the Bernstein form of the exact polynomials $\hat{f}(x, y)$ and $\hat{g}(x, y)$ of degrees $(13, 12)$ and $(11, 10)$ respectively, whose factorised forms are given by

$$\hat{f}(x, y) = (x + 0.56)(x + y + 0.0124)^5(x + y + 1.12)^3(x^2 + y^2 + 0.51)^2$$
$$\hat{g}(x, y) = (x + 0.56)(x + y + 0.4512)^3(x + y + 1.12)^3(x^2 + y^2 + 0.51)^2.$$

The GCD of $\hat{f}(x, y)$ and $\hat{g}(x, y)$, of degree $(t_1, t_2) = (8, 7)$, is given by

$$\hat{d}_t(x, y) = (x + 0.56)(x^2 + y^2 + 0.51)^2(x + y + 1.12)^3.$$

Noise is added to the coefficients of $\hat{f}(x, y)$ and $\hat{g}(x, y)$ and the coefficients of inexact polynomials $f(x, y)$ and $g(x, y)$ are given by (7.32), where $\{r_{f,i_1,i_2}\}$ and $\{r_{g,j_1,j_2}\}$ are uniformly distributed random variables in the interval $[-1, 1]$ and $\{\epsilon_{f,i_1,i_2}\}$ and $\{\epsilon_{g,j_1,j_2}\}$ are uniformly distributed random variables in the interval $[10^{-10}, 10^{-8}]$.

[(i)] The term "evaluation of" is used here to mean "construction of, preprocessing of and singular value decomposition of".

From Figure 7.11 it can be seen that the coefficients of $f(x,y)$ and $g(x,y)$ span approximately 14 orders of magnitude, while the coefficients of the preprocessed polynomials $\lambda_{1,1}\tilde{f}_{1,1}(\omega_1,\omega_2)$ and $\tilde{g}(\omega_1,\omega_2)$ span less than 6 orders of magnitude.

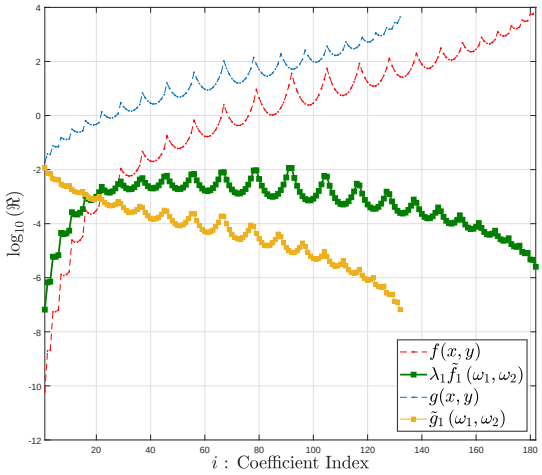


Figure 7.11: The coefficients of both the unprocessed polynomials $f(x,y)$ and $g(x,y)$ and the preprocessed polynomials $\lambda_{1,1}\tilde{f}_{1,1}(\omega_1,\omega_2)$ and $\tilde{g}_{1,1}(\omega_1,\omega_2)$ in Example 7.6.2

This example begins by considering the BVGCD method for the computation of the degree of the GCD of two bivariate polynomials. The BVGCD algorithm proceeds by computing the minimum singular values $\{\dot{\sigma}_{k_1,k_2} \mid k_1 = 0,\ldots,\min(m_1,n_1), k_2 = 0,\ldots,\min(m_2,n_2)\}$ of either the set of unprocessed or preprocessed subresultant matrices.

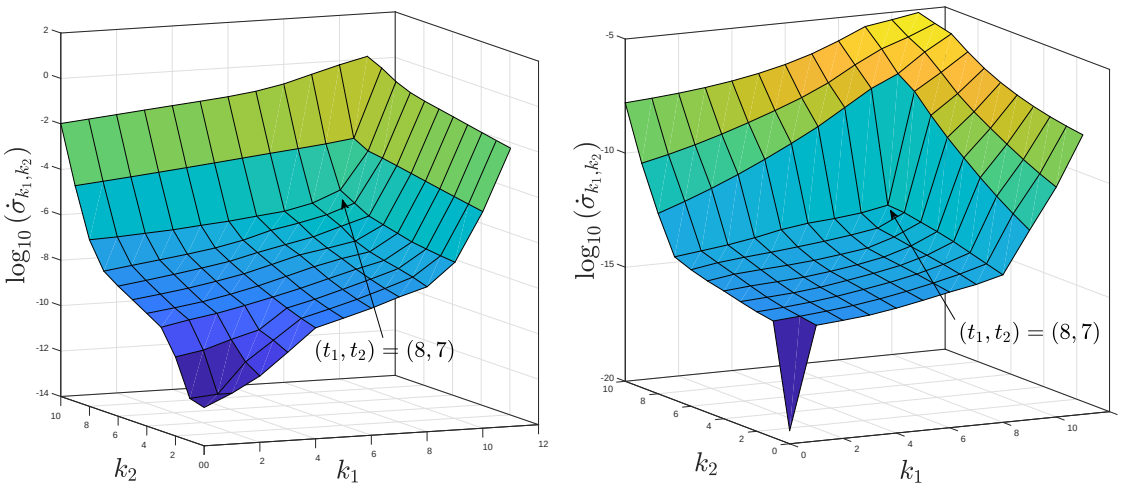
In Figure 7.12 the minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of both the (i) unprocessed and (ii) preprocessed subresultant matrices are plotted. In Figure 7.12i there is no distinct value $\dot{\rho}_{k_1,k_2} = \log_{10}(\dot{\sigma}_{k_1,k_2})$ such that $\delta\dot{\rho}_{k_1,k_2}$ is significantly larger than any other $\{\delta\dot{\rho}_{i,j}\}$. Therefore the degree of the is not correctly identified. However, in Figure 7.12ii there is a clear separation between the numerically zero and non-zero minimum singular values of the preprocessed subresultant matrices $\{S_{k_1,k_2}(\lambda_{k_1,k_2}\tilde{f}_{k_1,k_2}(\omega_1,\omega_2),\tilde{g}_{k_1,k_2}(\omega_1,\omega_2))\}$, and $\delta\dot{\sigma}_{8,7}$ is the the maximum entry of the set $\{\delta\dot{\sigma}_{i,j}\}$, that is, the maximum change in the minimum singular values occurs between $\dot{\sigma}_{8,7}$ and $\dot{\sigma}_{9,8}$. The degree of the AGCD is therefore correctly determined as $(t_1,t_2) = (8,7)$.

The BVDRGCD method is now considered. The polynomials $f(x,y)$ and $g(x,y)$ are degree elevated to $f^*(x,y)$ and $g^*(x,y)$ of degree $m^* = 13$ and $n^* = 11$ respectively, where the number of degree elevations of $f(x,y)$ and $g(x,y)$ are given by $(p_1,p_2) = (0,1)$ and $(q_1,q_2) = (0,1)$ respectively.

The minimum singular values $\{\dot{\sigma}_{k,k} \mid k = 1,\ldots,\min(m^*,n^*) = 11\}$ of the preprocessed subresultant matrices $\{S_{k,k}(\lambda_{k,k}\tilde{f}^*_{k,k}(\omega_1,\omega_2),\tilde{g}^*_{k,k}(\omega_1,\omega_2))\}$ are plotted in Figure 7.13. The degree t is given by the index of the last numerically rank deficient subresultant matrix in $\{S_{k,k}(\lambda_{k,k}\tilde{f}^*_{k,k}(\omega_1,\omega_2),\tilde{g}^*_{k,k}(\omega_1,\omega_2))\}$ so $t = 8$. Therefore, either

$$t_1 = t - \min(p_1,q_1) = 8 \qquad \text{or} \qquad t_2 = t - \min(p_2,q_2) = 7.$$

Suppose that the candidate $t_1 = 8$ is correctly identified, then t_2 is given by the last numerically rank deficient subresultant matrix in the sequence $\{S_{t_1,k_2}(\lambda_{t_1,k_2}\tilde{f}_{t_1,k_2}(\omega_1,\omega_2),\tilde{g}_{t_1,k_2}(\omega_1,\omega_2))\}$. The minimum singular values of this set of



(i) The minimum singular values of the unprocessed subresultant matrices $\{S_{k_1,k_2}(f(x,y),g(x,y))\}$

(ii) The minimum singular values of the preprocessed subresultant matrices $\{S_{k_1,k_2}(\lambda_{k_1,k_2}\tilde{f}_{k_1,k_2}(\omega_1,\omega_2),\tilde{g}_{k_1,k_2}(\omega_1,\omega_2))\}$

Figure 7.12: The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 7.6.2

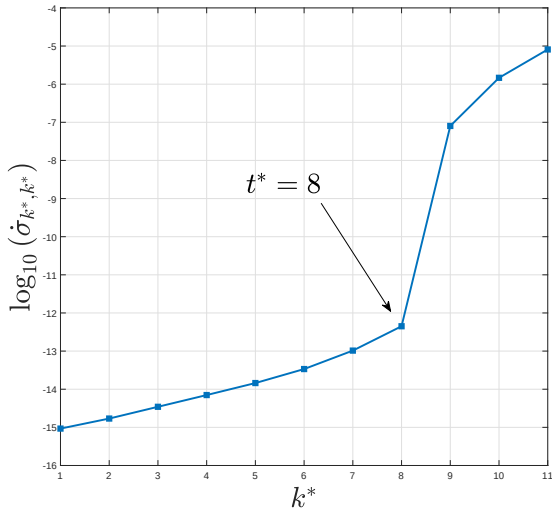
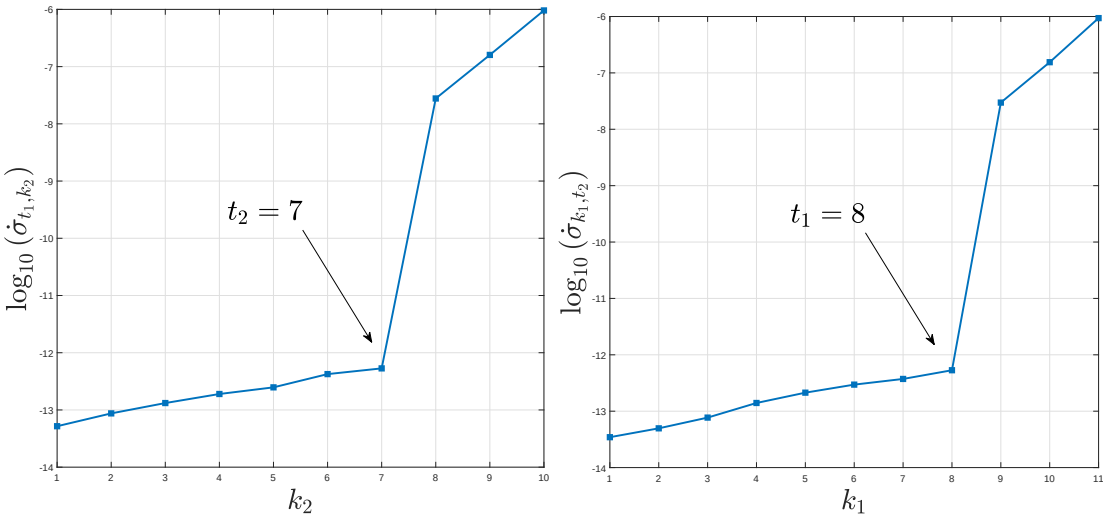


Figure 7.13: BVDRGCD Stage 1 :
The minimum singular values $\{\dot{\sigma}_{k,k}\}$ of the preprocessed subresultant matrices $\{S_k(\lambda_{k,k}\tilde{f}^*_{k,k}(\omega_1,\omega_2),\tilde{g}^*_{k,k}(\omega_1,\omega_2))\}$ in Example 7.6.2

subresultant matrices are plotted in Figure 7.14i, from which it can be seen that the degree of the AGCD with respect to y is given by $t_2 = 7$, so the candidate pair is given by $(t_1, t_2) = (8, 7)$.

Alternatively, suppose that the candidate $t_2 = 7$ is correctly identified, then t_1 is given by the last numerically rank deficient subresultant matrix in the sequence $\{S_{k_1,t_2}(\lambda_{k_1,t_2}\tilde{f}_{k_1,t_2}(\omega_1,\omega_2),\tilde{g}_{k_1,t_2}(\omega_1,\omega_2))\}$. The minimum singular values of this set of subresultant matrices are plotted in Figure 7.14ii, from which it can be seen that the degree of the AGCD with respect to x is given by $t_1 = 8$, so the candidate pair is given by $(t_1, t_2) = (8, 7)$. The candidate pairs are equal so the degree of the AGCD is given by $(t_1, t_2) = (8, 7)$.

Approximations of $\hat{u}_{t_1,t_2}(x,y)$, $\hat{v}_{t_1,t_2}(x,y)$ and $\hat{d}_{t_1,t_2}(x,y)$ are computed from the subresultant matrix of the preprocessed polynomials $S_{t_1,t_2}(\lambda_{t_1,t_2}\tilde{f}_{t_1,t_2}(\omega_1,\omega_2),\tilde{g}_{t_1,t_2}(\omega_1,\omega_2))$ and



(i) BVDRGCD Stage 2a : The minimum singular values of the preprocessed subresultant matrices $\{S_{t_1,k_2}\}$

(ii) BVDRGCD Stage 2b : The minimum singular values of the preprocessed subresultant matrices $\{S_{k_1,t_2}\}$

Figure 7.14: BVDRGCD Stage 2 : The minimum singular values of the subresultant matrices (i) $\{S_{t_1,k_2}\}$ and (ii) $\{S_{k_1,t_2}\}$ in Example 7.6.2

the relative errors between the exact polynomials and the approximations are given in Table 7.1. Note that the first column is left blank since the degree of the AGCD was not computed from the subresultant matrices of the unprocessed polynomials.

The upper bound of noise is reduced to 10^{-14}, and the degree of the AGCD is correctly determined from both the sets of unprocessed and preprocessed subresultant matrices. The coefficients of the cofactor polynomials are computed by the least squares method described in Section 7.5. The approximations $u_{t_1,t_2}(x,y)$, $v_{t_1,t_2}(x,y)$ and $d_{t_1,t_2}(x,y)$ are obtained from the (t_1,t_2)th unprocessed subresultant matrix, and the approximations $\tilde{u}_{t_1,t_2}(x,y)$, $\tilde{v}_{t_1,t_2}(x,y)$ and $\tilde{d}_{t_1,t_2}(x,y)$ are obtained from the (t_1,t_2)th preprocessed subresultant matrix.

The approximations derived from the preprocessed subresultant matrix are significantly better than those derived from the unprocessed subresultant matrix.

	Without Preprocessing $u_{t_1,t_2}(x,y)$, $v_{t_1,t_2}(x,y)$ and $d_{t_1,t_2}(x,y)$	With Preprocessing $\tilde{u}_{t_1,t_2}(x,y)$, $\tilde{v}_{t_1,t_2}(x,y)$ and $\tilde{d}_{t_1,t_2}(x,y)$
Error $\hat{u}_{t_1,t_2}(x,y)$	-	$2.101474e-05$
Error $\hat{v}_{t_1,t_2}(x,y)$	-	$3.082401e-05$
Error $\hat{d}_{t_1,t_2}(x,y)$	-	$2.037024e-05$
Average	-	$2.4070e-05$

Table 7.1: Error in the approximations of $\hat{u}_{t_1,t_2}(x,y)$, $\hat{v}_{t_1,t_2}(x,y)$ and $\hat{d}_{t_1,t_2}(x,y)$, where $\{\epsilon_{f,i_1,i_2}\}$ and $\{\epsilon_{g,j_1,j_2}\}$ are in the interval $[10^{-10}, 10^{-8}]$ in Example 7.6.2

	Without Preprocessing $u_{t_1,t_2}(x,y)$, $v_{t_1,t_2}(x,y)$, and $d_{t_1,t_2}(x,y)$	With Preprocessing $\tilde{u}_{t_1,t_2}(x,y)$, $\tilde{v}_{t_1,t_2}(x,y)$ and $\tilde{d}_{t_1,t_2}(x,y)$
Error $\hat{u}_{t_1,t_2}(x,y)$	$3.018324e-06$	$1.226623e-11$
Error $\hat{v}_{t_1,t_2}(x,y)$	$5.014151e-06$	$1.751399e-11$
Error $\hat{d}_{t_1,t_2}(x,y)$	$2.815919e-06$	$1.160994e-11$
Average	$3.616131e-06$	$1.379672e-11$

Table 7.2: Error in the approximations of $\hat{u}_{t_1,t_2}(x,y)$, $\hat{v}_{t_1,t_2}(x,y)$ and $\hat{d}_{t_1,t_2}(x,y)$ with upper bound of noise $\epsilon_f = \epsilon_g = 10^{-14}$ in Example 7.6.2

□

Example 7.6.3. Consider the Bernstein form of the exact polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ of degrees $(29,15)$ and $(20,17)$ respectively, whose factorisations are given by

$$\begin{aligned}\hat{f}(x,y) &= (x-0.8365498798)^3(x-0.145487821)^{10}(x-0.126479841321)^5\times\\ &\quad (x+y-0.16546978321)^2(x+y+0.5679814354)^3(x+y^2-0.2564878)^4\times\\ &\quad (x^2+y^2-0.46549871232156)\\ \hat{g}(x,y) &= (x-0.8365498798)^3(x-0.126479841321)^5(y-0.45489789123123)^5\times\\ &\quad (x+y-0.35648979126321)^3(x+y-0.16546978321)^2(x+y+0.5679814354)^3\times\\ &\quad (x^2+y^2-0.46549871232156)(x^2+y^2-0.45489789123123).\end{aligned}$$

The polynomials $\hat{f}(x,y)$ and $\hat{g}(x,y)$ have a GCD $\hat{d}_{t_1,t_2}(x,y)$ of degree $(t_1,t_2) = (15,7)$ which is given by

$$\begin{aligned}\hat{d}_{t_1,t_2}(x,y) &= (x-0.8365498798)^3(x-0.126479841321)^5(x+y-0.16546978321)^2\\ &\quad (x+y+0.5679814354)^3(x^2+y^2-0.46549871232156).\end{aligned}$$

Noise is added to the coefficients of $\hat{f}(x,y)$ and $\hat{g}(x,y)$ as in (7.32), where $\{\epsilon_{f,i_1,i_2}\}$ and $\{\epsilon_{g,j_1,j_2}\}$ are uniformly distributed random variables in the interval $[1e-11, 1e-10]$, and the sets of values $\{r_{f,i_1,i_2}\}$ and $\{r_{g,j_1,j_2}\}$ are uniformly distributed random variables in the interval $[-1,1]$.

The inexact polynomials $f(x,y)$ and $g(x,y)$ are preprocessed for each subresultant matrix S_{k_1,k_2} for $k_1 = 1,\ldots,\min(m_1,n_1)$; $k_2 = 1,\ldots,\min(m_2,n_2)$. The coefficients of both the unprocessed and preprocessed polynomials $f(x,y)$ and $g(x,y)$, and preprocessed polynomials $\lambda_{1,1}\tilde{f}_{1,1}(\omega_1,\omega_2)$ and $\tilde{g}_{1,1}(\omega_1,\omega_2)$ are plotted in Figure 7.15. The coefficients of the polynomials $f(x,y)$ and $g(x,y)$ span approximately 18 orders of magnitude, while the coefficients of the preprocessed polynomials span approximately 6 orders of magnitude.

The BVGCD method is considered first. The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices are plotted in Figure 7.16i and Figure 7.16ii respectively.

The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the unprocessed subresultant matrices $S_{k_1,k_2}(\lambda_{k_1,k_2}\tilde{f}_{k_1,k_2}(\omega_1,\omega_2,\tilde{g}_{k_1,k_2}(\omega_1,\omega_2)$ are plotted in Figure 7.16i. From these values it

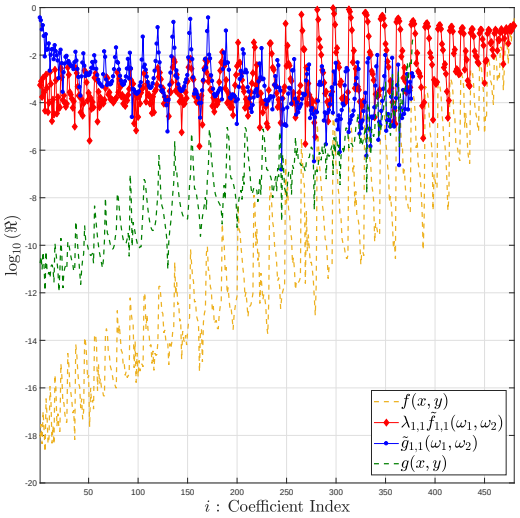
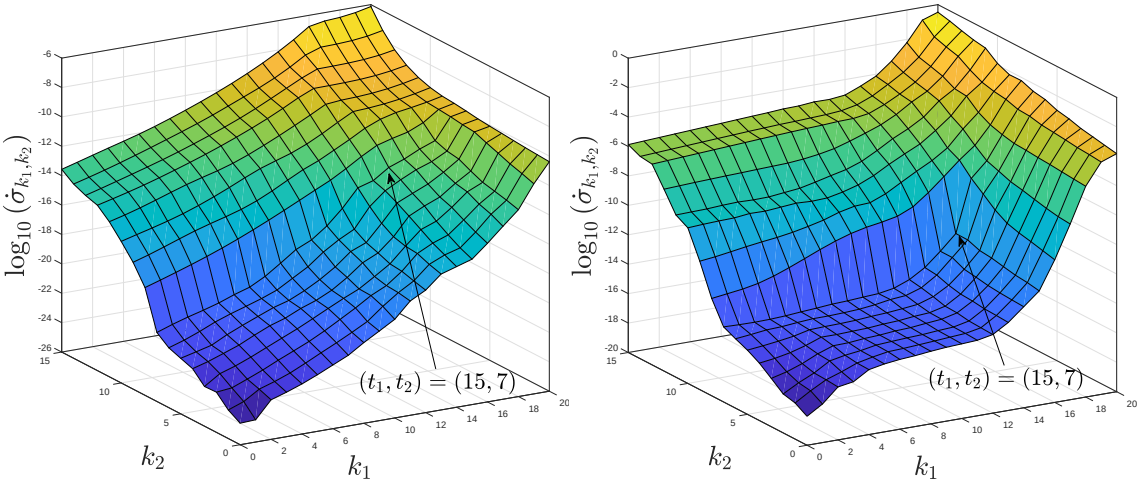


Figure 7.15: The coefficients of both the unprocessed polynomials $f(x,y)$ and $g(x,y)$ and the preprocessed polynomials $\lambda_{1,1}\tilde{f}_{1,1}(\omega_1,\omega_2)$ and $\tilde{g}_{1,1}(\omega_1,\omega_2)$ in Example 7.6.3

is clear to see that the degree of the AGCD cannot be determined, by the DC2 method, from the set of minimum singular values (an arrow points to the location of the last theoretically rank deficient subresultant matrix S_{t_1,t_2}). However, the degree of the AGCD can be determined from the minimum singular values of the preprocessed subresultant matrices by the DC2 method. There is a significant change between $\dot{\rho}_{15,7} = \log_{10}(\dot{\sigma}_{15,7})$ and $r\dot{h}o_{16,8} = \log_{10}(\dot{\sigma}_{16,8})$ and $\delta\rho_{15,7}$ is maximal in the set $\delta\rho_{i,j}$, so the degree of the AGCD is given by $(t_1,t_2) = (15,7)$.



