

Orthogonally Constrained Sparse Approximations with Applications to Geometry Processing

Sarah Anne Liddell

Submitted in accordance with the requirements
for the degree of Doctor of Philosophy

The University of Leeds



School of Mathematics

September 2019

The candidate confirms that the work submitted is their own and that appropriate credit has been given where reference has been made to the work of others. This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgement. The right of Sarah Anne Liddell to be identified as Author of this work has been asserted by Sarah Anne Liddell in accordance with the Copyright, Designs and Patents Act 1988.

Abstract

Compressed manifold modes are solutions to an optimisation problem involving the ℓ_1 norm and the orthogonality condition $X^T M X = I$. Such functions can be used in geometry processing as a basis for the function space of a mesh and are related to the Laplacian eigenfunctions.

Compressed manifold modes and other alternatives to the Laplacian eigenfunctions are all special cases of generalised manifold harmonics, introduced here as solutions to a more general problem. An important property of the Laplacian eigenfunctions is that they commute with isometry. A definition for isometry between meshes is given and it is proved that compressed manifold modes also commute with isometry. The requirements for generalised manifold harmonics to commute with isometry are explored.

A variety of alternative basis functions are tested for their ability to reconstruct specific functions – it is observed that the function type has more impact than the basis type. The bases are also tested for their ability to reconstruct functions transformed by functional map – it is observed that some bases work better for different shape collections.

The Stiefel manifold is given by the set of matrices $X \in \mathbb{R}^{n \times k}$ such that $X^T M X = I$, with $M = I$. Properties and results are generalised for the $M \neq I$ case. A sequential algorithm for optimisation on the generalised Stiefel manifold is given and applied to the calculation of compressed manifold modes. This involves a smoothing of the ℓ_1 norm.

Laplacian eigenfunctions can be approximated by solving an eigenproblem restricted to a subspace. It is proved that these restricted eigenfunctions also commute with isometry. Finally, a method for the approximation of compressed manifold modes is given. This combines the method of fast approximation of Laplacian eigenfunctions with the ADMM solution to the compressed manifold mode problem. A significant improvement is made to the speed of calculation.

To Bintou, the only person to literally be put to sleep by my
talking about maths.

“We are apt to think of mathematical definitions as too strict and rigid for common use, but their rigour is combined with all but endless freedom. The precise definition of an ellipse introduces us to all the ellipses in the world...” – D’Arcy
Wentworth Thompson [1]

Acknowledgements

The first of my acknowledgements must go to my supervisor, Kevin Houston, whose support, guidance and suggestions have been invaluable. I am most particularly thankful for his patience and ability to gently bolster confidence in times of great doubt.

I am indebted to the people who inspired a passion for problem solving throughout my school years. Most notably my appreciation is to Tracy-from-the-council, for encouraging children into believing they had discovered Euler's polyhedral formula; Mr Taylor, for encouraging an attitude of "But how does it work?" and for the important reminder to an obsessive teenager that there's more to life than maths; Mr Keddie, for preparing the way for university-level mathematics and inspiring a positive no-nonsense approach to teaching.

This thesis is the product of four years spent sitting in a basement so thanks go to all those who have supplied food/diversion/supportive ears/motivational pep talks/prayer. Specifically, Jayne Bird, Tom Bird, Laura Bown, Katherine Edgar, Ann Gilliam, reference-checkers Luke and Rosie Hatton, Freya Howden (and Mike), Helen Hume, Ishbel and Lars Jeuken, Dela Kpeglo and the tower blocks of Little London, Helen Stephenson, Leeds Concert Band – particularly the back-row flutes, members of Thursday Salt Central – past and present, the congregation of West Park URC.

Finally, thanks go to my family: Auntie Kath and Grandma for their quiet but solid support; to my sister Ruth for her genuine interest in what I do and for, more recently, brightening up days with baby photos; and to my parents who have provided all of the above love, and more, in such an unfailing way.