(i) The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of unprocessed subresultant matrices

(ii) The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of preprocessed subresultant matrices

Figure 7.16: The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the (i) unprocessed and (ii) preprocessed subresultant matrices in Example 7.6.3

Alternatively, the BVDRGCD method can be used to compute the degree of the AGCD. The polynomials $f(x,y)$ and $g(x,y)$ are degree elevated by $(p_1,p_2) = (0,14)$ and $(q_1,q_2) = (0,3)$ respectively such that the degree elevated forms $f^*(x,y)$ and $g^*(x,y)$ are of degrees $m^* = 29$ and $n^* = 20$ respectively.

The minimum singular values of the set of preprocessed subresultant matrices $\{S_{k,k}(\lambda_{k,k}\tilde{f}^*_{k,k}(\omega_1,\omega_2),\tilde{g}^*_{k,k}(\omega_1,\omega_2))\}$ are plotted in Figure 7.17, from which, the degree

of the AGCD is identified as $t = 10$. Therefore, either

$$t_1 = t - \min(p_1, q_1) = 10 \qquad \text{or} \qquad t_2 = t - \min(p_2, q_2) = 7.$$

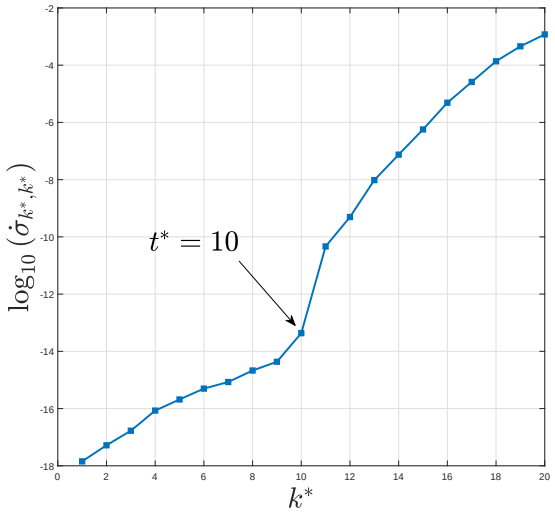


Figure 7.17: BVDRGCD Stage 1 :
The minimum singular values of the preprocessed subresultant matrices $\{S_k(\lambda_{k,k}\tilde{f}^*_{k,k}(\omega_1,\omega_2), \tilde{g}^*_{k,k}(\omega_1,\omega_2))\}$ in Example 7.6.3

Suppose the candidate $t_1 = 10$ is correct, then the corresponding candidate t_2 is given by the index of the last numerically rank deficient subresultant matrix in the set $S_{t_1,k_2}(\lambda_{t_1,k_2}\tilde{f}_{t_1,k_2}(\omega_1,\omega_2), \tilde{g}_{t_1,k_2}(\omega_1,\omega_2))$. The minimum singular values of these subresultant matrices are plotted in Figure 7.18i, from which, the degree of the AGCD with respect to y is given by $t_2 = 7$.

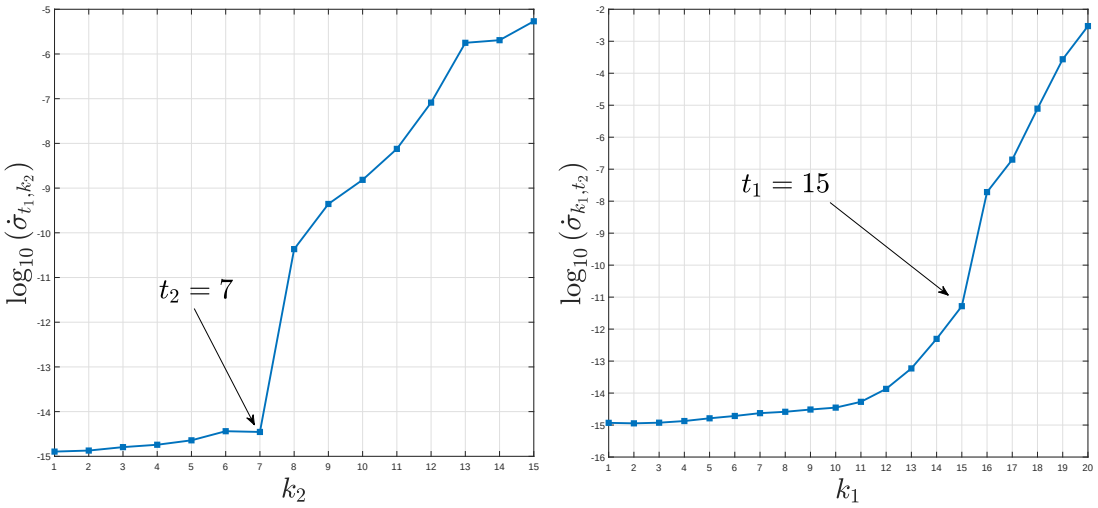
Alternatively, if the candidate $t_2 = 7$ is correct then t_1 is given by the index of the last numerically rank deficient subresultant matrix in the set $\{S_{k_1,t_2}(\lambda_{k_1,t_2}\tilde{f}_{k_1,t_2}(\omega_1,\omega_2), \tilde{g}_{k_1,t_2}(\omega_1,\omega_2))\}$. The minimum singular values of these subresultant matrices are plotted in Figure 7.18ii, from which, the degree of the AGCD with respect to x is given by $t_1 = 15$. The degree of the AGCD is therefore given by either $(10, 7)$ or $(15, 7)$, so $(t_1, t_2) = (15, 7)$.

The coefficients of the GCD triple $\hat{u}_{t_1,t_2}(x,y)$, $\hat{v}_{t_1,t_2}(x,y)$ and $d_{t_1,t_2}(x,y)$ are approximated by the method described in Section 7.5 and the errors are given in Table 7.3.

	Without Preprocessing $u_{t_1,t_2}(x,y)$, $v_{t_1,t_2}(x,y)$ and $d_{t_1,t_2}(x,y)$	With Preprocessing $\tilde{u}_{t_1,t_2}(x,y)$, $\tilde{v}_{t_1,t_2}(x,y)$ and $\tilde{d}_{t_1,t_2}(x,y)$
Error $\hat{u}_{t_1,t_2}(x,y)$	-	$1.455132e-06$
Error $\hat{v}_{t_1,t_2}(x,y)$	-	$5.197778e-06$
Error $\hat{d}_{t_1,t_2}(x,y)$	-	$2.548405e-06$
Average	-	$3.0671e-06$

Table 7.3: Error in the approximations of $\hat{u}_{t_1,t_2}(x,y)$, $\hat{v}_{t_1,t_2}(x,y)$ and $\hat{d}_{t_1,t_2}(x,y)$ in Example 7.6.3

□



(i) BVDRGCD Stage 2a : The minimum singular values of $\{S_{t_1,k_2}\}$

(ii) BVDRGCD Stage 2b : The minimum singular values of $\{S_{k_1,t_2}\}$

Figure 7.18: The minimum singular values of the subresultant matrices in the second stage of the BVDRGCD method in Example 7.6.3

7.6.2 Examples of the Three-Polynomial Problem

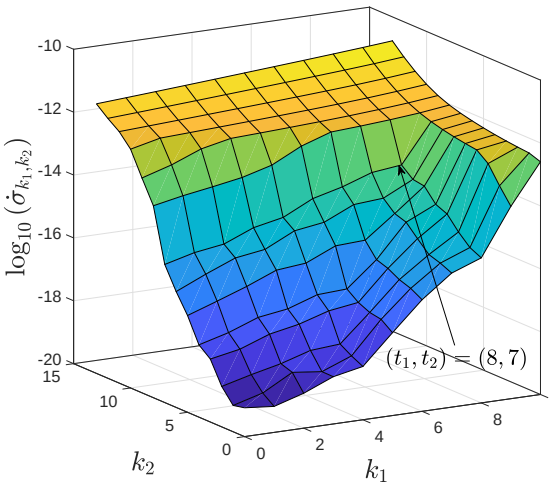
Example 7.6.4. This example uses the same set of polynomials as Example 7.2.2, in which it was shown how the optimal variant of the (k_1, k_2)th three-polynomial subresultant matrix was given by $DTQ^{(ii)}$. However, in this example, more noise is added to the coefficients of $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ such that the degree of the AGCD of the inexact forms is not recoverable from the set of unprocessed subresultant matrices. In Example 7.2.2 only the BVGCD method was considered in the computation of the degree of the AGCD, but in this example both the BVGCD and BVDRGCD methods are used.

The polynomials $\hat{f}(x,y)$, $\hat{g}(x,y)$ and $\hat{h}(x,y)$ were defined in (7.13), (7.14) and (7.15) respectively and noise is added to their coefficients such that the inexact coefficients are given by (7.16), where $\{r_{f,i_1,i_2}\}$, $\{r_{g,j_1,j_2}\}$ and $\{r_{h,p_1,p_2}\}$ are uniformly distributed random variables in the interval $[-1,1]$ and $\{\epsilon_{f,i_1,i_2}\}$, $\{\epsilon_{g,j_1,j_2}\}$ and $\{\epsilon_{h,p_1,p_2}\}$ are uniformly distributed random variables in the interval $[10^{-10}, 10^{-8}]$. This represents a small increase in the noise over Example 7.2.2, but this is sufficient for the computation of the degree of the AGCD to fail if the same methods are applied again.

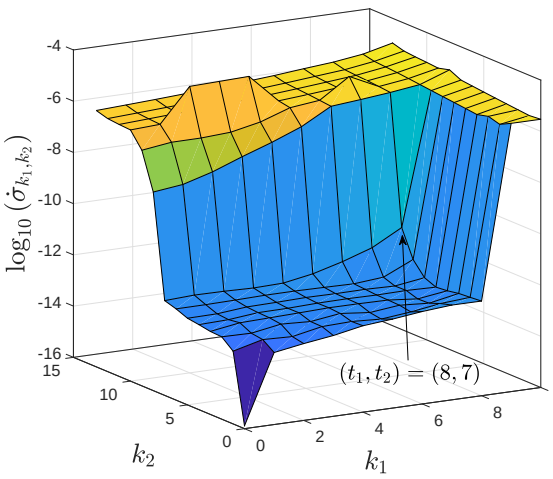
This example begins by considering the BVGCD method. The two-dimensional array of minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the set of unprocessed subresultant matrices $\{\hat{S}_{k_1,k_2}(f(x,y), g(x,y), h(x,y))\}$ are plotted in Figure 7.19i. The minimum singular values of the preprocessed subresultant matrices $\{\hat{S}_{k_1,k_2}(\lambda_{k_1,k_2}\tilde{f}_{k_1,k_2}(\omega_1,\omega_2), \tilde{g}_{k_1,k_2}(\omega_1,\omega_2), \rho_{k_1,k_2}\tilde{h}_{k_1,k_2}(\omega_1,\omega_2))\}$ are plotted in Figure 7.19ii.

There is no significant separation between the numerically zero and non-zero singular values of the unprocessed subresultant matrices, and the degree of the AGCD cannot reliably be determined. However, there is a distinct separation between the numerically zero and non-zero minimum singular values of the preprocessed subresultant matrices and the degree of the AGCD is correctly determined and is given by $(t_1, t_2) = (8, 7)$.

[ii] Where DTQ refers to either the (2×3) partitioned subresultant matrix $\hat{S}_{k_1,k_2}$ or the (3×3) partitioned subresultant matrix $\tilde{S}_{k_1,k_2}$.



(i) The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the unprocessed subresultant matrices



(ii) The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the preprocessed subresultant matrices

Figure 7.19: The minimum singular values $\{\dot{\sigma}_{k_1,k_2}\}$ of the (i) unprocessed (ii) preprocessed subresultant matrices in Example 7.6.4

Alternatively, the new BVDRGCD method is used. The polynomials $f(x,y)$, $g(x,y)$ and $h(x,y)$ are of degrees $(17,13)$, $(20,19)$ and $(10,13)$ respectively and these are degree elevated such that the polynomials $f^*(x,y)$, $g^*(x,y)$ and $h^*(x,y)$ are of degrees $m^*=17$, $n^*=20$ and $o^*=13$. The number of degree elevations of $f(x,y)$, $g(x,y)$ and $h(x,y)$ are given by $(p_1,p_2)=(0,4)$, $(q_1,q_2)=(0,1)$ and $(r_1,r_2)=(3,0)$.

The minimum singular values of the set of preprocessed subresultant matrices $\{\tilde{S}_{k,k}(\lambda_{k,k}\tilde{f}^*_{k,k}(\omega_1,\omega_2),\tilde{g}^*_{k,k}(\omega_1,\omega_2),\rho_{k,k}\tilde{h}^*_{k,k}(\omega_1,\omega_2))\}$ are plotted in Figure 7.20, from which the value t is computed and is given by $t=7$. Given t, either

$$t_1=t-\min(p_1,q_1,r_1)=7 \qquad \text{or} \qquad t_1=t=\min(p_1,q_1,r_1)=7.$$

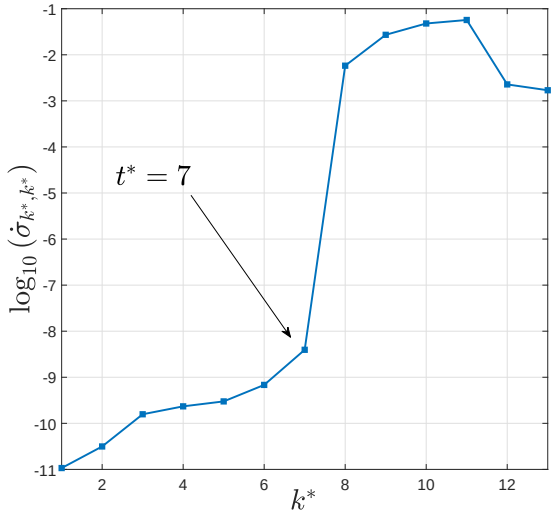
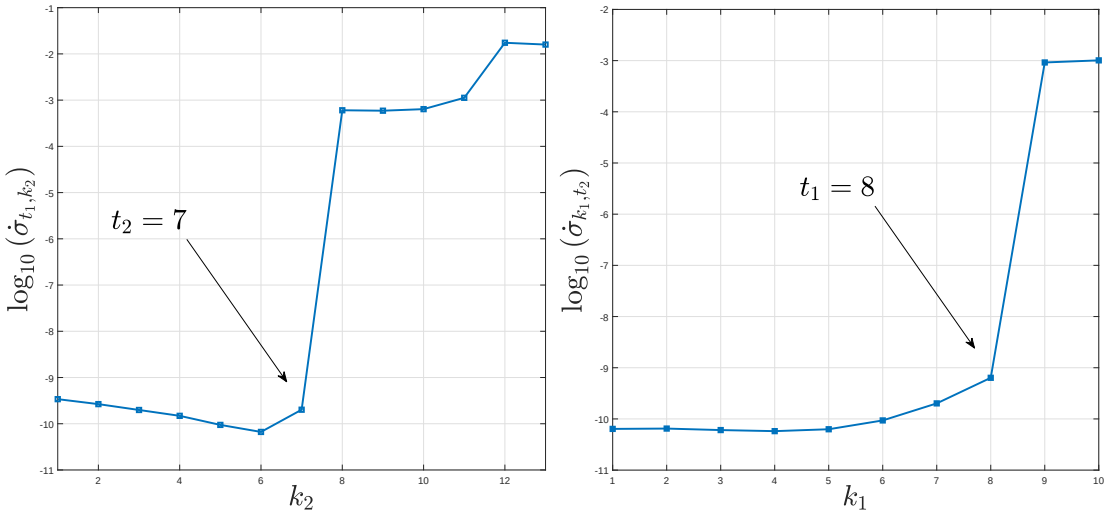


Figure 7.20: BVDRGCD Stage 1 :
The minimum singular values of $\{S_{k,k}\}$ in Example 7.6.4

Suppose that the candidate $t_1=7$ is correct, then the degree of the AGCD with respect to y is given by the index of the last numerically rank deficient subresultant

matrix in the set $S_{t_1,k_2}(\lambda_{t_1,k_2}\tilde{f}_{t_1,k_2}, \tilde{g}_{t_1,k_2}, \rho_{t_1,k_2}\tilde{h}_{t_1,k_2})$. The minimum singular values of these subresultant matrices are plotted in Figure 7.21i, from which, the degree of the AGCD with respect to y is given by $t_2 = 7$.

Alternatively, suppose the candidate $t_2 = 7$ is correct, then the degree of the AGCD with respect to x is given by the index of the last numerically rank deficient subresultant matrix in the set $S_{t_1,k_2}(\lambda_{t_1,k_2}\tilde{f}_{t_1,k_2}(\omega_1,\omega_2), \tilde{g}_{t_1,k_2}(\omega_1,\omega_2), \rho_{t_1,k_2}\tilde{h}_{t_1,k_2}(\omega_1,\omega_2))$. The minimum singular values of these subresultant matrices are plotted in Figure 7.21ii, from which, the degree of the AGCD with respect to x is given by $t_1 = 8$. The two candidate pairs are given by $(t_1, t_2) = (7, 7)$ and $(t_1, t_2) = (8, 7)$ so the degree of the AGCD is given by $(8, 7)$.



(i) BVDRGCD Stage 2a : The minimum singular values $\{\dot{\sigma}_{t_1,k_2}\}$ of the preprocessed subresultant matrices $\{S_{t_1,k_2}\}$

(ii) BVDRGCD Stage 2b : The minimum singular values $\{\dot{\sigma}_{k_1,t_2}\}$ of the preprocessed subresultant matrices $\{S_{k_1,t_2}\}$

Figure 7.21: BVDRGCD Stage 2 : The minimum singular values of (i) $\{S_{t_1,k_2}\}$ and (ii) $\{S_{k_1,t_2}\}$ in the second stage of the BVDRGCD method in Example 7.6.4

The BVGCD method required the evaluation of 150 subresultant matrices while the BVDRGCD method required the evaluation of only 36 subresultant matrices. This represents a significant reduction in computation time.

□

7.7 Conclusion

This chapter has considered extensions of the univariate GCD (UGCD) method to compute the GCD or AGCD of two or three bivariate polynomials in Bernstein form, where the polynomials are defined over a rectangular domain. The first method, BVGCD, was a simple extension of the UGCD in that the method computes the maximal change in a rank-related metric given over a two-dimensional array rather than a one-dimensional array. However this method was considered to be inefficient. The second method, BVDRGCD, uses degree elevation to reduce the number of subresultant matrices which require analysis, thus reducing the problem from a two-dimensional problem to a one-dimensional problem.

This means that it performs significantly faster than BVGCD.

Some further conclusions are outlined below :

The Subresultant Matrices : The Sylvester matrix and sequence of subresultant matrices have been defined for two or three polynomials in Bernstein form which are defined over a rectangular domain. A transformation used in generating this sequence is given in Appendix B.3.

Preprocessing : A simple extension of the preprocessing operations described in earlier chapters allows for the preprocessing of the two-polynomial and three-polynomial subresultant matrices where the polynomials are bivariate and defined over a rectangular domain.

Examples have shown that the degree of the GCD or AGCD of two or three bivariate polynomials is more reliably recovered from the array of subresultant matrices of preprocessed polynomials rather than the array of subresultant matrices of unprocessed polynomials.

Computational Complexity : The cost associated with constructing, preprocessing and computing the SVD of each subresultant matrix in a two-dimensional array is considerable. Instead, a method which makes use of degree elevated forms of $\hat{f}(x, y)$ and $\hat{g}(x, y)$ (and $\hat{h}(x, y)$ in the three-polynomial problem) reduces the problem to one dimension. Many examples have shown that this method has a significantly reduced computational cost with equally well defined results.

Degree elevation is useful in the computation of the degree of the GCD. However, the coefficients of the cofactor polynomials and the GCD are always computed from the (t_1, t_2)th subresultant matrix of preprocessed polynomials which are not in degree elevated form, thus giving the coefficients of the GCD in its base form[(iii)], and preventing any error magnification which may occur as a result of degree elevation.

[(iii)] Where the "base form" of a polynomial is defined as the non-degree elevated form.

Chapter 8

Conclusion

8.1 Thesis Conclusion

The main result of this work is a complete square-free factorisation algorithm for univariate polynomials, and the development of methods required in the computation of an irreducible factorisation of a bivariate polynomial in Bernstein form. The thesis has mostly focused on a variety of univariate and bivariate GCD finding methods for the computation of the GCD of two or three polynomials in Bernstein form.

The key developments in this thesis are now outlined:

The Univariate Sylvester Matrix : A robust method for the computation of the degree and coefficients of the GCD of two polynomials in Bernstein form was presented in Chapter 2. The analysis of the singular values of the sequence of subresultant matrices gave rise to a rank-related metric which was used to compute the degree of the GCD. This eliminated the requirement of a threshold value.

Extensions of the Sylvester Matrix : Two extensions of the Sylvester matrix and subresultant matrices have been presented in this work. Firstly, the three-polynomial Sylvester matrix was defined for univariate polynomials in Bernstein form and was used in the computation of the degree and coefficients of the GCD of three univariate polynomials. Secondly, the Sylvester matrix was extended for the computation of the GCD of two or three bivariate polynomials defined in Bernstein form. Both the triangular and rectangular domain were considered for the bivariate extensions. The Sylvester matrices associated with these two problems were of significantly different structure, and the methods for computing the degree of the GCD were significantly different. The optimal structure of these subresultant matrices has been carefully considered. Experiments have shown that the ordering of the three polynomials in the three-polynomial subresultant matrices can have a significant effect on the ability to determine the numerical rank, and as a consequence, in the ability to determine the degree of the GCD. It was shown how the different orderings related to computing different GCD pairs, and how less well-defined GCDs can also affect the numerical rank determination.

In addition, a previously developed set of preprocessing operations for univariate two-polynomial subresultant matrices has been extended to the new forms of Sylvester

matrix described in this thesis. It has been shown that the degree of the GCD of two or three polynomials is more reliably obtained when the subresultant matrices contain the coefficients of polynomials which have been preprocessed.

Efficient methods of constructing the sequence of subresultant matrices are described in the appendices of this thesis. Given the kth subresultant matrix, methods for computing the $(k+1)$th subresultant matrix of (i) three univariate polynomials in Bernstein form (Appendix B.1.1), (ii) two or three bivariate polynomials in Bernstein form defined over a triangular domain (Appendix B.2) and (iii) two or three bivariate polynomials in Bernstein form defined over a rectangular domain (Appendix B.3) are described.

GCD Coefficient Computation : In Section 3.5 methods for computing approximations of the GCD of two univariate polynomials were considered. A standard least squares based method and a structure preserving low rank approximation method were used to approximate the coefficients of the cofactor polynomials $\hat{u}_t(x)$ and $\hat{v}_t(x)$ and the GCD $\hat{d}_t(x)$. When using the second method, the addition of structured perturbations to the tth subresultant matrix gave a rank deficient subresultant matrix from which better approximations of the cofactor polynomials and GCD were obtained.

The MUGCD Method : The univariate polynomial square-free factorisation problem was considered in Chapter 4, and refinements of the univariate GCD (UGCD) algorithm were included in the modified univariate GCD (MUGCD) method, which was used specifically in the square-free factorisation problem. This thesis has shown that by using the modified univariate GCD (MUGCD) method, the degree of the ith GCD was reliably computed when upper and lower bounds were first determined, where these bounds were derived from the structure of the $(i-1)$th GCD problem. Given the upper and lower bounds, the MUGCD method was shown to be considerably faster than the UGCD method. It was shown that the amount of speed up obtained was dependent on the multiplicity structure of the roots of the polynomial whose square-free factorisation was sought. The algorithm which made use of the MUGCD method was significantly faster than the algorithm which made use of the UGCD method when the polynomial $\hat{f}(x)$ was of high degree and had few roots of high multiplicity.

Deconvolution in the SQFF Problem : Methods for solving the set of deconvolution problems in Gauss' square-free factorisation algorithm were developed in Section 4.2. The problem was shown to have a significant amount of structure in that (i) each $\hat{f}_i(x)$ is the divisor in the ith deconvolution problem and the numerator in the $(i+1)$th and (ii) certain subsets of the set of polynomials $\{\hat{h}_i\}$ are equal and the coefficient matrix can be structured accordingly. The structure of the deconvolution problem was exploited to yield methods which were shown to give improved approximations of the set of polynomials $\{\hat{h}_i(x)\}$.

Univariate Square-Free Factorisation : The root finding method developed in Chapter 4, SQFF, made use of the MUGCD method and the structured matrix based de-

convolution method. This was shown to give improved approximations of polynomial roots when compared with Zeng's `multroot()` and MATLAB `roots()`, which typically fail for polynomials whose roots are of high multiplicity and whose coefficients are defined inexactly.

Bivariate Square-Free Factorisation : The square-free factorisation algorithm due to Gauss (Algorithm 1) was extended to compute the square-free factorisation of a bivariate polynomial in Bernstein form, and it was shown that this problem reduced to (i) the computation of the GCD of three bivariate polynomials and (ii) the computation of a sequence of polynomial deconvolutions.

Three Univariate Polynomial GCD Computation : An initial investigation into the structure of the three univariate polynomial GCD finding problem was completed. Several variations of the subresultant matrix were considered. It was shown both theoretically and by example that it is important for row-partitions of the (2×3) partitioned three-polynomial subresultant matrices to be relatively scaled. In addition, it was shown that if the pairwise GCD of two of the three polynomials is poorly defined, then the (2×3) subresultant matrix which uses the other two pairwise GCDs is optimal.

The Computation of the GCD of Two or Three Bivariate Polynomials over a Triangular Domain : The problems of two-polynomial and three-polynomial GCD computation, where the polynomials are bivariate and defined over a triangular domain, follow from the equivalent univariate problems. The definitions of the Sylvester matrix and subresultant matrices were extended for this particular problem type. The preprocessing operations which yielded improved results for the univariate problems were also extended and gave similarly improved results.

The Computation of the GCD of Two or Three Bivariate Polynomials over a Rectangular Domain : The problems of two-polynomial and three polynomial GCD computation, where the polynomials are bivariate and defined over a rectangular domain, follow from the equivalent univariate problems. However, it has been shown that the extensions required are significantly different to those necessary in the GCD computation for polynomials defined over a triangular domain.

Two extensions for the computation of the GCD of bivariate polynomials were considered. The simple extension, BVGCD, was shown to be computationally expensive since it necessitates the computation of a two-dimensional array of subresultant matrices. Each of these subresultant matrices requires preprocessing and the computation of an SVD.

The second new method, BVDRGCD, makes use of bivariate polynomial degree elevation to reduce the two-dimensional array to a one-dimensional array. Therefore, it has been shown to be more computationally efficient than the BVGCD method while giving equally accurate results.

8.2 Suggestions for Future Research

The QR Decomposition of a Subresultant Matrix in Bernstein Form : The QR decomposition of a subresultant matrix can also be used in the computation of the degree of the GCD. The QR decomposition of the subresultant matrices of two polynomials in the power basis can make use of updating and downdating since each S_k is obtained by the removal of rows and columns from S_{k-1}. This is computationally less expensive than QR decomposition from scratch. However, the subresultant matrix S_k of two univariate polynomials in Bernstein form has entries which are dependent on k and the transformation to compute S_{k+1} given S_k is described in Section 3.1.3. It is therefore not possible to make use of QR updating and downdating methods since the entries of each subresultant matrix are unique. The transformation described in Section 3.1.3 gives S_{k+1} by pre and post multiplication of S_k by orthogonal diagonal matrices. Further work could exploit this to derive a fast method for the QR decomposition of S_{k+1} given the QR decomposition of S_k.

This could be extended to the decomposition of a subresultant matrix of two bivariate polynomials in Bernstein form defined over a rectangular domain. The development of a method in which the QR decomposition of S_{k_1,k_2} is updated to give the QR decomposition of S_{k_1+1,k_2} is particularly appealing. As previously discussed, the computation of the SVD of each subresultant matrix in the $n \times n$ array has significant cost associated, and alternative methods must be sought. One approach to a more efficient bivariate GCD finding method is given by the BVDRGCD method which was described in Section 7.4. Despite BVDRGCD being significantly faster than BVGCD, the method would still benefit from a QR update based method.

The Factorisation of a Bivariate Polynomial in Bernstein Form : The square-free factorisation of a bivariate polynomial is considered in Section 6.1, where it is shown that the problem reduces to a sequence of three-polynomial GCD problems followed by a set of deconvolutions. In this thesis, methods have been developed for the computation of the GCD of two or three arbitrary bivariate polynomials in Bernstein form. The bivariate deconvolution problem is omitted from the thesis but follows directly from the univariate deconvolution problem in Section 4.2.

The Intersection of Bézier Curves and Surfaces by Structured Matrix Methods : This thesis has described a set of methods necessary to compute the factorisation of univariate and bivariate polynomials in Bernstein form.

Although the problems of computing points or areas of intersection between Bézier curves or surfaces have not been fully addressed in this thesis, fundamental components of an algorithm which would solve such problems have been investigated with promising results. The implementation of these components could be used in future work as part of a composite algorithm for solving these intersection problems.

Other Generalisations : A method for the computation of the degree and coefficients of the GCD of two univariate polynomials in Bernstein form has been considered.

This was extended to compute the GCD of three univariate polynomials in Bernstein form. A second extension was to compute the GCD of two or three bivariate polynomials in Bernstein form. This is sufficient for the computation of the points of intersection of two curves or surfaces, however the work can theoretically be extended further to compute the GCD of n polynomials in m variables.

Appendices

Appendix A

Polynomials in Bernstein Form

A.1 Degree Elevation

A.1.1 The Degree Elevation of Univariate Polynomials in Bernstein Form

Degree elevation is unique to polynomials in Bernstein form and degree elevated polynomials can cause computational problems when determining the degree of a polynomial GCD. Methods for the computation of a degree elevated polynomial are discussed by Farouki [27, 29], and a deeper analysis of degree elevation techniques and their complexity is found in [62].

The univariate polynomial $\hat{f}(x)$ of degree m can be degree elevated to an equivalent polynomial $\hat{f}^*$ of degree $(m+p)$ by multiplication with a polynomial $\hat{g}(x)$ of degree p whose coefficients are equal to one. Multiplication of polynomials in Bernstein form is described in Section 2.2.1, and multiplication by a Bernstein polynomial whose coefficients are all equal to one is equivalent to multiplication by one.

Example A.1.1. A polynomial $\hat{f}(x)$ of degree $m = 2$ given by

$$\hat{f}(x) = 7B_0^2(x) + 9.5B_1^2(x) + 15B_2^2(x)$$

is degree elevated to a polynomial $\hat{f}^*(x)$ of degree $(m+p) = 4$ through multiplication with the polynomial

$$\hat{g}(x) = 1B_0^2(x) + 1B_1^2(x) + 1B_2^2(x) \equiv 1 \quad \text{for all } x.$$

The matrix-vector product is given by

$$\hat{\mathbf{h}} = D_4^{-1} T_2 \left(\hat{f}(x) \right) Q_2 \hat{\mathbf{g}},$$

where $D_4^{-1} \in \mathbb{R}^{5\times 5}$ is given by

$$D_4^{-1} = \operatorname{diag} \left[\begin{array}{ccccc} \frac{1}{\binom{4}{0}}, & \frac{1}{\binom{4}{1}}, & \frac{1}{\binom{4}{2}}, & \frac{1}{\binom{4}{3}}, & \frac{1}{\binom{4}{4}} \end{array} \right],$$

the matrix $T_2(\hat{f}(x)) \in \mathbb{R}^{5\times 3}$ is given by

$$\begin{bmatrix} 7\binom{2}{0} & & \\ 9.5\binom{2}{1} & 7\binom{2}{0} & \\ 15\binom{2}{2} & 9.5\binom{2}{1} & 7\binom{2}{0} \\ & 15\binom{2}{2} & 9.5\binom{2}{1} \\ & & 15\binom{2}{2} \end{bmatrix}$$

and the diagonal matrix $Q_2 \in \mathbb{R}^{3\times 3}$ is given by

$$Q_2 = \text{diag}\left[\binom{2}{0}, \quad \binom{2}{1}, \quad \binom{2}{2} \right].$$

The vector $\hat{\mathbf{g}}$ is given by

$$\hat{\mathbf{g}} = \left[\begin{array}{ccc} 1, & 1, & 1 \end{array} \right]^T$$

and the computed product $\hat{\mathbf{h}}$ is given by

$$\left[\begin{array}{ccccc} 7, & 8.25, & 10, & 12.25, & 15 \end{array} \right]^T,$$

so the degree elevated polynomial is given by

$$\hat{f}^*(x) = 7B_0^4(x) + 8.25B_1^4(x) + 10B_2^4(x) + 12.25B_3^4(x) + 15B_4^4(x).$$

A.2 Conversions Between the Bernstein Basis and Power Basis

The conversion from power to Bernstein form and vice versa are ill-conditioned and should be avoided [26]. Methods of conversion between the power and Bernstein bases are developed in [64]. In this section a simple conversion method is considered.

A.2.1 Basis Conversion for Univariate Polynomials

The univariate polynomial $\hat{f}(x)$ given by the power basis representation

$$\hat{f}(x) = \sum_{i=0}^{m} \hat{a}_i x^i$$

can be converted to a polynomial in Bernstein form over the interval $[x_{low}, x_{high}]$ by substituting

$$x = (1-t)x_{low} + tx_{high}$$

which yields

$$\hat{f}(t) = \hat{a}_i \left((1-t)x_{low} + tx_{high}\right).$$

Given a polynomial expressed in the power basis with the coefficients A_i for $i = 0, \ldots, m$, the coefficients of the equivalent polynomial in Bernstein form a_i for $i = 0, \ldots, m$ are given by

$$\hat{a}_i = \sum_{j=0}^{i} \frac{\binom{i}{j}}{\binom{n}{j}} A_k.$$

Similarly, the transformation from Bernstein form to power form is given by

$$A_i = \sum_{k=0}^{i} (-1)^{i-j} \binom{n}{i} \binom{i}{j} \hat{a}_j.$$

Example A.2.1. The polynomial $\hat{f}(x) = x^3 + 7x^2 + 5x^2 + 2$ can be converted to a polynomial in Bernstein form by the transformation matrix T

$$T = \begin{bmatrix} \frac{\binom{3}{0}}{\binom{3}{0}} & 0 & 0 & 0 \\ \frac{\binom{3}{1}}{\binom{3}{1}} & \frac{\binom{2}{0}}{\binom{3}{1}} & 0 & 0 \\ \frac{\binom{3}{2}}{\binom{3}{2}} & \frac{\binom{2}{1}}{\binom{3}{2}} & \frac{\binom{1}{0}}{\binom{3}{2}} & 0 \\ \frac{\binom{3}{3}}{\binom{3}{3}} & \frac{\binom{2}{2}}{\binom{3}{3}} & \frac{\binom{1}{1}}{\binom{3}{3}} & \frac{\binom{0}{0}}{\binom{3}{3}} \end{bmatrix}$$

and the vector of coefficients of $\hat{f}(x)$ in Bernstein form is given by

$$\begin{bmatrix} \frac{\binom{3}{0}}{\binom{3}{0}} & 0 & 0 & 0 \\ \frac{\binom{3}{1}}{\binom{3}{1}} & \frac{\binom{2}{0}}{\binom{3}{1}} & 0 & 0 \\ \frac{\binom{3}{2}}{\binom{3}{2}} & \frac{\binom{2}{1}}{\binom{3}{2}} & \frac{\binom{1}{0}}{\binom{3}{2}} & 0 \\ \frac{\binom{3}{3}}{\binom{3}{3}} & \frac{\binom{2}{2}}{\binom{3}{3}} & \frac{\binom{1}{1}}{\binom{3}{3}} & \frac{\binom{0}{0}}{\binom{3}{3}} \end{bmatrix} \begin{bmatrix} 6 \\ -7 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ \frac{11}{3} \\ \frac{23}{3} \\ 15 \end{bmatrix},$$

so the polynomial in Bernstein form is $2B_0^3(x) + \frac{11}{3}B_1^3(x) + \frac{23}{3}B_2^3(x) + 15B_3^3(x)$.

A.3 De Casteljau's Algorithm

The de Casteljau algorithm is an algorithm for the evaluation of a point on a Bézier curve. It is also used to subdivide curves and is useful for the intersection of curves and Bernstein polynomial zero finding. The de Casteljau algorithm is described throughout the relevant literature [23, Section 3.2] [46, Section 6.8] [38, Section 4.1] [24, Section 4.2.3].

Given the set of $(n+1)$ control points of a curve C_1 of degree n, and a parameter value t, the algorithm splits the curve at t into two curves, $C_{1,left}$ and $C_{1,right}$ both of degree n.

The de Casteljau algorithm is a numerically stable method of evaluating a Bernstein polynomial or Bézier curve at a given parameter value.

The repeated application of the de Casteljau algorithm produces an approximation of the Bézier curve, in which the curve is approximated by the control polygons of composite curves. This is useful in rendering, computing Bernstein polynomial roots and more generally computing intersections of Bézier curves.

Appendix B

Subresultant Matrix Sequences

B.1 The Subresultant Matrix Sequence for Univariate Polynomials in Bernstein Form

In Section 3.1.3 a transformation was described such that given the kth subresultant matrix of two univariate polynomials in Bernstein form, the $(k+1)$th could easily be obtained. This is now extended to the three polynomial subresultant matrix.

B.1.1 Constructing the Three-Polynomial Subresultant Matrix Sequence

The (2×3) partitioned subresultant matrix $\hat{S}_{k+1}(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ with dimensions $(2m+n+o-2k) \times (m+n+o-3k)$ can be computed by a transformation of $\hat{S}_{k+1}(\hat{f}(x), \hat{g}(x), \hat{h}(x))$ whose dimensions are $(2m+n+o-2k+2) \times (m+n+o-3k+3)$. The transformation is given by

$$\hat{S}_{k+1}\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right) = \hat{\mathcal{A}}_k \times \hat{S}_k\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right) \times \tilde{B}_k.$$

The block diagonal matrix $\hat{\mathcal{A}}_k \in \mathbb{R}^{(2m+n+o-2k)\times(2m+n+o-2k+2)}$ is given by

$$\hat{\mathcal{A}}_k = \text{diag}\left[\ \mathcal{A}_{m+n-k},\quad \mathcal{A}_{m+o-k}\ \right],$$

where $\mathcal{A}_{m+n-k} \in \mathbb{R}^{(m+n-k)\times(m+n-k+1)}$ is defined in Equation (3.8) and the matrix $\mathcal{A}_{m+o-k} \in \mathbb{R}^{(m+o-k)\times(m+n-k+1)}$ is similarly defined. The block diagonal matrix $\tilde{\mathcal{B}}_k \in \mathbb{R}^{(m+n+o-3k+3)\times(m+n+o-3k)}$ is given by

$$\tilde{\mathcal{B}}_k = \text{diag}\left[\ \mathcal{B}_{n-k},\quad \mathcal{B}_{o-k},\quad \mathcal{B}_{m-k}\ \right], \tag{B.1}$$

where $\mathcal{B}_{n-k}$ and $\mathcal{B}_{m-k}$ are defined in Equation (3.10) and $\mathcal{B}_{o-k}$ is similarly defined.

By a minor extension, the $(k+1)$th (3×3) partitioned three-polynomial subresultant matrix is given by

$$\tilde{S}_{k+1}\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right) = \tilde{A}_k \times \tilde{S}_k\left(\hat{f}(x), \hat{g}(x), \hat{h}(x)\right) \times \tilde{B}_k,$$

where the block diagonal matrix $\tilde{A}_k \in \mathbb{R}^{(2m+2n+2o-3k)\times(2m+2n+2o-3k+3)}$ is given by

$$\tilde{A}_k = \operatorname{diag}\left[\begin{array}{ccc} \mathcal{A}_{m+n-k}, & \mathcal{A}_{m+o-k}, & \mathcal{A}_{n+o-k} \end{array}\right],$$

and the matrix $\tilde{\mathcal{B}}_k$ is defined in (B.1).