Contents

Abstract	v
Dedication	vii
Quote	ix
Acknowledgements	xi
Contents	xiii
List of Figures	xvii
List of Tables	xxi
Notation	xxiii
1 Introduction	1
2 Basics	4
2.1 Matrix Norms	4
2.2 Matrix Decomposition	5
2.3 Matrix Calculus	8
2.4 Optimisation methods	8
2.5 An Introduction to Discrete Differential Geometry	11

3	Generalised Manifold Harmonics	28
3.1	Background: Supplementary Basis Functions	29
3.2	Background: Compressed Manifold Modes	36
3.3	Generalised Manifold Harmonics	60
3.4	Discrete Isometry	65
3.5	GMHs and Isometry	72
3.6	Summary and Future Work	76
4	A Comparison Of Basis Methods For Reconstructing Functions	77
4.1	Function choice	78
4.2	Function Reconstruction	83
4.3	Basis Choice and Calculation	86
4.4	Mesh Choice and Quality	89
4.5	Function Reconstruction Experiments	91
4.6	Basis Reconstruction	99
4.7	Summary and Future Work	107
5	A Comparison of Basis Methods For Calculating Functional Maps	109
5.1	Background: Functional Maps	110
5.2	Measuring Quality of a Functional Map	125
5.3	Functional map comparison experiments	128
5.4	Summary and Future Work	140

6	Optimisation on Generalised Stiefel Manifolds	141
6.1	Generalised Stiefel Manifolds	142
6.2	Background: Optimisation on Manifolds	153
6.3	Optimisation on Generalised Stiefel Manifolds	158
6.4	Sequential Optimisation on \mathcal{S}_M	160
6.5	Approximating Sequential Compressed Manifold Modes	172
6.6	Summary and Future Work	180
7	Fast Approximation of Compressed Manifold Modes	181
7.1	Fast Approximation of Laplacian Eigenfunctions	181
7.2	Extension to GLMHs	188
7.3	Fast Approximation of Compressed Manifold Modes	189
7.4	Summary and Future Work	200
8	Key Results and Conclusions	201
	Appendices	205
A	Assorted Short Proofs	206
B	Table of Mesh Details	212
C	List of pairs used in matching problem	215
D	Reliability of Compressed Manifold Modes	216
E	Additional figures	218
F	Properties of the projection P	221
G	Proof of the Woodbury Matrix Identity (theorem 6.1.17)	222

Bibliography

223

List of Figures

1.1	Compressed manifold modes	2
2.1	The effect of the number of vertices on the resulting mesh.	15
2.2	Barycentres of triangles	17
2.3	Circumcentres of triangles	17
2.4	A barycentric cell	17
2.5	A Voronoi cell	17
2.6	A mixed Voronoi cell	17
2.7	Construction of the cot Laplacian	22
2.8	Laplacian matrices	24
2.9	Laplacian eigenfunctions	24
3.1	The effect of basis truncation on reconstruction	29
3.2	Localised manifold harmonics	34
3.3	Orthogonality of LMHs	35
3.4	The l_1 norm in comparison to the l_2 norm.	38
3.5	The effect of the sparsity parameter on CMMs.	46
3.6	The soft thresholding operator	52

3.7	Alternative basis types	59
3.8	Choice of weight matrix affects isometry	71
4.1	Extremal and central points on three meshes.	79
4.2	Functions based at a point	81
4.3	Multi-scale functions	82
4.4	Segment indicator functions	83
4.5	Time taken to calculate various basis types	87
4.6	Failure to meet the GLMH orthogonality condition.	89
4.7	Mesh area comparison boxplots	90
4.8	Mesh area uniformity examples	91
4.9	A typical reconstruction error figure.	92
4.10	Reconstruction error when using the graph Laplacian.	93
4.11	Function reconstruction details	94
4.12	Function reconstruction error across all basis types	95
4.13	Function set reconstruction errors by mesh	96
4.14	Average method error	98
4.15	Average method error; scaled with constant function removed	98
4.16	Function reconstruction error boxplots	99
4.17	Distance between function space subspaces	104
4.18	Basis function reconstruction	105
4.19	CMM Sparsity	106
5.1	Functional map matrices	113