B.2 The Subresultant Matrix Sequence for Polynomials in Bernstein Form Defined over a Triangular Domain

Constructing the Two-Polynomial Subresultant Matrix Sequence

The $(k+1)$th subresultant matrix is given in terms of the kth subresultant matrix by

$$S_{k+1}\left(\hat{f}(x,y),\hat{g}(x,y)\right) = \mathcal{A}_{m+n-k} \times S_k\left(\hat{f}(x,y),\hat{g}(x,y)\right) \times \hat{\mathcal{B}}_k. \tag{B.2}$$

The matrix $\mathcal{A}_{m+n-k} \in \mathbb{R}^{\binom{m+n-k+1}{2}\times\binom{m+n-k+2}{2}}$ is given by

$$\mathcal{A}_{m+n-k} = \left[\begin{array}{cccc|c} \frac{m+n-k}{m+n-k}I_1 & & & & 0_{1,m+n-k+1} \\ & \frac{m+n-k-1}{m+n-k}I_2 & & & 0_{2,m+n-k+1} \\ & & \ddots & & \vdots \\ & & & \frac{1}{m+n-k}I_{m+n-k} & 0_{m+n-k,m+n-k+1} \end{array}\right], \tag{B.3}$$

where $I_j \in \mathbb{R}^{j\times j}$ is the jth identity matrix and $0_{j,m+n-k+1}$ is a zero matrix of size $j \times (m+n-k+1)$.
The matrix $\hat{\mathcal{B}}_k$ is a block diagonal matrix given by

$$\hat{\mathcal{B}} = \operatorname{diag}\left[\begin{array}{cc} \mathcal{B}_{n-k}, & \mathcal{B}_{m-k} \end{array}\right],$$

where the matrices $\mathcal{B}_{n-k} \in \mathbb{R}^{\binom{n-k+2}{2}\times\binom{n-k+1}{2}}$ and $\mathcal{B}_{m-k} \in \mathbb{R}^{\binom{m-k+2}{2}\times\binom{m-k+1}{2}}$ are given by

$$\mathcal{B}_{n-k} = \left[\begin{array}{cccc} \frac{n-k}{n-k}I_1 & & & \\ & \frac{n-k-1}{n-k}I_2 & & \\ & & \ddots & \\ & & & \frac{1}{n-k}I_{n-k} \\ \hline 0_{n-k+1,1} & 0_{n-k+1,2} & \cdots & 0_{n-k+1,n-k} \end{array}\right], \tag{B.4}$$

and

$$\mathcal{B}_{m-k} = \left[\begin{array}{cccc} \frac{m-k}{m-k}I_1 & & & \\ & \frac{m-k-1}{m-k}I_2 & & \\ & & \ddots & \\ & & & \frac{1}{m-k}I_{m-k} \\ \hline 0_{m-k+1,1} & 0_{m-k+1,2} & \cdots & 0_{m-k+1,m-k} \end{array}\right]. \tag{B.5}$$

Constructing the Three-Polynomial Subresultant Matrix Sequence

The (2×3) partitioned three-polynomial subresultant matrix $\hat{S}_{k+1}(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ is similarly derived from the kth subresultant matrix and is given by

$$\hat{S}_{k+1}\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)=\hat{\mathcal{A}}_k\times\hat{S}_k\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)\times\tilde{\mathcal{B}}_k.$$

The matrix $\hat{\mathcal{A}}_k$ is a block diagonal matrix given by

$$\hat{\mathcal{A}}_k=\operatorname{diag}\left[\begin{array}{cc}\mathcal{A}_{m+n-k}, & \mathcal{A}_{m+o-k}\end{array}\right],$$

where the matrix $\mathcal{A}_{m+n-k}$ is already defined in (B.3), and $\mathcal{A}_{m+o-k}$ has an equivalent structure. The matrix $\tilde{\mathcal{B}}_k$ is a block diagonal matrix given by

$$\tilde{\mathcal{B}}_k=\operatorname{diag}\left[\begin{array}{ccc}\mathcal{B}_{n-k}, & \mathcal{B}_{o-k}, & \mathcal{B}_{m-k}\end{array}\right], \tag{B.6}$$

where the matrices $\mathcal{B}_{n-k}$ and $\mathcal{B}_{m-k}$ are defined in (B.4) and (B.5) respectively, and $\mathcal{B}_{o-k}$ has an equivalent structure.

Given the kth (3×3) partitioned subresultant matrix, an extension of the above method allows for the computation of the $(k+1)$th (3×3) partitioned subresultant matrix

$$\tilde{S}_{k+1}\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)=\tilde{\mathcal{A}}_k\tilde{S}_k\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right)\tilde{\mathcal{B}}_k,$$

where $\tilde{\mathcal{A}}$ is the block diagonal matrix given by

$$\tilde{\mathcal{A}}=\operatorname{diag}\left[\begin{array}{ccc}\mathcal{A}_{m+n-k}, & \mathcal{A}_{m+o-k}, & \mathcal{A}_{n+o-k}\end{array}\right]$$

and $\tilde{\mathcal{B}}_k$ is given in (B.6).

B.3 The Subresultant Matrix Sequence for Polynomials in Bernstein Form Defined over a Rectangular Domain

Constructing the Two-Polynomial Subresultant Matrix Sequence

The subresultant matrix $S_{k_1,k_2+1}(\hat{f}(x,y),\hat{g}(x,y))$ is given by

$$S_{k_1,k_2+1}\left(\hat{f}(x,y),\hat{g}(x,y)\right)=\mathcal{A}_{m_2+n_2-k_2}\times S_{k_1,k_2}\left(\hat{f}(x,y),\hat{g}(x,y)\right)\times\hat{\mathcal{B}}_{k_2}.$$

The matrix $\mathcal{A}_{m_2+n_2-k_2}\in\mathbb{R}^{r_{k_1,k_2+1}\times r_{k_1,k_2}}$ is given by

$$\left[\begin{array}{ccccc|c}\frac{m_2+n_2-k_2}{m_2+n_2-k_2}I_{m_1+n_1-k_1+1} & & & & & 0_{m_1+n_1-k_1+1}\\ & \frac{m_2+n_2-k_2-1}{m_2+n_2-k_2}I_{m_1+n_1-k_1+1} & & & & 0_{m_1+n_1-k_1+1}\\ & & \ddots & & & \vdots\\ & & & \frac{1}{m_2+n_2-k_2}I_{m_1+n_1-k_1+1} & & 0_{m_1+n_1-k_1+1}\end{array}\right],$$

where $I_j\in\mathbb{R}^{j\times j}$ is the jth identity matrix and each partition $0_{m_1+n_1-k_1+1}\in\mathbb{R}^{(m_1+n_1-k_1+1)\times(m_1+n_1-k_1+1)}$ is a zero matrix.

The block diagonal matrix $\hat{\mathcal{B}}_{k_2} \in \mathbb{R}^{c_{k_1,k_2} \times c_{k_1,k_2+1}}$ is given by

$$\hat{\mathcal{B}}_{k_2} = \text{diag}\left[\ \mathcal{B}_{n_2-k_2},\quad \mathcal{B}_{m_2-k_2}\ \right],$$

where $\mathcal{B}_{n_2-k_2}$ is a diagonal matrix of order $(n_1-k_1+1)(n_2-k_2+1)$ given by

$$\mathcal{B}_{n_2-k_2} = \left[\begin{array}{cccc} \frac{n_2-k_2}{n_2-k_2} I_{n_1-k_1+1} & & & \\ & \frac{n_2-k_2-1}{n_2-k_2} I_{n_1-k_1+1} & & \\ & & \ddots & \\ & & & \frac{1}{n_2-k_2} I_{n_1-k_1+1} \\ \hline 0_{n_1-k_1+1} & 0_{n_1-k_1+1} & \cdots & 0_{n_1-k_1+1} \end{array}\right]$$

and where $\mathbf{0}_{n_1-k_1+1}$ is a zero matrix of dimensions $(n_1-k_1+1)\times(n_1-k_1+1)$. The diagonal matrix $\mathcal{B}_{m_2-k_2}$ of order $(m_1-k_1+1)(m_2-k_2+1)\times(m_1-k_1+1)(m_2-k_2)$ is given by

$$\mathcal{B}_{m_2-k_2} = \left[\begin{array}{cccc} \frac{m_2-k_2}{m_2-k_2} I_{m_1-k_1+1} & & & \\ & \frac{m_2-k_2-1}{m_2-k_2} I_{m_1-k_1+1} & & \\ & & \ddots & \\ & & & \frac{1}{m_2-k_2} I_{m_1-k_1+1} \\ \hline 0_{m_1-k_1+1} & 0_{m_1-k_1+1} & \cdots & 0_{m_1-k_1+1} \end{array}\right].$$

Constructing the Three-Polynomial Subresultant Matrix Sequence

The (2×3) partitioned subresultant matrix $\hat{S}_{k_1,k_2+1}(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ can be written as a transformation of $\hat{S}_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ and is given by

$$\hat{S}_{k_1,k_2+1}\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right) = \hat{A}_{k_2} \times \hat{S}_{k_1,k_2}\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right) \times \tilde{\mathcal{B}}_{k_2},$$

where the block diagonal matrix $\hat{\mathcal{A}}_{k_2}$ is given by

$$\hat{\mathcal{A}}_{k_2} = \text{diag}\left[\ \mathcal{A}_{m_2+n_2-k_2},\quad \mathcal{A}_{m_2+o_2-k_2}\ \right]$$

and the block diagonal matrix $\tilde{\mathcal{B}}_{k_2}$ is given by

$$\tilde{\mathcal{B}}_{k_2} = \text{diag}\left[\ \mathcal{B}_{n_2-k_2},\quad \mathcal{B}_{o_2-k_2},\quad \mathcal{B}_{m_2-k_2}\ \right]. \tag{B.7}$$

By a simple extension, the (3×3) partitioned subresultant matrix $\tilde{S}_{k_1,k_2+1}(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ can be written in terms of $\tilde{S}_{k_1,k_2}(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y))$ and is given by

$$\tilde{S}_{k_1,k_2+1}\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right) = \tilde{\mathcal{A}}_{k_2} \times \tilde{S}_{k_1,k_2}\left(\hat{f}(x,y),\hat{g}(x,y),\hat{h}(x,y)\right) \times \tilde{\mathcal{B}}_{k_2},$$

where the block diagonal matrix $\tilde{\mathcal{A}}_{k_2}$ is given by

$$\tilde{A}_{k_2} = \text{diag}\left[\ \mathcal{A}_{m_2+n_2-k_2},\quad \mathcal{A}_{m_2+o_2-k_2},\quad \mathcal{A}_{n_2+o_2-k_2}\ \right]$$

and $\tilde{\mathcal{B}}_{k_2}$ is given by (B.7).

Appendix C

Preprocessing

C.1 Preprocessing the Subresultant Matrices of Univariate Polynomials in Bernstein Form

C.1.1 Preprocessing the Three-Polynomial Subresultant Matrices

This section considers the computation of the optimal values of λ, ρ and θ such that the ratio of entry of maximum magnitude in $S_k(\lambda\ddot{f}(\theta,\omega), \ddot{g}(\theta,\omega), \rho\ddot{h}(\theta,\omega))$ to entry of minimum magnitude in $S_k(\lambda\ddot{f}(\theta,\omega), \ddot{g}(\theta,\omega), \rho\ddot{h}(\theta,\omega))$ is minimised. This problem is described in Section 5.3 and the optimal values λ_k, ρ_k and θ_k are given by the solution of a linear programming problem. The minimisation problem is stated as

$$\text{Minimise} \quad \frac{u}{v}$$

Such that

$$\begin{aligned}
u &\geq \frac{\left|\lambda\bar{a}_i\theta^i\binom{m}{i}\binom{n-k}{j}\right|}{\binom{m+n-k}{i+j}} \quad i=0,\ldots,m; \quad j=0,\ldots,n-k,\\
u &\geq \frac{\left|\bar{b}_i\theta^i\binom{n}{i}\binom{m-k}{j}\right|}{\binom{m+n-k}{i+j}} \quad i=0,\ldots,n; \quad j=0,\ldots,m-k,\\
u &\geq \frac{\left|\lambda\bar{a}_i\theta^i\binom{m}{i}\binom{o-k}{j}\right|}{\binom{m+n-k}{i+j}} \quad i=0,\ldots,m; \quad j=0,\ldots,o-k,\\
u &\geq \frac{\left|\rho\bar{c}_i\theta^i\binom{o}{i}\binom{m-k}{j}\right|}{\binom{m+o-k}{i+j}} \quad i=0,\ldots,o; \quad j=0,\ldots,m-k,\\
v &\leq \frac{\left|\lambda\bar{a}_i\theta^i\binom{m}{i}\binom{n-k}{j}\right|}{\binom{m+n-k}{i+j}} \quad i=0,\ldots,m; \quad j=0,\ldots,n-k,\\
v &\leq \frac{\left|\bar{b}_i\theta^i\binom{n}{i}\binom{m-k}{j}\right|}{\binom{m+n-k}{i+j}} \quad i=0,\ldots,n; \quad j=0,\ldots,m-k,\\
v &\leq \frac{\left|\lambda\bar{a}_i\theta^i\binom{m}{i}\binom{o-k}{j}\right|}{\binom{m+n-k}{i+j}} \quad i=0,\ldots,m; \quad j=0,\ldots,o-k,\\
v &\leq \frac{\left|\rho\bar{c}_i\theta^i\binom{o}{i}\binom{m-k}{j}\right|}{\binom{m+o-k}{i+j}} \quad i=0,\ldots,o; \quad j=0,\ldots,m-k.
\end{aligned}$$

Given the transformations

$$
\begin{aligned}
&U = \log_{10}(u), \qquad V = \log_{10}(v), \qquad \bar{\phi} = \log_{10}(\theta)\\
&\bar{\mu}_1 = \log_{10}(\lambda), \qquad , \qquad \bar{\mu}_2 = \log_{10}(\rho)\\
&\bar{\alpha}_{i,j} = \log_{10}\left(\frac{\left|\bar{a}_i\binom{m}{i}\binom{n-k}{j}\right|}{\binom{m+n-k}{i+_j}}\right), \qquad \bar{\beta}_{i,j} = \log_{10}\left(\frac{\left|\bar{b}_i\binom{n}{i}\binom{m-k}{j}\right|}{\binom{m+n-k}{i+j}}\right)\\
&\bar{\mathcal{C}}_{i,j} = \log_{10}\left(\frac{\left|\bar{a}_i\binom{m}{i}\binom{o-k}{j}\right|}{\binom{m+o-k}{i+j}}\right), \qquad \bar{\mathcal{D}}_{i,j} = \log_{10}\left(\frac{\left|\bar{c}_i\binom{o}{i}\binom{m-k}{j}\right|}{\binom{m+o-k}{i+j}}\right),
\end{aligned}
$$

the minimisation problem can be written as

Minimise $\quad U - V$

subject to

$$
\begin{array}{rrrrcll}
U & -i\bar{\phi} & -\bar{\mu}_1 & & \geq & \bar{\alpha}_{i,j} & i = 0,\ldots,m;\quad j = 0,\ldots,n-k,\\
U & -i\bar{\phi} & & & \geq & \bar{\beta}_{i,j} & i = 0,\ldots,n;\quad j = 0,\ldots,m-k,\\
U & -i\bar{\phi} & -\bar{\mu}_1 & & \geq & \bar{\mathcal{C}}_{i,j} & i = 0,\ldots,m;\quad j = 0,\ldots,o-k,\\
U & -i\bar{\phi} & & -\bar{\mu}_2 & \geq & \bar{\mathcal{D}}_{i,j} & i = 0,\ldots,o;\quad j = 0,\ldots,m-k,\\
-V & +i\bar{\phi} & +\bar{\mu}_1 & & \geq & -\bar{\alpha}_{i,j} & i = 0,\ldots,m;\quad j = 0,\ldots,n-k,\\
-V & +i\bar{\phi} & & & \geq & -\bar{\beta}_{i,j} & i = 0,\ldots,n;\quad j = 0,\ldots,m-k,\\
-V & +i\bar{\phi} & +\bar{\mu}_1 & & \geq & -\bar{\mathcal{C}}_{i,j} & i = 0,\ldots,m;\quad j = 0,\ldots,o-k,\\
-V & +i\bar{\phi} & & +\bar{\mu}_2 & \geq & -\bar{\mathcal{D}}_{i,j} & i = 0,\ldots,o;\quad j = 0,\ldots,m-k.
\end{array}
$$

Since j only occurs on the right hand side, the sets

$$
\begin{aligned}
&\bar{\mathcal{M}}_{1,i} = \max\{\,\bar{\alpha}_{i,j} \mid j = 0,\ldots,n-k\,\}, \quad \bar{m}_{1,i} = \min\{\,\bar{\alpha}_{i,j} \mid j = 0,\ldots,n-k\} \quad i = 0,\ldots,m,\\
&\bar{\mathcal{M}}_{2,i} = \max\{\,\bar{\beta}_{i,j} \mid j = 0,\ldots,m-k\}, \quad \bar{m}_{2,i} = \min\{\,\bar{\beta}_{i,j} \mid j = 0,\ldots,m-k\} \quad i = 0,\ldots,n,\\
&\bar{\mathcal{M}}_{3,i} = \max\{\,\bar{\mathcal{C}}_{i,j} \mid j = 0,\ldots,o-k\}, \quad \bar{m}_{3,i} = \min\{\,\bar{\mathcal{C}}_{i,j} \mid j = 0,\ldots,o-k\,\} \quad i = 0,\ldots,m,\\
&\bar{\mathcal{M}}_{4,i} = \max\{\,\bar{\mathcal{D}}_{i,j} \mid j = 0,\ldots,m-k\,\}, \quad \bar{m}_{4,i} = \min\{\,\bar{\mathcal{D}}_{i,j} \mid j = 0,\ldots,m-k\} \quad i = 0,\ldots,o
\end{aligned}
$$

are defined, such that the conditions of the minimisation problem can now be written as

$$
\begin{array}{rrrrcll}
U & -i\bar{\phi} & -\bar{\mu}_1 & & \geq & \bar{\mathcal{M}}_{1,i} & i = 0,\ldots,m,\\
U & -i\bar{\phi} & & & \geq & \bar{\mathcal{M}}_{2,i} & i = 0,\ldots,n,\\
U & -i\bar{\phi} & -\bar{\mu}_1 & & \geq & \bar{\mathcal{M}}_{3,i} & i = 0,\ldots,m,\\
U & -i\bar{\phi} & & -\bar{\mu}_2 & \geq & \bar{\mathcal{M}}_{4,i} & i = 0,\ldots,o,\\
-V & +i\bar{\phi} & +\bar{\mu}_1 & & \geq & -\bar{m}_{1,i} & i = 0,\ldots,m,\\
-V & +i\bar{\phi} & & & \geq & -\bar{m}_{2,i} & i = 0,\ldots,n,\\
-V & +i\bar{\phi} & +\bar{\mu}_1 & & \geq & -\bar{m}_{3,i} & i = 0,\ldots,m,\\
-V & +i\bar{\phi} & & +\bar{\mu}_2 & \geq & -\bar{m}_{4,i} & i = 0,\ldots,o.
\end{array}
$$

The minimisation problem can therefore be written in matrix form as

$$\text{Minimise} \quad \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} U \\ V \\ \bar{\phi} \\ \bar{\mu}_1 \\ \bar{\mu}_2 \end{bmatrix} \quad \text{subject to} \quad A \begin{bmatrix} U \\ V \\ \bar{\phi} \\ \bar{\mu}_1 \\ \bar{\mu}_2 \end{bmatrix} \geq \mathbf{b}. \tag{C.1}$$

The matrix $A \in \mathbb{R}^{(4m+2n+2o+8)\times 5}$ in (C.1) is given by

$$A = \left[\begin{array}{cccc|cccc} \bar{\mathcal{A}}_1 & \bar{\mathcal{A}}_2 & \bar{\mathcal{A}}_1 & \bar{\mathcal{A}}_3 & \bar{a}_1 & \bar{a}_2 & \bar{a}_1 & \bar{a}_3 \end{array} \right]^T,$$

where the matrices $\bar{\mathcal{A}}_1$ and $\bar{a}_1 \in \mathbb{R}^{(m+1)\times 5}$ contain rows of the form

$$\bar{\mathcal{A}}_1 = \begin{bmatrix} 1 & 0 & -i & -1 & 0 \end{bmatrix}, \quad \bar{a}_1 = \begin{bmatrix} 0 & -1 & i & 1 & 0 \end{bmatrix}, \quad \text{for} \quad i = 0, \ldots, m,$$

the matrices $\bar{\mathcal{A}}_2$ and $\bar{a}_2 \in \mathbb{R}^{(n+1)\times 5}$ contain rows of the form

$$\bar{\mathcal{A}}_2 = \begin{bmatrix} 1 & 0 & -i & 0 & 0 \end{bmatrix}, \quad \bar{a}_2 = \begin{bmatrix} 0 & -1 & i & 0 & 0 \end{bmatrix}, \quad \text{for} \quad i = 0, \ldots, n,$$

and the matrices $\bar{\mathcal{A}}_3$ and $\bar{a}_3 \in \mathbb{R}^{(o+1)\times 5}$ contain rows of the form

$$\bar{\mathcal{A}}_3 = \begin{bmatrix} 1 & 0 & -i & 0 & -1 \end{bmatrix}, \quad \bar{a}_3 = \begin{bmatrix} 0 & -1 & i & 0 & 1 \end{bmatrix}, \quad \text{for} \quad i = 1, \ldots, o.$$

The vector $\mathbf{b}$ in (C.1) is given by

$$\mathbf{b} = \left[\begin{array}{cccc|cccc} \bar{\boldsymbol{m}}_1 & \bar{\boldsymbol{m}}_2 & \bar{\boldsymbol{m}}_3 & \bar{\boldsymbol{m}}_4 & -\bar{\boldsymbol{m}}_1 & -\bar{\boldsymbol{m}}_2 & -\bar{\boldsymbol{m}}_3 & -\bar{\boldsymbol{m}}_4 \end{array} \right]^T,$$

where

$$\bar{\boldsymbol{m}}_i = \begin{cases} \begin{bmatrix} \bar{m}_{i,0} & \bar{m}_{i,1} & \ldots & \bar{m}_{i,m} \end{bmatrix}^T \in \mathbb{R}^{m+1} & i = 1, 3, \\ \begin{bmatrix} \bar{m}_{i,0} & \bar{m}_{i,1} & \ldots & \bar{m}_{i,n} \end{bmatrix}^T \in \mathbb{R}^{n+1} & i = 2, \\ \begin{bmatrix} \bar{m}_{i,0} & \bar{m}_{i,1} & \ldots & \bar{m}_{i,o} \end{bmatrix}^T \in \mathbb{R}^{o+1} & i = 4, \end{cases}$$

and

$$\bar{\boldsymbol{m}}_i = \begin{cases} \begin{bmatrix} \bar{m}_{i,0} & \bar{m}_{i,1} & \ldots & \bar{m}_{i,m} \end{bmatrix}^T \in \mathbb{R}^{m+1} & i = 1, 3, \\ \begin{bmatrix} \bar{m}_{i,0} & \bar{m}_{i,1} & \ldots & \bar{m}_{i,n} \end{bmatrix}^T \in \mathbb{R}^{n+1} & i = 2, \\ \begin{bmatrix} \bar{m}_{i,0} & \bar{m}_{i,1} & \ldots & \bar{m}_{i,o} \end{bmatrix}^T \in \mathbb{R}^{o+1} & i = 4. \end{cases}$$

The optimal values λ_k, ρ_k and θ_k are given by 10^{ϕ}, 10^{μ_1} and 10^{μ_2}.