5.2	ICP refinement of a functional map	125
5.3	Simple transformation error – the effect of refinement.	131
5.4	Function transformation error – the effect of refinement.	132
5.5	Simple transformation error – comparing functional map direction. . .	134
5.6	Simple transformation error – comparing refined functional map direction.	135
5.7	C -orthogonality error	136
5.8	C -orthogonality error, GL basis removed.	136
5.9	C -orthogonality error (refined)	137
5.10	Geodesic functional map error – the effect of refinement.	139
5.11	Geodesic functional map error – comparing refined functional map direction.	139
6.1	Gradient descent on a manifold	155
6.2	Step size choice	155
6.3	Line search conditions	157
6.4	Projection onto the ellipse via a line through the centre.	168
6.5	Projection onto the ellipse via periodic retraction.	169
6.6	Smoothed approximations of the modulus function.	175
6.7	Smoothed approximations of the modulus function close to zero. . . .	175
6.8	Gradients of smoothed approximations to the modulus function. . . .	176
6.9	The effect of small x values on the difference between $f_{*,\varepsilon}$ and $ x $. . .	176
6.10	Various approximation of CMMs.	177
6.11	Compressed eigenvalues for approximated CMMs	178

6.12	A comparison of <code>eigs</code> eigenfunctions and sequential eigenfunctions . . .	179
7.1	A comparison of eigenvalues and restricted eigenvalues.	185
7.2	A comparison of eigenfunctions and restricted eigenfunctions.	186
7.3	A comparison of <code>eigs</code> and the fast approximation algorithm timings.	187
7.4	The restricted matrices	195
7.5	CMMs calculated via fast approximation and unrestricted ADMM	199
D.1	CMM calculation details	217
E.2	Failure to meet the GLMH orthogonality condition (all data points).	218
E.3	Function transformation error (all data points)	219
E.4	Mesh area comparison boxplots – complete mesh set.	220

List of Tables

6.1	Smoothings of the l_1 norm.	174
7.1	Table comparing ADMM with Fast Approximation for CMMs on assorted meshes.	197
7.2	Table comparing the unrestricted ADMM and fast approximation step times.	198
B.1	Mesh details.	212

Notation

GENERAL

M, X – some matrix.

M_{ij} – the ij element of a matrix M .

v, x – some vector (point).

α – some scalar.

θ – some angle.

e – the number e .

e_i – the standard basis vector, 1 in the i -th entry, 0 elsewhere.

i, j – used as indexes. (Note, no imaginary numbers anywhere...)

I – an identity matrix of the appropriate size.

Λ – a matrix of Lagrange multipliers Λ_{ij} .

$\mathcal{L}(\cdot, \Lambda)$ – a Lagrange multiplier function.

λ_i – the i -th eigenvalue of a matrix.

δ, ε – small, positive numbers.

δ_{ij} – the Kronecker delta.

$\text{tr}(M)$ – the trace of the matrix M .

$\text{vec}(\cdot)$ – the vec operator.

\odot – the Hadamard product.

\otimes – the Kronecker product.

Π – a permutation matrix.

∇ – gradient.

$\text{diag}(x)$ – the diagonal matrix with elements given by vector x .

\mathbb{E}_M – an ellipsoid defined via matrix M .

$\text{Im}(\cdot)$ – the image of a matrix transformation

$\text{Ker}(\cdot)$ – the kernel of a matrix transformation

$\text{dim}(\cdot)$ – the dimension of a vector space

$\text{Rank}(\cdot)$ – the rank of a matrix transformation ($\text{dim}(\text{Im}(\cdot))$)

$\text{Null}(\cdot)$ – the nullity of a matrix transformation ($\text{dim}(\text{Ker}(\cdot))$)

MESHES

\mathcal{N}, \mathcal{P} – (smooth) manifolds.

N, P – meshes, discretisation of the manifolds \mathcal{N}, \mathcal{P} .

n, p – the number of vertices in mesh N, P ; a dimension.

\mathcal{V}_N – the vertex set for mesh N , stored as a $n \times 3$ matrix of points in \mathbb{R}^3 .

v_i – an indexed vertex.