C.2 Preprocessing the Subresultant Matrices of Polynomials in Bernstein Form Defined over a Triangular Domain

C.2.1 The Arithmetic Mean of the Non-Zero Entries of the $(n-k)$th Order Convolution Matrix

This section simplifies the expression used for the computation of the arithmetic mean of the non-zero entries in the $(n-k)$th order convolution matrix of the polynomial $\hat{f}(x,y)$. Given the simplified expression, an expression is derived such that the arithmetic mean of the $(n-k-1)$th order convolution matrix can be determined with minimal computational cost.

Proposition 4. *The arithmetic mean of the non-zero entries in the first partition, $C_{n-k}(\hat{f}(x,y))$, of the kth subresultant matrix is given by*

$$\mathcal{A}_{n-k}\left(\hat{f}(x,y)\right) = \frac{\binom{m+n-k+2}{2}}{\binom{m+2}{2}^2\binom{n-k+2}{2}} \sum_{i_1+i_2=0}^{m} \hat{a}_{i_1,i_2}.$$

Proof. Each of the $\binom{m+2}{2}$ coefficients of $\hat{f}(x,y)$ appear in each of the $\binom{n-k+2}{2}$ columns of $C_{n-k}(\hat{f}(x,y))$ so the arithmetic mean is given by

$$\mathcal{A}_{n-k}\left(\hat{f}(x,y)\right) = \frac{1}{\binom{m+2}{2}\binom{n-k+2}{2}} \sum_{i_1+i_2=0}^{m} \sum_{j_1+j_2=0}^{n-k} \hat{a}_{i_1,i_2} \frac{\binom{i_1+j_1}{i_1}\binom{i_2+j_2}{i_2}\binom{m+n-k-i_1-i_2-j_1-j_2}{m-i_1-i_2}}{\binom{m+n-k}{n-k}}. \tag{C.2}$$

The sum of the entries containing any given $\hat{a}_{i_1,i_2}$ is given by

$$\begin{aligned}\sum_{j_1+j_2=0}^{n-k} \hat{a}_{i_1,i_2} \frac{\binom{i_1+j_1}{i_1}\binom{i_2+j_2}{i_2}\binom{m+n-k-i_1-i_2-j_1-j_2}{m-i_1-i_2}}{\binom{m+n-k}{m}} &= \hat{a}_{i_1,i_2} \frac{\binom{m+n-k+2}{2}}{\binom{m+2}{2}} \\ &= \hat{a}_{i_1,i_2} \frac{(m+n-k+1)(m+n-k+2)}{(m+1)(m+2)}\end{aligned}$$

and therefore (C.2) can be reduced to the significantly less complex expression

$$\mathcal{A}_{n-k}\left(\hat{f}(x,y)\right) = \frac{1}{\binom{m+2}{2}\binom{n-k+2}{2}} \sum_{i_1+i_2=0}^{m} \hat{a}_{i_1,i_2} \frac{\binom{m+n-k+2}{2}}{\binom{m+2}{2}} = \frac{\binom{m+n-k+2}{2}}{\binom{m+2}{2}^2\binom{n-k+2}{2}} \sum_{i_1+i_2=0}^{m} \hat{a}_{i_1,i_2}. \tag{C.3}$$

□

Proposition 5. *Given the arithmetic mean $\mathcal{A}_{n-k}(\hat{f}(x,y))$ of the non-zero entries in the $(n-k)$th order convolution matrix, the arithmetic mean $\mathcal{A}_{n-k-1}(\hat{f}(x,y))$ given by*

$$\mathcal{A}_{n-k-1}\left(\hat{f}(x,y)\right) = \frac{\binom{m+n-k+1}{2}}{\binom{m+2}{2}^2\binom{n-k+1}{2}} \sum_{i_1+i_2=0}^{m} \hat{a}_{i_1,i_2}$$

is more simply given by

$$\mathcal{A}_{n-k-1}\left(\hat{f}(x,y)\right) = \frac{(m+n-k)(n-k+2)}{(n-k)(m+n-k+2)}\mathcal{A}_{n-k}\left(\hat{f}(x,y)\right).$$

Proof. The arithmetic mean of the non-zero entries in the first partition of the kth subresultant matrix are given by (C.3) and the arithmetic mean of the non-zero entries in the first partition of the $(k+1)$th subresultant matrix is given by

$$\mathcal{A}_{n-k-1}\left(\hat{f}(x,y)\right) = \frac{\binom{m+n-k+1}{2}}{\binom{m+2}{2}^2\binom{n-k+1}{2}}\sum_{i_1+i_2=0}^{m}\hat{a}_{i_1,i_2}.$$

The expression x such that $\mathcal{A}_{n-k-1}(\hat{f}(x,y)) = x\mathcal{A}_{n-k}(\hat{f}(x,y))$ is given by

$$\begin{aligned} x == & \frac{\frac{\binom{m+n-k+1}{2}}{\binom{m+2}{2}^2\binom{n-k+1}{2}}\sum_{i1+i2=0}^{m}\hat{a}_{i_1,i_2}}{\frac{\binom{m+n-k+2}{2}}{\binom{m+2}{2}^2\binom{n-k+2}{2}}\sum_{i1+i2=0}^{m}\hat{a}_{i_1,i_2}} = \frac{\binom{m+n-k+1}{2}\binom{m+2}{2}^2\binom{n-k+2}{2}\hat{a}_{i_1,i_2}}{\binom{m+2}{2}^2\binom{n-k+1}{2}\binom{m+n-k+2}{2}\hat{a}_{i_1,i_2}} \\ = & \frac{(m+n-k+1)(n-k+2)}{(n-k)(m+n-k+2)}, \end{aligned}$$

so the arithmetic mean of the entries in the first partition of the $(k+1)$th subresultant in terms of $\mathcal{A}_{n-k}(\hat{f}(x,y))$ is given by

$$\mathcal{A}_{n-k-1}\left(\hat{f}(x,y)\right) = \mathcal{A}_{n-k}\left(\hat{f}(x,y)\right) \times \frac{(m+n-k+1)(n-k+2)}{(n-k)(m+n-k+2)}.$$

A similar expression is easily obtained for $\mathcal{A}_{m-k-1}(\hat{g}(x,y))$. □

C.2.2 The Geometric Mean of the Non-Zero Entries of the $(n-k)$th Order Convolution Matrix

This section considers the computation of the geometric mean $\mathcal{G}_{n-k}\left(\hat{f}(x,y)\right)$ of the non-zero entries in the $(n-k)$th convolution matrix of the polynomial $\hat{f}(x,y)$. Each of the $\binom{m+2}{2}$ coefficients $\hat{a}_{i_1,i_2}$ occur in each of the $\binom{n-k+2}{2}$ columns of $C_{n-k}(\hat{f}(x,y))$, the first partition of $S_k(\hat{f}(x,y),\hat{g}(x,y))$. The geometric mean $\mathcal{G}_{n-k}\left(\hat{f}(x,y)\right)$ of the non-zero entries in $C_{n-k}(\hat{f}(x,y))$ is given by

$$\mathcal{G}_{n-k}\left(\hat{f}(x,y)\right) = \prod_{j_1+j_2=0}^{n-k}\prod_{i_1+i_2=0}^{m}\left(\hat{a}_{i_1,i_2}\frac{\binom{m}{i_1,i_2}\binom{n-k}{j_1,j_2}}{\binom{m+n-k}{i_1+j_1,i_2+j_2}}\right)^{\frac{1}{\binom{m+2}{2}\times\binom{n-k+2}{2}}}. \tag{C.4}$$

Proposition 6. *The geometric mean of the non-zero entries of $C_{n-k}(\hat{f}(x,y))$ is written as*

$$\mathcal{G}_{n-k}\left(\hat{f}(x,y)\right) = \prod_{j_1+j_2=0}^{n-k}\prod_{i_1+i_2=0}^{m}\left(\frac{\hat{a}_{i_1,i_2}\binom{i_1+j_1}{i_1}\binom{i_2+j_2}{i_2}\binom{m+n-k-i_1-i_2-j_1-j_2}{m-i_1-i_2}}{\binom{m+n-k}{n-k}}\right)^{\frac{1}{\binom{m+2}{2}\binom{n-k+2}{2}}}$$

and this can be simplified to

$$\mathcal{G}_{n-k}\left(\hat{f}(x,y)\right) = \frac{1}{\binom{m+n-k}{n-k}} \left(\prod_{i_1+i_2=0}^{m} \hat{a}_{i_1,i_2} \right)^{\frac{1}{\binom{m+2}{2}}} \left(\prod_{i_1+i_2=0}^{m} \prod_{j_1+j_2=0}^{n-k} \binom{i_1+j_1}{j_1} \right)^{\frac{3}{\binom{m+2}{2}\binom{n-k+2}{2}}}.$$

Proof. The proof proceeds by considering the computation of $\mathcal{G}_{n-k}\left(\hat{f}(x,y)\right)$ in three parts

$$\mathcal{A}_{n-k}\left(\hat{f}(x,y)\right) = \mathcal{A} \times \mathcal{B}_k \times C_k.$$

Firstly, the geometric mean of the coefficients $\hat{a}_{i_1,i_2}$ is simplified as

$$\mathcal{A} = \left(\prod_{i_1+i_2=0}^{m} \prod_{j_1+j_2=0}^{n-k} \hat{a}_{i_1,i_2} \right)^{\frac{1}{\binom{m+2}{2}\binom{n-k+2}{2}}} = \left(\prod_{i_1+i_2=0}^{m} \hat{a}_{i_1,i_2} \right)^{\frac{1}{\binom{m+2}{2}}}.$$

Secondly, let $\mathcal{B}_k$ be given by

$$\mathcal{B}_k = \left(\prod_{i_1+i_2=0}^{m} \prod_{j_1+j_2=0}^{n-k} \binom{i_1+j_1}{j_1} \binom{i_2+j_2}{j_2} \binom{m+n-k-i_1-i_2-j_1-j_2}{m-i_1-i_2} \right)^{\frac{1}{\binom{m+2}{2}\binom{n-k+2}{2}}}$$

and notice that

$$\prod_{i_1+i_2=0}^{m} \prod_{j_1+j_2=0}^{n-k} \binom{i_1+j_1}{j_1} = \prod_{i_1+i_2=0}^{m} \prod_{j_1+j_2=0}^{n-k} \binom{i_2+j_2}{j_2} = \prod_{i_1+i_2=0}^{m} \prod_{j_1+j_2=0}^{n-k} \binom{m+n-k-i_1-i_2-j_1-j_2}{m-i_1-i_2},$$

so $\mathcal{B}_k$ reduces to

$$\mathcal{B}_k = \left(\prod_{i_1+i_2=0}^{m} \prod_{j_1+j_2=0}^{n-k} \binom{i_1+j_1}{j_1} \right)^{\frac{3}{\binom{m+2}{2}\binom{n-k+2}{2}}}.$$

Finally, the denominator of each of the non-zero entries in the first partition of the subresultant matrix is constant and

$$C_k = \left(\prod_{j_1+j_2=0}^{n-k} \prod_{i_1+i_2=0}^{m} \frac{1}{\binom{m+n-k}{n-k}} \right)^{\frac{1}{\binom{m+2}{2}\binom{n-k+2}{2}}} = \frac{1}{\binom{m+n-k}{n-k}}.$$

So, $\mathcal{G}_{n-k}(\hat{f}(x,y))$ can be written as

$$\mathcal{G}_{n-k}(\hat{f}(x,y)) = \frac{1}{\binom{m+n-k}{n-k}} \left(\prod_{i_1+i_2=0}^{m} \hat{a}_{i_1,i_2} \right)^{\frac{1}{\binom{m+2}{2}}} \left(\prod_{i_1+i_2=0}^{m} \prod_{j_1+j_2=0}^{n-k} \binom{i_1+j_1}{j_1} \right)^{\frac{3}{\binom{m+2}{2}\binom{n-k+2}{2}}}.$$

□

C.2.3 Preprocessing the Two-Polynomial Subresultant Matrices

This section addresses the minimisation problem in preprocessing the subresultant matrix of two bivariate polynomials in Bernstein form defined over a triangular domain. Variables

λ, θ_1 and θ_2 must be optimised such that the ratio of entry of maximum magnitude to entry of minimum magnitude in the kth subresultant matrix is minimised. The minimisation problem staged in Section 6.4 is reduced to the form

$$
\begin{array}{ll}
\text{Minimise} & \dfrac{u}{v} \\
\text{subject to} & \\
u & \geq \dfrac{\left|\lambda \bar{a}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{m}{i_1,i_2}\binom{n-k}{j_1,j_2}\right|}{\binom{m+n-k}{i_1+j_1,i_2+j_2}} \quad i_1+i_2=0,\ldots,m; \quad j_1+j_2=0,\ldots,n-k, \\
u & \geq \dfrac{\left|\bar{b}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{n}{i_1,i_2}\binom{m-k}{j_1,j_2}\right|}{\binom{m+n-k}{i_1+j_1,i_2+j_2}} \quad i_1+i_2=0,\ldots,n; \quad j_1+j_2=0,\ldots,m-k, \\
v & \leq \dfrac{\left|\lambda \bar{a}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{m}{i_1,i_2}\binom{n-k}{j_1,j_2}\right|}{\binom{m+n-k}{i_1+j_1,i_2+j_2}} \quad i_1+i_2=0,\ldots,m; \quad j_1+j_2=0,\ldots,n-k, \\
v & \leq \dfrac{\left|\bar{b}_{i_1,i_2}\theta_1^{i_1}\theta_2^{i_2}\binom{n}{i_1,i_2}\binom{m-k}{j_1,j_2}\right|}{\binom{m+n-k}{i_1+j_1,i_2+j_2}} \quad i_1+i_2=0,\ldots,n; \quad j_1+j_2=0,\ldots,m-k, \\
v & > 0, \\
\theta_1 & > 0, \\
\theta_2 & > 0, \\
\lambda & > 0.
\end{array}
\tag{C.5}
$$

By the transformations

$$
U=\log_{10}(u), \quad V=\log_{10}(v), \quad \bar{\phi}_1=\log_{10}(\theta_1), \quad \bar{\phi}_2=\log_{10}(\theta_2), \quad \bar{\mu}_1=\log_{10}(\lambda) \tag{C.6}
$$

$$
\bar{\alpha}_{i_1,i_2,j_1,j_2}=\log_{10}\left(\frac{\left|\bar{a}_{i_1,i_2}\binom{m}{i_1,i_2}\binom{n-k}{j_1,j_2}\right|}{\binom{m+n-k}{i_1+j_1,i_2+j_2}}\right), \quad \bar{\beta}_{i_1,i_2,j_1,j_2}=\log_{10}\left(\frac{\left|\bar{b}_{i_1,i_2}\binom{n}{i_1,i_2}\binom{m-k}{j_1,j_2}\right|}{\binom{m+n-k}{i_1+j_1,i_2+j_2}}\right), \tag{C.7}
$$

the minimisation problem (C.5) can be written as

$$
\begin{array}{lllllll}
\text{Minimise} & U-V & & & & & \\
\text{Subject to} & & & & & & \\
U & -i_1\bar{\phi}_1 & -i_2\bar{\phi}_2 & -\bar{\mu}_1 & \geq & \bar{\alpha}_{i_1,i_2,j_1,j_2} & i_1+i_2=0,\ldots,m; \quad j_1+j_2=0,\ldots,n-k, \\
U & -i_1\bar{\phi}_1 & -i_2\bar{\phi}_2 & & \geq & \bar{\beta}_{i_1,i_2,j_1,j_2} & i_1+i_2=0,\ldots,n; \quad j_1+j_2=0,\ldots,m-k, \\
-V & +i_1\bar{\phi}_1 & +i_2\bar{\phi}_2 & +\bar{\mu}_1 & \geq & -\bar{\alpha}_{i_1,i_2,j_1,j_2} & i_1+i_2=0,\ldots,m; \quad j_1+j_2=0,\ldots,n-k, \\
-V & +i_1\bar{\phi}_1 & +i_2\bar{\phi}_2 & & \geq & -\bar{\beta}_{i_1,i_2,j_1,j_2} & i_1+i_2=0,\ldots,n; \quad j_1+j_2=0,\ldots,m-k.
\end{array}
\tag{C.8}
$$

The counters j_1 and j_2 appear only on the right hand side of these inequalities, thus $\bar{m}_{1,i_1,i_2}$, $\bar{m}_{2,i_1,i_2}$, $\bar{m}_{1,i_1,i_2}$ and $\bar{m}_{2,i_1,i_2}$ can be defined as

$$
\bar{m}_{1,i_1,i_2}=\max_{j_1+j_2=0,\ldots,n-k}\{\bar{\alpha}_{i_1,i_2,j_1,j_2}\} \quad \text{for} \quad i_1+i_2=0,\ldots,m, \tag{C.9}
$$

$$
\bar{m}_{2,i_1,i_2}=\max_{j_1+j_2=0,\ldots,m-k}\{\bar{\beta}_{i_1,i_2,j_1,j_2}\} \quad \text{for} \quad i_1+i_2=0,\ldots,n, \tag{C.10}
$$

$$
\bar{m}_{1,i_1,i_2}=\min_{j_1+j_2=0,\ldots,n-k}\{\bar{\alpha}_{i_1,i_2,j_1,j_2}\} \quad \text{for} \quad i_1+i_2=0,\ldots,m, \tag{C.11}
$$

$$
\bar{m}_{2,i_1,i_2}=\min_{j_1+j_2=0,\ldots,m-k}\{\bar{\beta}_{i_1,i_2,j_1,j_2}\} \quad \text{for} \quad i_1+i_2=0,\ldots,n, \tag{C.12}
$$

and the minimisation problem (C.8) can be written as

$$
\begin{aligned}
&\text{Minimise} \quad U - V \\
&\text{subject to} \\
&\begin{array}{cccccccl}
U & & -i_1\bar{\phi}_1 & -i_2\bar{\phi}_2 & -\bar{\mu}_1 & \geq & \bar{m}_{1,i_1,i_2} & i_1+i_2=0,\ldots,m, \\
U & & -i_1\bar{\phi}_1 & -i_2\bar{\phi}_2 & & \geq & \bar{m}_{2,i_1,i_2} & i_1+i_2=0,\ldots,n, \\
 & -V & +i_1\bar{\phi}_1 & +i_2\bar{\phi}_2 & +\bar{\mu}_1 & \geq & -\bar{m}_{1,i_1,i_2} & i_1+i_2=0,\ldots,m, \\
 & -V & +i_1\bar{\phi}_1 & +i_2\bar{\phi}_2 & & \geq & -\bar{m}_{2,i_1,i_2} & i_1+i_2=0,\ldots,n,
\end{array}
\end{aligned}
$$

which in matrix form is given by

$$
\text{Minimise} \quad \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \end{bmatrix} \quad \text{subject to} \quad A \begin{bmatrix} U \\ V \\ \bar{\phi}_1 \\ \bar{\phi}_2 \\ \bar{\mu}_1 \end{bmatrix} \geq \mathbf{b}. \tag{C.13}
$$

The matrix $A \in \mathbb{R}^{\left(2\binom{m+2}{2}+2\binom{n+2}{2}\right)\times 5}$ in (C.13) is given by

$$
A = \begin{bmatrix} \bar{\mathcal{A}}_1 & \bar{\mathcal{A}}_2 & \bar{a}_1 & \bar{a}_2 \end{bmatrix}^T,
$$

where the matrices $\bar{\mathcal{A}}_1$ and $\bar{a}_1 \in \mathbb{R}^{\binom{m+2}{2}\times 5}$ are given by

$$
\bar{\mathcal{A}}_1 = \left[\begin{array}{ccccc}
1 & 0 & 0 & 0 & -1 \\ \hline
1 & 0 & -1 & 0 & -1 \\
1 & 0 & 0 & -1 & -1 \\ \hline
1 & 0 & -2 & 0 & -1 \\
1 & 0 & -1 & -1 & -1 \\
1 & 0 & 0 & -2 & -1 \\ \hline
\vdots & \vdots & \vdots & \vdots & \vdots \\ \hline
1 & 0 & -m & 0 & -1 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
1 & 0 & 0 & -m & -1
\end{array}\right], \quad
\bar{a}_1 = \left[\begin{array}{ccccc}
0 & -1 & 0 & 0 & 1 \\ \hline
0 & -1 & 1 & 0 & 1 \\
0 & -1 & 0 & 1 & 1 \\ \hline
0 & -1 & 2 & 0 & 1 \\
0 & -1 & 1 & 1 & 1 \\
0 & -1 & 0 & 2 & 1 \\ \hline
\vdots & \vdots & \vdots & \vdots & \vdots \\ \hline
0 & -1 & m & 0 & 1 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
0 & -1 & 0 & m & 1
\end{array}\right]
$$

and the matrices $\bar{\mathcal{A}}_2$ and $\bar{a}_2 \in \mathbb{R}^{\binom{n+2}{2}\times 5}$ are given by

$$
\bar{\mathcal{A}}_2 = \left[\begin{array}{ccccc}
1 & 0 & 0 & 0 & 0 \\ \hline
1 & 0 & -1 & 0 & 0 \\
1 & 0 & 0 & -1 & 0 \\ \hline
1 & 0 & -2 & 0 & 0 \\
1 & 0 & -1 & -1 & 0 \\
1 & 0 & 0 & -2 & 0 \\ \hline
\vdots & \vdots & \vdots & \vdots & \vdots \\ \hline
1 & 0 & -n & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
1 & 0 & 0 & -n & 0
\end{array}\right], \quad
\bar{a}_2 = \left[\begin{array}{ccccc}
0 & -1 & 0 & 0 & 0 \\ \hline
0 & -1 & 1 & 0 & 0 \\
0 & -1 & 0 & 1 & 0 \\ \hline
0 & -1 & 2 & 0 & 0 \\
0 & -1 & 1 & 1 & 0 \\
0 & -1 & 0 & 2 & 0 \\ \hline
\vdots & \vdots & \vdots & \vdots & \vdots \\ \hline
0 & -1 & n & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
0 & -1 & 0 & n & 0
\end{array}\right].
$$

The vector $\mathbf{b} \in \mathbb{R}^{\left(2\binom{m+2}{2}\times 2\binom{n+2}{2}\right)}$ in (C.13) is given by

$$\mathbf{b} = \begin{bmatrix} \bar{\boldsymbol{m}}_1, & \bar{\boldsymbol{m}}_2, & -\bar{\boldsymbol{m}}_1, & -\bar{\boldsymbol{m}}_2 \end{bmatrix}^T \in \mathbb{R}^{\left(2\binom{m+2}{2}+2\binom{n+2}{2}\right)},$$

where

$$\bar{\boldsymbol{m}}_1 = \begin{bmatrix} \bar{m}_{1,0,0}, & \bar{m}_{1,1,0}, & \bar{m}_{1,0,1}, & \ldots, & \bar{m}_{1,0,m} \end{bmatrix} \in \mathbb{R}^{\binom{m+2}{2}} \tag{C.14}$$

$$\bar{\boldsymbol{m}}_2 = \begin{bmatrix} \bar{m}_{2,0,0}, & \bar{m}_{2,1,0}, & \bar{m}_{2,0,1}, & \ldots, & \bar{m}_{2,0,n} \end{bmatrix} \in \mathbb{R}^{\binom{n+2}{2}} \tag{C.15}$$

$$\bar{\boldsymbol{m}}_1 = \begin{bmatrix} \bar{m}_{1,0,0}, & \bar{m}_{1,1,0}, & \bar{m}_{1,0,1}, & \ldots, & \bar{m}_{1,0,m} \end{bmatrix} \in \mathbb{R}^{\binom{m+2}{2}} \tag{C.16}$$

$$\bar{\boldsymbol{m}}_2 = \begin{bmatrix} \bar{m}_{2,0,0}, & \bar{m}_{2,1,0}, & \bar{m}_{2,0,1}, & \ldots, & \bar{m}_{2,0,n} \end{bmatrix} \in \mathbb{R}^{\binom{n+2}{2}}. \tag{C.17}$$

The optimal values λ, θ_1 and θ_2 are given by $10^{\bar{\mu}_1}$, $10^{\bar{\phi}_1}$ and $10^{\bar{\phi}_2}$ respectively, such that the optimally preprocessed polynomials for use in the kth subresultant matrix are given by $\lambda_k \tilde{f}_k(\omega_1, \omega_2)$ and $\tilde{g}_k(\omega_1, \omega_2)$.

C.2.4 Preprocessing the Three-Polynomial Subresultant Matrices

In this section the preprocessing of the three-polynomial subresultant is considered, where the three bivariate polynomials are defined over a triangular domain. This extends the work in Appendix C.2.3. The minimisation problem (C.5) is extended, with the additional constraints

$$\begin{aligned} u &\geq \frac{\left|\lambda \bar{a}_{i_1,i_2} \theta_1^{i_1} \theta_2^{i_2} \binom{m}{i_1,i_2}\binom{o-k}{j_1,j_2}\right|}{\binom{m+o-k}{i_1+j_1,i_2+j_2}} \quad i_1+i_2 = 0,\ldots,m; \quad j_1+j_2 = 0,\ldots,o-k, \\ v &\leq \frac{\left|\bar{b}_{i_1,i_2} \theta_1^{i_1} \theta_2^{i_2} \binom{n}{i_1,i_2}\binom{m-k}{j_1,j_2}\right|}{\binom{m+n-k}{i_1+j_1,i_2+j_2}} \quad i_1+i_2 = 0,\ldots,n; \quad j_1+j_2 = 0,\ldots,m-k. \end{aligned} \tag{C.18}$$

By the set of transformations (C.6) and (C.7) and additional transformations $\bar{\mu}_2 = \log_{10}(\rho)$ and

$$\begin{aligned} \bar{\mathcal{C}}_{i_1,i_2,j_1,j_2} &= \log_{10}\left(\frac{\left|\bar{a}_{i_1,i_2}\binom{m}{i_1,i_2}\binom{o-k}{j_1,j_2}\right|}{\binom{m+o-k}{i_1+j_1,i_2+j_2}}\right) \quad i_1+i_2 = 0,\ldots,m; \quad j_1+j_2 = 0,\ldots,o-k, \\ \bar{\mathcal{D}}_{i_1,i_2,j_1,j_2} &= \log_{10}\left(\frac{\left|\bar{c}_{i_1,i_2}\binom{o}{i_1,i_2}\binom{m-k}{j_1,j_2}\right|}{\binom{m+o-k}{i_1+j_1,i_2+j_2}}\right) \quad i_1+i_2 = 0,\ldots,o; \quad j_1+j_2 = 0,\ldots,m-k, \end{aligned}$$

the extended minimisation problem can be written as

$$
\begin{array}{ll}
\text{Minimise} & U - V \\
\text{Subject to} & \\
U - i_1\bar{\phi}_1 - i_2\bar{\phi}_2 - \bar{\mu}_1 & \geq \quad \bar{\alpha}_{i_1,i_2,j_1,j_2} \quad i_1+i_2 = 0,\ldots,m; \quad j_1+j_2 = 0,\ldots,n-k, \\
U - i_1\bar{\phi}_1 - i_2\bar{\phi}_2 & \geq \quad \bar{\beta}_{i_1,i_2,j_1,j_2} \quad i_1+i_2 = 0,\ldots,n; \quad j_1+j_2 = 0,\ldots,m-k, \\
U - i_1\bar{\phi}_1 - i_2\bar{\phi}_2 - \bar{\mu}_1 & \geq \quad \bar{\mathcal{C}}_{i_1,i_2,j_2,j_2} \quad i_1+i_2 = 0,\ldots,m; \quad j_1+j_2 = 0,\ldots,o-k, \\
U - i_1\bar{\phi}_1 - i_2\bar{\phi}_2 - \bar{\mu}_2 & \geq \quad \bar{\mathcal{D}}_{i_1,i_2,j_2,j_2} \quad i_1+i_2 = 0,\ldots,o; \quad j_1+j_2 = 0,\ldots,m-k, \\
-V + i_1\bar{\phi}_1 + i_2\bar{\phi}_2 + \bar{\mu}_1 & \geq -\bar{\alpha}_{i_1,i_2,j_1,j_2} \quad i_1+i_2 = 0,\ldots,m; \quad j_1+j_2 = 0,\ldots,n-k, \\
-V + i_1\bar{\phi}_1 + i_2\bar{\phi}_2 & \geq -\bar{\beta}_{i_1,i_2,j_1,j_2} \quad i_1+i_2 = 0,\ldots,n; \quad j_1+j_2 = 0,\ldots,m-k, \\
-V + i_1\bar{\phi}_1 + i_2\bar{\phi}_2 + \bar{\mu}_1 & \geq -\bar{\mathcal{C}}_{i_1,i_2,j_1,j_2} \quad i_1+i_2 = 0,\ldots,m; \quad j_1+j_2 = 0,\ldots,o-k, \\
-V + i_1\bar{\phi}_1 + i_2\bar{\phi}_2 + \bar{\mu}_2 & \geq -\bar{\mathcal{D}}_{i_1,i_2,j_1,j_2} \quad i_1+i_2 = 0,\ldots,o; \quad j_1+j_2 = 0,\ldots,m-k.
\end{array}
\tag{C.19}
$$

The sets $\bar{\mathcal{M}}_1$, $\bar{\mathcal{M}}_2$, $\bar{m}_1$ and $\bar{m}_2$ are already defined in (C.9, C.10, C.11, C.9) and now $\bar{\mathcal{M}}_3$, $\bar{\mathcal{M}}_4$, $\bar{m}_3$ and $\bar{m}_4$ are given by

$$
\begin{array}{lcll}
\bar{\mathcal{M}}_{3,i_1,i_2} & = & \max_{j_1+j_2=0,\ldots,o-k}\{\bar{\mathcal{C}}_{i_1,i_2,j_1,j_2}\} & \text{for } i_1+i_2 = 0,\ldots,m, \\
\bar{\mathcal{M}}_{4,i_1,i_2} & = & \max_{j_1+j_2=0,\ldots,m-k}\{\bar{\mathcal{D}}_{i_1,i_2,j_1,j_2}\} & \text{for } i_1+i_2 = 0,\ldots,o, \\
\bar{m}_{3,i_1,i_2} & = & \min_{j_1+j_2=0,\ldots,o-k}\{\bar{\mathcal{C}}_{i_1,i_2,j_1,j_2}\} & \text{for } i_1+i_2 = 0,\ldots,m, \\
\bar{m}_{4,i_1,i_2} & = & \min_{j_1+j_2=0,\ldots,m-k}\{\bar{\mathcal{D}}_{i_1,i_2,j_1,j_2}\} & \text{for } i_1+i_2 = 0,\ldots,o,
\end{array}
$$

such that the minimisation problem (C.19) can be written as

$$
\begin{array}{ll}
\text{Minimise} & U - V \\
\text{subject to} & \\
U - i_1\bar{\phi}_1 - i_2\bar{\phi}_2 - \bar{\mu}_1 & \geq \bar{\mathcal{M}}_{1,i_1,i_2} \quad i_1+i_2 = 0,\ldots,m, \\
U - i_1\bar{\phi}_1 - i_2\bar{\phi}_2 & \geq \bar{\mathcal{M}}_{2,i_1,i_2} \quad i_1+i_2 = 0,\ldots,n, \\
U - i_1\bar{\phi}_1 - i_2\bar{\phi}_2 - \bar{\mu}_1 & \geq \bar{\mathcal{M}}_{3,i_1,i_2} \quad i_1+i_2 = 0,\ldots,m, \\
U - i_1\bar{\phi}_1 - i_2\bar{\phi}_2 - \bar{\mu}_2 & \geq \bar{\mathcal{M}}_{4,i_1,i_2} \quad i_1+i_2 = 0,\ldots,o, \\
-V + i_1\bar{\phi}_1 + i_2\bar{\phi}_2 + \bar{\mu}_1 & \geq -\bar{m}_{1,i_1,i_2} \quad i_1+i_2 = 0,\ldots,m, \\
-V + i_1\bar{\phi}_1 + i_2\bar{\phi}_2 & \geq -\bar{m}_{2,i_1,i_2} \quad i_1+i_2 = 0,\ldots,n, \\
-V + i_1\bar{\phi}_1 + i_2\bar{\phi}_2 + \bar{\mu}_1 & \geq -\bar{m}_{3,i_1,i_2} \quad i_1+i_2 = 0,\ldots,m, \\
-V + i_1\bar{\phi}_1 + i_2\bar{\phi}_2 + \bar{\mu}_2 & \geq -\bar{m}_{4,i_1,i_2} \quad i_1+i_2 = 0,\ldots,o.
\end{array}
$$

This can be written in matrix form as

$$
\text{Minimise} \quad \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} U \\ V \\ \bar{\phi}_1 \\ \bar{\phi}_2 \\ \bar{\mu}_1 \\ \bar{\mu}_2 \end{bmatrix} \quad \text{subject to} \quad A \begin{bmatrix} U \\ V \\ \bar{\phi}_1 \\ \bar{\phi}_2 \\ \bar{\mu}_1 \\ \bar{\mu}_2 \end{bmatrix} \geq \mathbf{b}, \tag{C.20}
$$

where the matrix A is given by

$$A = \left[\begin{array}{cccc|cccc} \bar{\mathcal{A}}_1 & \bar{\mathcal{A}}_2 & \bar{\mathcal{A}}_1 & \bar{\mathcal{A}}_3 & \bar{a}_1 & \bar{a}_2 & \bar{a}_1 & \bar{a}_3 \end{array} \right].$$

The rows of matrices $\bar{\mathcal{A}}_1$ and $\bar{a}_1 \in \mathbb{R}^{\binom{m+2}{2} \times 6}$ are given by

$$\bar{\mathcal{A}}_1 = \left[\begin{array}{cccccc} 1 & 0 & -i_1 & -i_2 & -1 & 0 \end{array} \right], \quad \bar{a}_1 = \left[\begin{array}{cccccc} 0 & -1 & i_1 & i_2 & 1 & 0 \end{array} \right],$$

for $i_1 + i_2 = 0, \ldots, m$. The rows of $\bar{\mathcal{A}}_2$, $\bar{a}_2 \in \mathbb{R}^{\binom{n+2}{2} \times 6}$ are given by

$$\bar{\mathcal{A}}_2 = \left[\begin{array}{cccccc} 1 & 0 & -i_1 & -i_2 & 0 & 0 \end{array} \right], \quad \bar{a}_2 = \left[\begin{array}{cccccc} 0 & -1 & i_1 & i_2 & 0 & 0 \end{array} \right],$$

for $i_1 + i_2 = 0, \ldots, n$. The rows of $\bar{\mathcal{A}}_3$, $\bar{a}_3 \in \mathbb{R}^{\binom{o+2}{2} \times 6}$ are given by

$$\bar{\mathcal{A}}_3 = \left[\begin{array}{cccccc} 1 & 0 & -i_1 & -i_2 & 0 & -1 \end{array} \right], \quad \bar{a}_3 = \left[\begin{array}{cccccc} 0 & 1 & i_1 & i_2 & 0 & 1 \end{array} \right].$$

The vector $\mathbf{b}$ in (C.20) is given by

$$\mathbf{b} = \left[\begin{array}{cccc|cccc} \bar{m}_1 & \bar{m}_2 & \bar{m}_3 & \bar{m}_4 & -\bar{\boldsymbol{m}}_1 & -\bar{\boldsymbol{m}}_2 & -\bar{\boldsymbol{m}}_3 & -\bar{\boldsymbol{m}}_4 \end{array} \right]^T.$$

The vectors $\bar{m}_1$, $\bar{m}_2$, $\bar{\boldsymbol{m}}_1$ and $\bar{\boldsymbol{m}}_2$ are already defined in (C.14, C.15, C.16, C.17) and the vectors $\bar{m}_3$, $\bar{m}_4$, $\bar{\boldsymbol{m}}_3$ and $\bar{m}_4$ are similarly defined as

$$\bar{\boldsymbol{m}}_3 = \left[\begin{array}{ccccc} \bar{m}_{3,0,0} & \bar{m}_{3,1,0} & \bar{m}_{3,0,1} & \ldots & \bar{m}_{3,0,m} \end{array} \right] \in \mathbb{R}^{\binom{m+2}{2}}, \tag{C.21}$$

$$\bar{\boldsymbol{m}}_4 = \left[\begin{array}{ccccc} \bar{m}_{4,0,0} & \bar{m}_{4,1,0} & \bar{m}_{4,0,1} & \ldots & \bar{m}_{4,0,o} \end{array} \right] \in \mathbb{R}^{\binom{o+2}{2}}, \tag{C.22}$$

$$\bar{\boldsymbol{m}}_3 = \left[\begin{array}{ccccc} \bar{m}_{3,0,0} & \bar{m}_{3,1,0} & \bar{m}_{3,0,1} & \ldots & \bar{m}_{3,0,m} \end{array} \right] \in \mathbb{R}^{\binom{m+2}{2}}, \tag{C.23}$$

$$\bar{\boldsymbol{m}}_4 = \left[\begin{array}{ccccc} \bar{m}_{4,0,0} & \bar{m}_{4,1,0} & \bar{m}_{4,0,1} & \ldots & \bar{m}_{4,0,o} \end{array} \right] \in \mathbb{R}^{\binom{o+2}{2}}. \tag{C.24}$$

The optimal values λ_k, ρ_k, $\theta_{1,k}$ and $\theta_{2,k}$ are given by 10^{μ_1}, 10^{μ_2}, 10^{ϕ_1} and 10^{ϕ_2}.

C.3 Preprocessing the Subresultant Matrices of Bivariate Polynomials Defined over a Rectangular Domain

C.3.1 The Geometric Mean of the Non-Zero Entries of the $(n-k)$th Order Convolution Matrix

Proposition 7. *The geometric mean $\mathcal{G}_{n_1-k_1,n_2-k_2}(\hat{f}(x,y))$ of the non-zero entries of $C_{n_1-k_1,n_2-k_2}(\hat{f}(x,y))$ is given by*

$$\prod_{j_2=0}^{n_2-k_2} \prod_{j_1=0}^{n_1-k_1} \prod_{i_2=0}^{m_2} \prod_{i_1=0}^{m_1} \left(\hat{a}_{i_1,i_2} \frac{\binom{i_1+j_1}{i_1}\binom{m_1+n_1-k_1-i_1}{n_1-k_1-i_1-j_1}}{\binom{m_1+n_1-k_1}{n_1-k_1}} \frac{\binom{i_2+j_2}{i_2}\binom{m_2+n_2-k_2-i_2}{n_2-k_2-i_2-j_2}}{\binom{m_2+n_2-k_2}{n_2-k_2}} \right)^{\frac{1}{(m_1+1)(m_2+1)(n_1-k_1+1)(n_2-k_2+1)}}$$

and can be simplified to

$$\frac{1}{\binom{m_1+n_1-k_1}{n_1-k_1}\binom{m_2+n_2-k_2}{n_2-k_2}} \times \left(\prod_{i_2=0}^{m_2}\prod_{i_1=0}^{m_1} a_{i_1,i_2}\right)^{\frac{1}{(m_1+1)(m_2+1)}} \times \prod_{j_1=0}^{n_1-k_1}\prod_{i_1=0}^{m_1}\binom{i_1+j_1}{i_1}^{\frac{2}{(m_1+1)(n_1-k_1+1)}} \times \prod_{j_2=0}^{n_2-k_2}\prod_{i_2=0}^{m_2}\binom{i_2+j_2}{j_2}^{\frac{2}{(m_2+1)(n_2-k_2+1)}}.$$

Proof. The geometric mean $\mathcal{G}_{n_1-k_1,n_2-k_2}(\hat{f}(x,y))$ can be considered as the product of four parts

$$\mathcal{A} \times \mathcal{B}_{k_1} \times \mathcal{C}_{k_2} \times \mathcal{D}_{k_1,k_2}.$$

Firstly,

$$\begin{aligned}\mathcal{A} &= \prod_{j_2=0}^{n_2-k_2}\prod_{j_1=0}^{n_1-k_1}\prod_{i_2=0}^{m_2}\prod_{i_1=0}^{m_1}(\hat{a}_{i_1,i_2})^{\frac{1}{(m_1+1)(m_2+1)(n_1-k_1+1)(n_2-k_2+1)}} \\ &= \left(\prod_{i_2=0}^{m_2}\prod_{i_1=0}^{m_1} a_{i_1,i_2}\right)^{\frac{1}{(m_1+1)(m_2+1)}}.\end{aligned}$$

Secondly,

$$\begin{aligned}\mathcal{B}_{k_1} &= \prod_{j_2=0}^{n_2-k_2}\prod_{j_1=0}^{n_1-k_1}\prod_{i_2=0}^{m_2}\prod_{i_1=0}^{m_1}\left(\binom{i_1+j_1}{i_1}\binom{m_1+n_1-k_1-i_1-j_1}{n_1-k_1-j_1}\right)^{\frac{1}{(m_1+1)(m_2+1)(n_1-k_1+1)(n_2-k_2+1)}} \\ &= \prod_{j_1=0}^{n_1-k_1}\prod_{i_1=0}^{m_1}\left(\binom{i_1+j_1}{i_1}\binom{m_1+n_1-k_1-i_1-j_1}{n_1-k_1-j_1}\right)^{\frac{1}{(m_1+1)(n_1-k_1+1)}}\end{aligned}$$

and since

$$\prod_{j_1=0}^{n_1-k_1}\prod_{i_1=0}^{m_1}\binom{i_1+j_1}{i_1} = \prod_{j_1=0}^{n_1-k_1}\prod_{i_1=0}^{m_1}\binom{m_1+n_1-k_1-i_1-j_1}{n_1-k_1-j_1},$$

$\mathcal{B}_{k_1}$ can be written as

$$\mathcal{B}_{k_1} = \prod_{j_1=0}^{n_1-k_1}\prod_{i_1=0}^{m_1}\binom{i_1+j_1}{i_1}^{\frac{2}{(m_1+1)(n_1-k_1+1)}}.$$

Thirdly, by a similar deduction, the geometric mean of the second set of binomials in the numerator can be given by

$$\mathcal{C}_{k_2} = \prod_{j_2=0}^{n_2-k_2}\prod_{i_2=0}^{m_2}\binom{i_2+j_2}{j_2}^{\frac{2}{(m_2+1)(n_2-k_2+1)}}.$$

Finally, the terms in the denominator are constants

$$\begin{aligned}\mathcal{D}_{k_1,k_2} &= \prod_{j_2=0}^{n_2-k_2}\prod_{j_1=0}^{n_1-k_1}\prod_{i_2=0}^{m_2}\prod_{i_1=0}^{m_1}\left(\frac{1}{\binom{m_1+n_1-k_1}{n_1-k_1}\binom{m_2+n_2-k_2}{n_2-k_2}}\right)^{\frac{1}{(m_1+1)(m_2+1)(n_1-k_1+1)(n_2-k_2-1)}} \\ &= \frac{1}{\binom{m_1+n_1-k_1}{n_1-k_1}\binom{m_2+n_2-k_2}{n_2-k_2}}.\end{aligned}$$

□

A similar expression for $\mathcal{G}_{n_1-k_1,n_2-k_2}(\hat{f}(x,y))$ can be derived.