\mathcal{T}_N – the face set for a mesh N , stored as a (number of faces) \times 3 matrix. Each row is a triple of vertex indices.

$\mathcal{V}_1(v_i)$ – the one-ring neighbourhood of a vertex v_i

$\mathcal{F}(N, \mathbb{R})$ – the space of real-valued functions on the mesh N .

\mathbf{f} – a function defined on the vertices, represented by n -dimensional vector.

T – a map between meshes, usually a bijection between vertex sets.

$\text{Adj}(N)$ – the adjacency matrix of the mesh N .

$\text{Deg}(N)$ – the degree matrix of the mesh N .

$\mathcal{A}(\cdot, \cdot)$ – a function used to define an area matrix.

$\mathcal{W}(\cdot, \cdot)$ – a function used to define a weight matrix.

L_N – a Laplacian matrix for mesh N .

A_N, W_N – the area and weight matrices for Laplacian $L_N = A_N^{-1}W_N$.

L, A, W – as above, when it is clear which mesh is being referenced.

$S_\kappa(\cdot)$ – the soft thresholding operator, $\kappa \in \mathbb{R}$.

BASES

ϕ_i, ψ_i – basis functions.

Φ – a matrix with basis functions ϕ_i as columns.

Ψ – a matrix with basis functions ψ_i as columns.

Φ_k – a truncated basis matrix.

k, l – the number of basis functions.

μ_R – a parameter controlling localisation.

μ_\perp – a parameter controlling orthogonality.

μ_f – a parameter controlling weight of a specific function.

NORMS AND INNER PRODUCTS

$\|\cdot\|_F$ – the Frobenius norm.

$\langle \cdot, \cdot \rangle_M$ – the M inner product.

$\|\cdot\|_{F,M}$ – the norm induced by the M inner product, also denoted $\|\cdot\|_M$.

$\|\cdot\|_1$ – the ℓ_1 norm.

$\|\cdot\|_{2,1}$ – the $\ell_{2,1}$ norm.

$\|\cdot\|_2$ – the ℓ_2 norm (for vectors).

$\langle \cdot, \cdot \rangle_c$ – the (generalised) canonical inner product (for Stiefel manifolds).

FUNCTIONAL MAPS

\mathbf{F}, \mathbf{G} – function matrices with indexed functions (as columns).

f, g – the number of functions.

a, b – vectors of coefficients.

A, B, R, S – coefficient matrices, formed by sets of coefficient vectors (as columns).

T_F – a functional map via T .

C – a functional map matrix.

\mathcal{O}_N – an operator on the function space $\mathcal{F}(N, \mathbb{R}^n)$.

$\bar{\mathcal{O}}_N$ – the matrix representation of \mathcal{O}_N .

RECONSTRUCTION

$\delta_r(p)$ – the delta function of radius r at point p .

$HKS_t(x)$ – the heat kernel signature of point x at time t .

$WKS(x, t)$ – the wave kernel signature of point x at time t .

gt – a ground truth map.

Err_f – the reconstruction error function, a vector.

Err – the reconstruction error, a scalar.

Err_S – the simple transformation error.

Err_T – the function transformation error.

Err_{GD} – the geodesic functional map error.

OPTIMISATION ON STIEFEL MANIFOLDS

$\text{Sym}(n)$ – the set of $n \times n$ symmetric matrices.

$\text{sym}(M)$ – projection of $M \in \mathbb{R}^{n \times n}$ into $\text{Sym}(n)$.

$\text{Skew}(n)$ – the set of $n \times n$ skew-symmetric matrices.

$\text{skew}(M)$ – projection of $M \in \mathbb{R}^{n \times n}$ into $\text{Skew}(n)$.

\mathcal{S} – a Stiefel manifold.

\mathcal{S}_M – a generalised Stiefel manifold.

$T_X \mathcal{S}_M$ – the tangent space of \mathcal{S}_M at X .

CT_X – the Cayley transform.

TCT_X – the Cayley retraction.

\mathbb{P} – the set of $n \times n$ positive definite matrices.

\mathbb{L} – the set of $n \times n$ lower triangular matrices with positive diagonal entries.