The Arithmetic Mean of the Non-Zero Entries of the $(n-k)$th Order Convolution Matrix

This section considers the fast computation of the arithmetic mean of the non-zero entries in the $(n-k)$th order convolution matrix of the polynomial $\hat{f}(x,y)$. The sum of the non-zero entries in $C_{n-k}(\hat{f}(x,y))$ containing any one coefficient $\hat{a}_{i_1,i_2}$ is given by

$$\sum_{j_2=0}^{n_2-k_2}\sum_{j_1=0}^{n_1-k_1}\hat{a}_{i_1,i_2}\frac{\binom{i_1+j_1}{i_1}\binom{i_2+j_2}{i_2}\binom{m_1+n_1-k_1-(i_1+j_1)}{m_1-i_1}\binom{m_2+n_2-k_2-(i_2+j_2)}{m_2-i_2}}{\binom{m_1+n_1-k_1}{m_1}\binom{m_2+n_2-k_2}{m_2}},$$

and this expression reduces to

$$\hat{a}_{i_1,i_2}\times\frac{m_1+n_1-k_1+1}{m_1+1}\times\frac{m_2+n_2-k_2+1}{m_2+1}.$$

C.3.2 Preprocessing the Two-Polynomial Subresultant Matrices

This section considers the determination of the optimal values of λ, θ_1 and θ_2 such that the ratio of the entry of maximum magnitude to entry of minimum magnitude in $S_{k_1,k_2}(f,g)$ is minimised. This problem arises in Section 7.3, and the values λ_{k_1,k_2}, θ_{1,k_1,k_2} and θ_{2,k_1,k_2} are given by the solution of a linear programming problem. The minimisation problem is written as

$$\begin{array}{lll}\text{Minimise} & \dfrac{u}{v} & \\ \text{subject to} & & \\ u & \geq & \mathcal{P}_{1,k_1,k_2}(\lambda,\theta_1,\theta_2) \quad \begin{array}{ll} i_1=0,\ldots,m_1; & j_1=0,\ldots,n_1-k_1; \\ i_2=0,\ldots,m_2; & j_2=0,\ldots,n_2-k_2, \end{array} \\ u & \geq & \mathcal{P}_{2,k_1,k_2}(\theta_1,\theta_2) \quad \begin{array}{ll} i_1=0,\ldots,n_1; & j_1=0,\ldots,m_1-k_1; \\ i_2=0,\ldots,n_2; & j_2=0,\ldots,n_2-k_2, \end{array} \\ v & \leq & \mathcal{P}_{1,k_1,k_2}(\lambda,\theta_1,\theta_2) \quad \begin{array}{ll} i_1=0,\ldots,m_1; & j_1=0,\ldots,n_1-k_1; \\ i_2=0,\ldots,m_2; & j_2=0,\ldots,n_2-k_2, \end{array} \\ v & \leq & \mathcal{P}_{2,k_1,k_2}(\theta_1,\theta_2) \quad \begin{array}{ll} i_1=0,\ldots,n_1; & j_1=0,\ldots,m_1-k_1; \\ i_2=0,\ldots,n_2; & j_2=0,\ldots,m_2-k_2, \end{array} \\ v & > & 0, \\ \theta_1 & > & 0, \\ \theta_2 & > & 0, \\ \alpha & > & 0. \end{array} \tag{C.25}$$

Given the set of transformations

$$U = \log_{10}(u), \quad V = \log_{10}(v), \quad \bar{\phi}_1 = \log_{10}(\theta_1), \quad \bar{\phi}_2 = \log_{10}(\theta_2), \quad \bar{\mu}_1 = \log_{10}(\lambda) \tag{C.26}$$

and

$$\begin{aligned}
\bar{\alpha}_{i_1,i_2,j_1,j_2} &= \log_{10}\left(\frac{\left|\bar{a}_{i_1,i_2}\binom{m_1}{i_1}\binom{m_2}{i_2}\binom{n_1-k_1}{j_1}\binom{n_2-k_2}{j_2}\right|}{\binom{m_1+n_1-k_1}{i_1+j_1}\binom{m_2+n_2-k_2}{i_2+j_2}}\right) \\
\bar{\beta}_{i_1,i_2,j_1,j_2} &= \log_{10}\left(\frac{\left|\bar{b}_{i_1,i_2}\binom{n_1}{i_1}\binom{n_2}{i_2}\binom{m_1-k_1}{j_1}\binom{m_2-k_2}{j_2}\right|}{\binom{m_1+n_1-k_1}{i_1+j_1}\binom{m_2+n_2-k_2}{i_2+j_2}}\right),
\end{aligned} \tag{C.27}$$

the minimisation problem (C.25) can be written as

Minimise $U - V$

Subject to

$$\begin{array}{rrrrcll}
U & -i_1\bar{\phi}_1 & -i_2\bar{\phi}_2 & -\bar{\mu}_1 & \geq & \bar{\alpha}_{i_1,i_2,j_1,j_2} & i_1 = 0,\ldots,m_1;\quad j_1 = 0,\ldots,n_1-k_1; \\
 & & & & & & i_2 = 0,\ldots,m_2;\quad j_2 = 0,\ldots,n_2-k_2, \\
U & -i_1\bar{\phi}_1 & -i_2\bar{\phi}_2 & & \geq & \bar{\beta}_{i_1,i_2,j_1,j_2} & i_1 = 0,\ldots,n_1;\quad j_1 = 0,\ldots,m_1-k_1; \\
 & & & & & & i_2 = 0,\ldots,n_2;\quad j_2 = 0,\ldots,m_2-k_2, \\
-V & +i_1\bar{\phi}_1 & +i_2\bar{\phi}_2 & +\bar{\mu}_1 & \geq & -\bar{\alpha}_{i_1,i_2,j_1,j_2} & i_1 = 0,\ldots,m_1;\quad j_1 = 0,\ldots,n_1-k_1; \\
 & & & & & & i_2 = 0,\ldots,m_2;\quad j_2 = 0,\ldots,n_2-k_2, \\
-V & +i_1\bar{\phi}_1 & +i_2\bar{\phi}_2 & & \geq & -\bar{\beta}_{i_1,i_2,j_1,j_2} & i_1 = 0,\ldots,n_1;\quad j_1 = 0,\ldots,m_1-k_1; \\
 & & & & & & i_2 = 0,\ldots,n_2;\quad j_2 = 0,\ldots,m_2-k_2.
\end{array}$$

The counters j_1 and j_2 appear on the right hand sides only of these inequalities, and thus if $\bar{m}_{1,i_1,i_2}$, $\bar{m}_{2,i_1,i_2}$, $\bar{m}_{1,i_1,i_2}$ and $\bar{m}_{2,i_1,i_2}$ are defined as

$$\bar{m}_{1,i_1,i_2} = \max_{\substack{j_1=0,\ldots,n_1-k_1 \\ j_2=0,\ldots,n_2-k_2}} \{\alpha_{i_1,i_2,j_1,j_2}\} \quad i_1 = 0,\ldots,m_1, \quad i_2 = 0,\ldots,m_2, \tag{C.28}$$

$$\bar{m}_{2,i_1,i_2} = \max_{\substack{j_1=0,\ldots,m_1-k_1 \\ j_2=0,\ldots,m_2-k_2}} \{\beta_{i_1,i_2,j_1,j_2}\} \quad i_1 = 0,\ldots,n_1, \quad i_2 = 0,\ldots,n_2, \tag{C.29}$$

$$\bar{m}_{1,i_1,i_2} = \min_{\substack{j_1=0,\ldots,n_1-k_1 \\ j_2=0,\ldots,n_2-k_2}} \{\alpha_{i_1,i_2,j_1,j_2}\} \quad i_1 = 0,\ldots,m_1, \quad i_2 = 0,\ldots,m_2, \tag{C.30}$$

$$\bar{m}_{2,i_1,i_2} = \min_{\substack{j_1=0,\ldots,n_1-k_1 \\ j_2=0,\ldots,n_2-k_2}} \{\beta_{i_1,i_2,j_1,j_2}\} \quad i_1 = 0,\ldots,n_1, \quad i_2 = 0,\ldots,n_2, \tag{C.31}$$

the constraints of the minimisation problem above can be written as

$$\begin{array}{rrrrcll}
U & -i_1\bar{\phi}_1 & -i_2\bar{\phi}_2 & -\bar{\mu}_1 & \geq & \bar{m}_{1,i_1,i_2} & i_1 = 0,\ldots,m_1;\quad i_2 = 0,\ldots,m_2, \\
U & -i_1\bar{\phi}_1 & -i_2\bar{\phi}_2 & & \geq & \bar{m}_{2,i_1,i_2} & i_1 = 0,\ldots,n_1;\quad i_2 = 0,\ldots,n_2, \\
-V & +i_1\bar{\phi}_1 & +i_2\bar{\phi}_2 & +\bar{\mu}_1 & \geq & -\bar{m}_{1,i_1,i_2} & i_1 = 0,\ldots,m_1;\quad i_2 = 0,\ldots,m_2, \\
-V & +i_1\bar{\phi}_1 & +i_2\bar{\phi}_2 & & \geq & -\bar{m}_{2,i_1,i_2} & i_1 = 0,\ldots,n_1;\quad i_2 = 0,\ldots,n_2.
\end{array} \tag{C.32}$$

The minimisation in matrix form is given by

$$\text{Minimise} \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} U \\ V \\ \bar{\phi}_1 \\ \bar{\phi}_2 \\ \bar{\mu}_1 \end{bmatrix} \qquad \text{Subject to} \qquad A \begin{bmatrix} U \\ V \\ \bar{\phi}_1 \\ \bar{\phi}_2 \\ \bar{\mu}_1 \end{bmatrix} \geq \mathbf{b}. \tag{C.33}$$

The matrix A has the structure

$$A = \left[\begin{array}{cc|cc} \bar{\mathcal{A}}_1 & \bar{\mathcal{A}}_2 & \bar{a}_1 & \bar{a}_2 \end{array} \right]^T,$$

where the matrices $\bar{\mathcal{A}}_1$ and $\bar{a}_1 \in \mathbb{R}^{(m_1+1)(m_2+1)\times 5}$ are given by

$$\bar{\mathcal{A}}_1 = \left[\begin{array}{ccccc} 1 & 0 & 0 & 0 & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 0 & -m_2 & -1 \\ \hline 1 & 0 & -1 & 0 & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & -1 & -m_2 & -1 \\ \hline \vdots & \vdots & \vdots & \vdots & \vdots \\ \hline 1 & 0 & -m_1 & 0 & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & -m_1 & -m_2 & -1 \end{array} \right], \quad \bar{a}_1 = \left[\begin{array}{ccccc} 0 & -1 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -1 & 0 & m_2 & 1 \\ \hline 0 & -1 & 1 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -1 & 1 & m_2 & 1 \\ \hline \vdots & \vdots & \vdots & \vdots & \vdots \\ \hline 0 & -1 & m_1 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -1 & m_1 & m_2 & 1 \end{array} \right]$$

and the matrices $\bar{\mathcal{A}}_2$ and $\bar{a}_2 \in \mathbb{R}^{(n_1+1)(n_2+1)\times 5}$ are given by

$$\bar{\mathcal{A}}_2 = \left[\begin{array}{ccccc} 1 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 0 & -n_2 & 0 \\ \hline 1 & 0 & -1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & -1 & -n_2 & 0 \\ \hline \vdots & \vdots & \vdots & \vdots & \vdots \\ \hline 1 & 0 & -n_1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & -n_1 & -n_2 & 0 \end{array} \right], \quad \bar{a}_2 = \left[\begin{array}{ccccc} 0 & -1 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -1 & 0 & n_2 & 0 \\ \hline 0 & -1 & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -1 & 1 & n_2 & 0 \\ \hline \vdots & \vdots & \vdots & \vdots & \vdots \\ \hline 0 & -1 & n_1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -1 & n_1 & n_2 & 0 \end{array} \right].$$

The vector $\mathbf{b}$ in (C.33) is given by

$$\mathbf{b} = \left[\begin{array}{cc|cc} \bar{m}_1 & \bar{m}_2 & -\bar{m}_1 & -\bar{m}_2 \end{array} \right]^T,$$

where the vectors $\bar{\boldsymbol{m}}_1$, $\bar{\boldsymbol{m}}_2$, $\bar{\boldsymbol{m}}_1$ and $\bar{\boldsymbol{m}}_2$ are given by

$$\bar{\boldsymbol{m}}_1 = \begin{bmatrix} \bar{m}_{1,0,0} & \dots & \bar{m}_{1,m_1,m_2} \end{bmatrix}^T \in \mathbb{R}^{(m_1+1)(m_2+1)}, \tag{C.34}$$

$$\bar{\boldsymbol{m}}_2 = \begin{bmatrix} \bar{m}_{2,0,0} & \dots & \bar{m}_{2,n_1,n_2} \end{bmatrix}^T \in \mathbb{R}^{(n_1+1)(n_2+1)}, \tag{C.35}$$

$$\bar{\boldsymbol{m}}_1 = \begin{bmatrix} \bar{m}_{1,0,0} & \dots & \bar{m}_{1,n_1,n_2} \end{bmatrix}^T \in \mathbb{R}^{(m_1+1)(m_2+1)}, \tag{C.36}$$

$$\bar{\boldsymbol{m}}_2 = \begin{bmatrix} \bar{m}_{2,0,0} & \dots & \bar{m}_{2,n_1,n_2} \end{bmatrix}^T \in \mathbb{R}^{(n_1+1)(n_2+1)}. \tag{C.37}$$

The optimal values λ_{k_1,k_2}, θ_{1,k_1,k_2} and θ_{2,k_1,k_2} are given by 10^{μ_1}, 10^{ϕ_1} and 10^{ϕ_2} respectively.

C.3.3 Preprocessing the Three-Polynomial Subresultant Matrices

This section considers the preprocessing of the three-polynomial subresultant matrices and extends the minimisation problem in Appendix C.3.2. The minimisation problem (C.25) now has the added constraints

$$\begin{aligned}
u &\geq \mathscr{P}_{3,k_1,k_2}(\lambda,\theta_1,\theta_2) && i_1 = 0,\dots,m_1; \quad j_1 = 0,\dots,o_1-k_1; \\
&&& i_2 = 0,\dots,m_2; \quad j_2 = 0,\dots,o_2-k_2, \\
u &\geq \mathscr{P}_{4,k_1,k_2}(\rho,\theta_1,\theta_2) && i_1 = 0,\dots,o_1; \quad j_1 = 0,\dots,m_1-k_1; \\
&&& i_2 = 0,\dots,o_2; \quad j_2 = 0,\dots,m_2-k_2, \\
v &\leq \mathscr{P}_{3,k_1,k_2}(\lambda,\theta_1,\theta_2) && i_1 = 0,\dots,m_1; \quad j_1 = 0,\dots,o_1-k_1; \\
&&& i_2 = 0,\dots,m_2; \quad j_2 = 0,\dots,o_2-k_2, \\
v &\leq \mathscr{P}_{4,k_1,k_2}(\rho,\theta_1,\theta_2) && i_1 = 0,\dots,o_1; \quad j_1 = 0,\dots,m_1-k_1; \\
&&& i_2 = 0,\dots,o_2; \quad j_2 = 0,\dots,m_2-k_2.
\end{aligned}$$

Consider the same set of transformations in (C.26) and (C.27) with additional transformations $\log_{10}(\rho) = \mu_2$ and

$$\bar{\mathscr{C}}_{i_1,i_2,j_1,j_2} = \log_{10}\left(\frac{\left|\bar{a}_{i_1,i_2}\binom{m_1}{i_1}\binom{m_2}{i_2}\binom{o_1-k_1}{j_1}\binom{o_2-k_2}{j_2}\right|}{\binom{m_1+o_1-k_1}{i_1+j_1}\binom{m_2+o_2-k_2}{i_2+j_2}}\right)$$

$$\bar{\mathscr{D}}_{i_1,i_2,j_1,j_2} = \log_{10}\left(\frac{\left|\bar{c}_{i_1,i_2}\binom{o_1}{i_1}\binom{o_2}{i_2}\binom{m_1-o_1}{j_1}\binom{m_2-o_2}{j_2}\right|}{\binom{m_1+o_1-k_1}{i_1+j_1}\binom{m_2+o_2-k_2}{i_2+j_2}}\right).$$

The extended minimisation problem can therefore be written as

Minimise $U - V$

subject to

$$
\begin{array}{llllll}
U & -i_1\phi_1 & -i_2\phi_2 & -\mu_1 & \geq & \bar{\alpha}_{i_1,i_2,j_1,j_2} \quad i_1 = 0,\ldots,m_1;\ j_1 = 0,\ldots,n_1 - k_1;\ i_2 = 0,\ldots,m_2;\ j_2 = 0,\ldots,n_2 - k_2,\\
U & -i_1\phi_1 & -i_2\phi_2 & & \geq & \bar{\beta}_{i_1,i_2,j_1,j_2} \quad i_1 = 0,\ldots,n_1;\ j_1 = 0,\ldots,m_1 - k_1;\ i_2 = 0,\ldots,n_2;\ j_2 = 0,\ldots,m_2 - k_2,\\
U & -i_1\phi_1 & -i_2\phi_2 & -\mu_1 & \geq & \bar{C}_{i_1,i_2,j_1,j_2} \quad i_1 = 0,\ldots,m_1;\ j_1 = 0,\ldots,o_1 - k_1;\ i_2 = 0,\ldots,m_2;\ j_2 = 0,\ldots,o_2 - k_2,\\
U & -i_1\phi_1 & -i_2\phi_2 & -\mu_2 & \geq & \bar{\mathcal{D}}_{i_1,i_2,j_1,j_2} \quad i_1 = 0,\ldots,o_1;\ j_1 = 0,\ldots,n_1 - k_1;\ i_2 = 0,\ldots,o_2;\ j_2 = 0,\ldots,n_2 - k_2,\\
-V & +i_1\phi_1 & +i_2\phi_2 & +\mu_1 & \geq & -\bar{\alpha}_{i_1,i_2,j_1,j_2} \quad i_1 = 0,\ldots,m_1;\ j_1 = 0,\ldots,n_1 - k_1;\ i_2 = 0,\ldots,m_2;\ j_2 = 0,\ldots,n_2 - k_2,\\
-V & +i_1\phi_1 & +i_2\phi_2 & & \geq & -\bar{\beta}_{i_1,i_2,j_1,j_2} \quad i_1 = 0,\ldots,n_1;\ j_1 = 0,\ldots,m_1 - k_1;\ i_2 = 0,\ldots,n_2;\ j_2 = 0,\ldots,m_2 - k_2,\\
-V & +i_1\phi_1 & +i_2\phi_2 & +\mu_1 & \geq & -\bar{C}_{i_1,i_2,j_1,j_2} \quad i_1 = 0,\ldots,m_1;\ j_1 = 0,\ldots,o_1 - k_1;\ i_2 = 0,\ldots,m_2;\ j_2 = 0,\ldots,o_2 - k_2,\\
-V & +i_1\phi_1 & +i_2\phi_2 & +\mu_2 & \geq & -\bar{\mathcal{D}}_{i_1,i_2,j_1,j_2} \quad i_1 = 0,\ldots,o_1;\ j_1 = 0,\ldots,n_1 - k_1;\ i_2 = 0,\ldots,o_2;\ j_2 = 0,\ldots,n_2 - k_2.
\end{array}
$$

The sets $\{\,\bar{m}_{1,i_1,i_2} \mid i_1 = 0,\ldots,m_1;\ i_2 = 0,\ldots,m_2\,\}$ and $\{\,\bar{m}_{2,i_1,i_2} \mid i_1 = 0,\ldots,n_1;\ i_2 = 0,\ldots,n_2\,\}$ are already defined in (C.28) and (C.29) respectively. Now $\bar{m}_{3,i_1,i_2}$ and $\bar{m}_{4,i_1,i_2}$ are defined as

$$
\begin{aligned}
\bar{m}_{3,i_1,i_2} &= \max\{\,\bar{C}_{i_1,i_2,j_1,j_2} \mid j_1 = 0,\ldots,o_1 - k_1;\ j_2 = 0,\ldots,o_2 - k_2\}\\
&\qquad \text{for} \quad i_1 = 0,\ldots,m_1;\ i_2 = 0,\ldots,m_2\\
\bar{m}_{4,i_1,i_2} &= \max\{\,\bar{\mathcal{D}}_{i_1,i_2,j_1,j_2} \mid j_1 = 0,\ldots,m_1 - k_1;\ j_2 = 0,\ldots,m_2 - k_2\}\\
&\qquad \text{for} \quad i_1 = 0,\ldots,o_1;\ i_2 = 0,\ldots,o_2.
\end{aligned}
\tag{C.38}
$$

The sets $\bar{m}_{1,i_1,i_2}$ and $\bar{m}_{2,i_1,i_2}$ are already defined in (C.30) and (C.31)

$$
\begin{aligned}
\bar{m}_{3,i_1,i_2} &= \min\{\,\bar{C}_{i_1,i_2,j_1,j_2} \mid j_1 = 0,\ldots,o_1 - k_1;\ j_2 = 0,\ldots,o_2 - k_2\}\\
&\qquad \text{for} \quad i_1 = 0,\ldots,m_1;\ i_2 = 0,\ldots,m_2\\
\bar{m}_{3,i_1,i_2} &= \min\{\,\bar{\mathcal{D}}_{i_1,i_2,j_1,j_2} \mid j_1 = 0,\ldots,m_1 - k_1;\ j_2 = 0,\ldots,m_2 - k_2\}\\
&\qquad \text{for} \quad i_1 = 0,\ldots,o_1;\ i_2 = 0,\ldots,o_2.
\end{aligned}
\tag{C.39}
$$

The minimisation problem can now be written as

Minimise $U - V$

subject to (C.40)

$$
\begin{array}{llllll}
U & -i_1\bar{\phi}_1 & -i_2\bar{\phi}_2 & -\bar{\mu}_1 & \geq & \bar{m}_{1,i_1,i_2} \quad i_1 = 0,\ldots,m_1;\ i_2 = 0,\ldots,m_2,\\
U & -i_1\bar{\phi}_1 & -i_2\bar{\phi}_2 & & \geq & \bar{m}_{2,i_1,i_2} \quad i_1 = 0,\ldots,n_1;\ i_2 = 0,\ldots,n_2,\\
U & -i_1\bar{\phi}_1 & -i_2\bar{\phi}_2 & -\mu_1 & \geq & \bar{m}_{3,i_1,i_2} \quad i_1 = 0,\ldots,m_1;\ i_2 = 0,\ldots,m_2,\\
U & -i_1\bar{\phi}_1 & -i_2\bar{\phi}_2 & -\bar{\mu}_2 & \geq & \bar{m}_{4,i_1,i_2} \quad i_1 = 0,\ldots,o_1;\ i_2 = 0,\ldots,o_2,\\
-V & +i_1\bar{\phi}_1 & +i_2\bar{\phi}_2 & +\bar{\mu}_1 & \geq & -\bar{m}_{1,i_1,i_2} \quad i_1 = 0,\ldots,m_1;\ i_2 = 0,\ldots,m_2,\\
-V & +i_1\bar{\phi}_1 & +i_2\bar{\phi}_2 & & \geq & -\bar{m}_{2,i_1,i_2} \quad i_1 = 0,\ldots,n_1;\ i_2 = 0,\ldots,n_2,\\
-V & +i_1\bar{\phi}_1 & +i_2\bar{\phi}_2 & +\bar{\mu}_1 & \geq & -\bar{m}_{3,i_1,i_2} \quad i_1 = 0,\ldots,m_1;\ i_2 = 0,\ldots,m_2,\\
-V & +i_1\bar{\phi}_1 & +i_2\bar{\phi}_2 & +\bar{\mu}_2 & \geq & -\bar{m}_{4,i_1,i_2} \quad i_1 = 0,\ldots,o_1;\ i_2 = 0,\ldots,o_2.
\end{array}
$$

which in matrix form is given by

$$\text{Minimise} \quad \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} U \\ V \\ \bar{\phi}_1 \\ \bar{\phi}_2 \\ \bar{\mu}_1 \\ \bar{\mu}_2 \end{bmatrix} \quad \text{Subject to} \quad A \begin{bmatrix} U \\ V \\ \bar{\phi}_1 \\ \bar{\phi}_2 \\ \bar{\mu}_1 \\ \bar{\mu}_2 \end{bmatrix} \geq \mathbf{b}. \tag{C.41}$$

The matrix A is given by

$$A = \left[\begin{array}{cccc|cccc} \bar{\mathcal{A}}_1 & \bar{\mathcal{A}}_2 & \bar{\mathcal{A}}_1 & \bar{\mathcal{A}}_3 & \bar{a}_1 & \bar{a}_2 & \bar{a}_1 & \bar{a}_3 \end{array} \right]^T,$$

where the matrices $\bar{\mathcal{A}}_1$ and $\bar{a}_1 \in \mathbb{R}^{(m_1+1)(m_2+1)\times 6}$ contain rows of the form

$$\bar{\mathcal{A}}_1 = \begin{bmatrix} 1 & 0 & -i_1 & -i_2 & -1 & 0 \end{bmatrix}, \quad \bar{a}_1 = \begin{bmatrix} 0 & -1 & i_1 & i_2 & 1 & 0 \end{bmatrix}$$

for $i_1 = 0, \ldots, m_1;\ i_2 = 0, \ldots, m_2$, the matrices $\bar{\mathcal{A}}_2$, $\bar{a}_2 \in \mathbb{R}^{(n_1+1)(n_2+1)\times 6}$ contain rows of the form

$$\bar{\mathcal{A}}_2 = \begin{bmatrix} 1 & 0 & -i_1 & -i_2 & 0 & 0 \end{bmatrix}, \quad \bar{a}_2 = \begin{bmatrix} 0 & -1 & i_1 & i_2 & 0 & 0 \end{bmatrix}$$

for $i_1 = 0, \ldots, n_1;\ i_2 = 0, \ldots, n_2$ and the matrices $\bar{\mathcal{A}}_3$, $\bar{a}_3 \in \mathbb{R}^{(o_1+1)(o_2+1)\times 6}$ contain rows of the form

$$\bar{\mathcal{A}}_3 = \begin{bmatrix} 1 & 0 & -i_1 & -i_2 & 0 & -1 \end{bmatrix}, \quad \bar{a}_3 = \begin{bmatrix} 0 & -1 & i_1 & i_2 & 0 & 1 \end{bmatrix}$$

for $i_1 = 0, \ldots, o_1;\ i_2 = 0, \ldots, o_2$.
The vector $\mathbf{b}$ in (C.41) is given by

$$\mathbf{b} = \left[\begin{array}{cccc|cccc} \bar{\boldsymbol{m}}_1 & \bar{\boldsymbol{m}}_2 & \bar{\boldsymbol{m}}_3 & \bar{\boldsymbol{m}}_4 & -\bar{\boldsymbol{m}}_1 & -\bar{\boldsymbol{m}}_2 & -\bar{\boldsymbol{m}}_3 & -\bar{\boldsymbol{m}}_4 \end{array} \right]^T,$$

where $\bar{\boldsymbol{m}}_1$, $\bar{\boldsymbol{m}}_2$, $\bar{\boldsymbol{m}}_1$ and $\bar{\boldsymbol{m}}_2$ are already defined in (C.34, C.35, C.36, C.37) and $\bar{\boldsymbol{m}}_3$, $\bar{\boldsymbol{m}}_4$, $\bar{\boldsymbol{m}}_3$ and $\bar{\boldsymbol{m}}_4$ are given by

$$\bar{\boldsymbol{m}}_3 = \begin{bmatrix} \bar{m}_{1,0,0} & \ldots & \bar{m}_{1,m_1,m_2} \end{bmatrix}^T \in \mathbb{R}^{(m_1+1)(m_2+1)}, \tag{C.42}$$

$$\bar{\boldsymbol{m}}_4 = \begin{bmatrix} \bar{m}_{2,0,0} & \ldots & \bar{m}_{2,n_1,n_2} \end{bmatrix}^T \in \mathbb{R}^{(n_1+1)(n_2+1)}, \tag{C.43}$$

$$\bar{\boldsymbol{m}}_3 = \begin{bmatrix} \bar{m}_{1,0,0} & \ldots & \bar{m}_{1,m_1,m_2} \end{bmatrix}^T \in \mathbb{R}^{(m_1+1)(m_2+1)}, \tag{C.44}$$

$$\bar{\boldsymbol{m}}_4 = \begin{bmatrix} \bar{m}_{2,0,0} & \ldots & \bar{m}_{2,n_1,n_2} \end{bmatrix}^T \in \mathbb{R}^{(n_1+1)(n_2+1)}. \tag{C.45}$$

The optimal values λ_{k_1,k_2}, ρ_{k_1,k_2}, θ_{1,k_1,k_2} and θ_{2,k_1,k_2} are given by 10^{μ_1}, 10^{μ_2}, 10^{ϕ_1} and 10^{ϕ_2}.

Bibliography

[1] Aigner, M., Poteaux, A., and Jüttler, B. *Numerical and Symbolic Scientific Computing*. Springer-Verlag Wien, 2012, ch. Approximate Implicitization of Space Curves, pp. 1–19.

[2] Alkhaldi, N., and Winkler, J. R. Blind image deconvolution using the Sylvester resultant matrix. In *2015 IEEE International Conference on Image Processing (ICIP)* (Sept 2015), IEEE, pp. 784–788.

[3] Barton, M., and Jüttler, B. Computing roots of polynomials by quadratic clipping. *Computer Aided Geometric Design 24*, 3 (Apr. 2007), 125–141.

[4] Berchtold, J., and Bowyer, A. Robust arithmetic for multivariate Bernstein-form polynomials. *Computer-Aided Design 32*, 11 (Sept. 2000), 681–689.

[5] Bini, D. A., and Boito, P. Structured matrix-based methods for polynomial $\in$-gcd: analysis and comparisons. In *Proc. 2007 International Symposium on Symbolic and Algebraic Computation* (2007), ACM Press, pp. 9–16.

[6] Bini, D. A., and Boito, P. *Numerical Methods for Structured Matrices and Applications*. Birkhäuser Basel, 2010, ch. A fast algorithm for approximate polynomial gcd based on structured matrix computations, pp. 155–173.

[7] Bini, D. A., and Gemignani, L. Bernstein-Bezoutian matrices. *Theoretical Computer Science 315*, 2-3 (May 2004), 319–333.

[8] Bini, D. A., Gemignani, L., and Winkler, J. R. Structured matrix methods for CAGD: an application to computing the resultant of polynomials in the Bernstein basis. *Numerical Linear Algebra with Applications 12*, 8 (Oct. 2005), 685–698.

[9] Bourne, M., Winkler, J. R., and Su, Y. The computation of the degree of an approximate greatest common divisor of two Bernstein polynomials. *Applied Numerical Mathematics 111* (Jan. 2017), 17–35.

[10] Bourne, M., Winkler, J. R., and Su, Y. A non-linear structure-preserving matrix method for the computation of the coefficients of an approximate greatest common divisor of two Bernstein polynomials. *Journal of Computational and Applied Mathematics 320* (Aug. 2017), 221–241.

[11] Brent, R. P. An algorithm with guaranteed convergence for finding a zero of a function. *The Computer Journal 14*, 4 (1971), 422–425.

[12] Brown, W. S., and Traub, J. F. On Euclid's algorithm and the theory of subresultants. *Journal of the ACM 18*, 4 (Oct. 1971), 505–514.

[13] Chionh, E.-W., and Goldman, R. N. Using multivariate resultants to find the implicit equation of a rational surface. *The Visual Computer 8*, 3 (May 1992), 171–180.

[14] Cohen, J. S. *Computer Algebra and Symbolic Computation: Elementary Algorithms.* A. K. Peters, Ltd., 2002.

[15] Corless, R. M., Gianni, P. M., Trager, B. M., and Watt, S. M. The singular value decomposition for polynomial systems. In *Proc. 1995 International Symposium on Symbolic and Algebraic Computation* (1995), A. Levelt, Ed., vol. 1, ACM Press, pp. 195–207.

[16] Corless, R. M., Watt, S. M., and Zhi, L. QR factoring to compute the GCD of univariate approximate polynomials. *IEEE Transactions on Signal Processing 52*, 12 (Dec. 2004), 3394–3402.

[17] Daniel, M., and Daubisse, J.-C. The numerical problem of using Bézier curves and surfaces in the power basis. *Computer Aided Geometric Design 6*, 2 (May 1989), 121–128.

[18] de Casteljau, P. Courbes et surfaces à pôles. Tech. rep., Citroën, Paris, 1963.

[19] Dokken, T., and Skytt, V. *Geometric Modelling, Numerical Simulation, and Optimization.* Springer-Verlag, Berlin, 2007, ch. Intersection algorithms and CAGD, pp. 41–90.

[20] Eliás, J., and Zítko, J. Approximate polynomial GCD. In *Programs and Algorithms of Numerical Mathematics 16* (Prague, 2013), J. Chleboun, K. Segeth, J. Šìstek, and T. Vejchodskỳ, Eds., pp. 63–68.

[21] Emiris, I. Z., Galligo, A., and Lombardi, H. Certified approximate univariate GCDs. *Journal of Pure and Applied Algebra 117-118* (May 1997), 229–251.

[22] Farin, G. Triangular Bernstein-Bézier patches. *Computer Aided Geometric Design 3*, 2 (1986), 83–127.

[23] Farin, G. *Curves and Surfaces for Computer Aided Geometric Design*, 4th ed., vol. i. Academic Press, 1997.

[24] Farin, G. E., Hoschek, J., and Kim, M.-S., Eds. *Handbook of Computer Aided Geometric Design.* Elsevier, 2002.

[25] Farouki, R., and Goodman, T. On the optimal stability of the bernstein basis. *Mathematics of Computation of the American Mathematical Society 65*, 216 (1996), 1553–1566.

[26] Farouki, R. T. On the stability of transformations between power and Bernstein polynomial forms. *Computer Aided Geometric Design 8*, 1 (Feb. 1991), 29–36.

[27] Farouki, R. T. The Bernstein polynomial basis: A centennial retrospective. *Computer Aided Geometric Design 29*, 6 (Aug. 2012), 379–419.

[28] Farouki, R. T., and Rajan, V. On the numerical condition of polynomials in bernstein form. *Computer Aided Geometric Design 4*, 3 (1987), 191–216.

[29] Farouki, R. T., and Rajan, V. Algorithms for polynomials in Bernstein form. *Computer Aided Geometric Design 5*, 1 (June 1988), 1–26.

[30] Foster, L. V. Generalizations of Laguerre's method: higher order methods. *SIAM Journal on Numerical Analysis 18*, 6 (1981), 1004–1018.

[31] Gao, S. *Communications, Information and Network Security.* Springer, Boston, MA, 2002, ch. A new algorithm for decoding Reed-Solomon codes, pp. 55–68.

[32] Gao, S., Kaltofen, E., May, J., Yang, Z., and Zhi, L. Approximate factorization of multivariate polynomials via differential equations. In *Proc. 2004 International Symposium on Symbolic and Algebraic Computation* (2004), ACM, New York, pp. 167–174.

[33] Geddes, K. O., Czapor, S. R., and Labahn, G. *Algorithms for Computer Algebra.* Springer, Norwell, MA, 1992.

[34] Glassner, A. S., Ed. *Graphics Gems.* Academic Press, 1990.

[35] Goldman, R. N., Sederberg, T. W., and Anderson, D. Vector elimination : A technique for the intersection of planar parametric rational polynomial curves. *Computer Aided Geometric Design 1*, 4 (Dec. 1984), 327–356.

[36] Golub, G. H., and van Loan, C. F. *Matrix Computations*, 3rd ed. Johns Hopkins University Press, 1996.

[37] Hansen, E., Patrick, M., and Rusnak, J. Some modifications of Laguerre's method. *BIT Numerical Mathematics 17*, 4 (Dec. 1977), 409–417.

[38] Hoschek, J., and Lasser, D. *Fundamentals of Computer Aided Geometric Design.* A. K. Peters, Ltd., Natick, MA, 1993.

[39] Kaltofen, E., Yang, Z., and Zhi, L. Approximate greatest common divisors of several polynomials with linearly constrained coefficients and singular polynomials. In *Proc. 2006 Internation Symposium on Symbolic and Algebraic Computation* (2006), ACM, New York, p. 169.

[40] Kaltofen, E., Yang, Z., and Zhi, L. *Symbolic-Numeric Computation.* Birkhäuser, Basel, 2007, ch. Structured low rank approximation of a Sylvester matrix, pp. 69–83.

[41] Knuth, D. *The Art of Computer Programming : Fundamental Algorithms.* Pearson Education, 1997.

[42] Laidacker, M. Another theorem relating Sylvester's matrix and the greatest common divisor. *Mathematics Magazine 42*, 3 (May 1969), 126–128.

[43] Lane, J. M., and Riesenfeld, R. F. A theoretical development for the computer generation and display of piecewise polynomial surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence 2*, 1 (Jan. 1980), 35–46.

[44] Li, B., Yang, Z., and Zhi, L. Fast Low Rank Approximation of a Sylvester Matrix by Structured Total Least Norm. *J.JSSAC 11*, 3,4 (2005), 165–174.

[45] Madsen, K. A root-finding algorithm based on Newton's method. *BIT Numerical Mathematics 13*, 1 (Mar. 1973), 71–75.

[46] Marsh, D. *Applied Geometry for Computer Graphics and CAD*, 2nd ed. Springer-Verlag, London, 2005.

[47] Mørken, K., and Reimers, M. An unconditionally convergent method for computing zeros of splines and polynomials. *Mathematics of Computation 76*, 258 (Apr. 2007), 845–865.

[48] Moses, J., and Yun, D. Y. Y. The EZ GCD algorithm. In *Proc. 1973 ACM Annual Conference* (1973), ACM, New York, pp. 159–166.

[49] Musser, D. R. *Algorithms for Polynomial Factorization.* PhD thesis, The University of Wisconsin, 1971.

[50] Noda, M.-T., and Sasaki, T. Approximate GCD and its application to ill-conditioned algebraic equations. *Journal of Computational and Applied Mathematics 38*, 1-3 (Dec. 1991), 335–351.

[51] O'Rourke, J. *Computational Geometry in C*, 2nd ed. Cambridge University Press, Cambridge, 1998.

[52] Peters, G., and Wilkinson, J. H. Practical problems arising in the solution of polynomial equations. *IMA Journal of Applied Mathematics 8*, 1 (1971), 16–35.

[53] Pillai, S., and Liang, B. Blind image deconvolution using a robust GCD approach. *IEEE Transactions on Image Processing 8*, 2 (Feb. 1999), 295–301.

[54] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. *Numerical recipes in C*, vol. 2. Cambridge university press Cambridge, 1996.

[55] Rosen, J. B., Park, H., and Glick, J. Structured total least norm for nonlinear problems. *SIAM Journal on Matrix Analysis and Applications 20*, 1 (1998), 14–30.

[56] Schönhage, A. Quasi-gcd computations. *Journal of Complexity 1*, 1 (1985), 118–137.

[57] Sederberg, T. W. Planar piecewise algebraic curves. *Computer Aided Geometric Design 1*, 3 (Dec. 1984), 241–255.

[58] Sederberg, T. W. Applications to computer aided geometric design. In *Proc. Symposia in Applied Mathematics* (1998), vol. 53, pp. 67–89.

[59] Sederberg, T. W., Anderson, D., and Goldman, R. N. Implicit representation of parametric curves and surfaces. *Computer Vision, Graphics, and Image Processing 28*, 1 (Oct. 1984), 72–84.

[60] Sederberg, T. W., and Parry, S. R. Comparison of three curve intersection algorithms. *Computer-Aided Design 18*, 1 (Jan. 1986), 58–63.

[61] Sederberg, T. W., White, S. C., and Zundel, A. K. Fat arcs: A bounding region with cubic convergence. *Computer Aided Geometric Design 6*, 3 (Aug. 1989), 205–218.

[62] Szafnicki, B. On the degree elevation of Bernstein polynomial representation. *Journal of Computational and Applied Mathematics 180*, 2 (Aug. 2005), 443–459.

[63] Uspensky, J. *Theory of Equations*. McGraw-Hill Book Co., 1948.

[64] Winkler, J. R. Polynomial basis conversion made stable by truncated singular value decomposition. *Applied Mathematical Modelling 21*, 9 (Sept. 1997), 557–568.

[65] Winkler, J. R. High order terms for condition estimation of univariate polynomials. *SIAM Journal on Scientific Computing 28*, 4 (2006), 1420–1436.

[66] Winkler, J. R. Structured matrix methods for the computation of multiple roots of a polynomial. *Journal of Computational and Applied Mathematics 272* (Dec. 2014), 449–467.

[67] Winkler, J. R. Polynomial computations for blind image deconvolution. *Linear Algebra and its Applications 502* (Aug. 2016), 77–103.

[68] Winkler, J. R., and Allan, J. D. Structured total least norm and approximate GCDs of inexact polynomials. *Journal of Computational and Applied Mathematics 215*, 1 (May 2008), 1–13.

[69] Winkler, J. R., and Hasan, M. A non-linear structure preserving matrix method for the low rank approximation of the Sylvester resultant matrix. *Journal of Computational and Applied Mathematics 234*, 12 (Oct. 2010), 3226–3242.

[70] Winkler, J. R., and Hasan, M. An improved non-linear method for the computation of a structured low rank approximation of the Sylvester resultant matrix. *Journal of Computational and Applied Mathematics 237*, 1 (Jan. 2013), 253–268.

[71] Winkler, J. R., Hasan, M., and Lao, X. Two methods for the calculation of the degree of an approximate greatest common divisor of two inexact polynomials. *Calcolo 49*, 4 (Dec. 2012), 241–267.

[72] Winkler, J. R., and Lao, X. The calculation of the degree of an approximate greatest common divisor of two polynomials. *Journal of Computational and Applied Mathematics 235*, 6 (Jan. 2011), 1587–1603.

[73] Winkler, J. R., Lao, X., and Hasan, M. The computation of multiple roots of a polynomial. *Journal of Computational and Applied Mathematics 236*, 14 (Aug. 2012), 3478–3497.

[74] Winkler, J. R., and Yang, N. Resultant matrices and the computation of the degree of an approximate greatest common divisor of two inexact Bernstein basis polynomials. *Computer Aided Geometric Design 30*, 4 (May 2013), 410 – 429.

[75] Yang, N. *Structured Matrix Methods for Computations on Bernstein Basis Polynomials*. PhD thesis, The University of Sheffield, 2013.

[76] Yun, D. Y. On square-free decomposition algorithms. In *Proc. 3rd ACM Symposium on Symbolic and Algebraic Computation* (1976), ACM, New York, pp. 26–35.

[77] Zarowski, C. J., Ma, X., and Fairman, F. W. QR-factorization method for computing the greatest common divisor of polynomials with inexact coefficients. *IEEE Transactions on Signal Processing 48*, 11 (Nov. 2000), 3042–3051.

[78] Zeng, Z. Computing multiple roots of inexact polynomials. *Mathematics of Computation 74*, 250 (2005), 869–903.

[79] Zeng, Z., and Dayton, B. H. The approximate GCD of inexact polynomials part ii : a multivariate algorithm. In *Proc. 2004 International Symposium on Symbolic and Algebraic Computation* (2004), pp. 320–327.

[80] Zhi, L., and Yang, Z. Computing approximate GCD of multivariate polynomials by structure total least norm. *MM Research Preprint*, 23 (Dec. 2004), 388–401.

[81] Zippel, R. Probabilistic algorithms for sparse polynomials. In *Symbolic and Algebraic Computation. EUROSAM 1979. Lecture Notes in Computer Science* (1979), E. W. Ng, Ed., vol. 72, Springer, Berlin, pp. 216–226.

[82] Zippel, R. Interpolating polynomials from their values. *Journal of Symbolic Computation 9*, 3 (Mar. 1990), 375 – 403.