$\text{Chol}(M)$ – the Cholesky decomposition of matrix M .

R_X^{Chol} – the Cholesky retraction.

FAST APPROXIMATION

d – the sample size.

U – a matrix of locally supported functions.

ABBREVIATIONS

LMHs – localised manifold harmonics.

CMMs – compressed manifold modes.

GMHs – generalised manifold harmonics.

GLMHs – generalised localised manifold harmonics.

Chapter 1

Introduction

Geometry processing applies the ideas and structures of differential geometry to discrete surfaces called meshes. These surfaces can be constructed as a sampling of smooth 2-manifolds. Meshes are widely used in computerised models, arising in animation, medical imaging and computer aided design. Functions on meshes can be represented as an n -dimensional vector where n is the number of known points, called vertices. The number of vertices can be very large, and so, to store information about functions data is compressed, using a truncated basis of the function space of the mesh. Typically, this is a basis of eigenfunctions of a discrete Laplace-Beltrami operator [2],[3],[4],[5].

Laplacian eigenfunctions are minimisers of a discrete Dirichlet energy, and commute with isometry. These functions are also easy to calculate (as solutions to eigenvalue problems are well studied), however alternatives have been suggested:

- Localised manifold harmonics [6];
- Hamiltonian eigenfunctions [7];
- Compressed manifold modes [8],[9].

The first and second of these are also solutions to eigenvalue problems, and are designed to improve reconstruction on specific regions of the mesh. To construct such functions the failure to reconstruct a specific function, for example vertex

positions, is used. Here the question is asked, does this have a negative impact on the ability of the basis to reconstruct other types of function, say geodesic distance functions?

Sparsity is a desirable property when dealing with storing data. Compressed manifold modes are alternatives which are sparse, i.e. the functions are locally supported (see figure 1.1), so can be stored as sparse vectors. They arise as solutions to the orthogonally constrained optimisation problem

$$\arg \min_{\Psi} \text{tr}(\Psi^T W \Psi) + \mu \|A\Psi\|_1 \text{ subject to } \Psi^T A \Psi = I. \quad (1.0.1)$$

This is made difficult to solve by the sparsity inducing term, the ℓ_1 norm. An improved method to construct compressed manifold modes would lead to a rise in their popularity.



Figure 1.1: Compressed manifold modes

The functional maps framework [5] uses truncated bases of Laplacian eigenfunctions and reconstructions of functions to find maps between meshes, relying heavily on the property that they commute with isometry. In existing work the quality of a functional map is measured by extracting a point-to-point match and comparing it to a known match [10],[11],[12],[13]. Given that functional maps allow the transformation of functions without knowledge of a point-to-point match, questions arise about which bases should be used when reconstructing and transforming specific functions.

This work aims to evaluate the properties of alternatives to the Laplacian eigenfunctions and to provide improvement to their calculation. Towards this a new definition of discrete isometry is given (definition 3.4.8), allowing verification that the alternative basis methods commute with isometry. In the same chapter a

general problem is posed which has all of the existing alternative basis methods as special cases.

Following this, in chapters 4 and 5, the alternative basis methods are tested for their ability to reconstruct functions. Two methods of function reconstruction are evaluated, followed by a test of the ability of the basis types to reconstruct functions after transformation by functional map.

The second part of the thesis focuses on the calculation of alternative basis functions. In chapter 6 the ℓ_1 term in problem (1.0.1) is smoothed, and the problem viewed as an optimisation on the generalised Stiefel manifold. Stiefel manifolds are matrix manifolds of the form $X \in \mathbb{R}^{n \times k}$ such that $X^T M X = I$, with $M = I$. Known results about Stiefel manifolds are generalised to the $M \neq I$ case. A sequential algorithm (algorithm 6) for optimisation on the generalised Stiefel manifold is presented and applied to the calculation of compressed manifold modes.

Finally, chapter 7 gives a novel method for approximating compressed manifold modes (algorithm 8). This is based on the method of fast approximation of Laplacian eigenfunctions [14].

Chapter 2

Basics

This chapter provides details of some frequently used definitions and optimisation results. Section 2.5 gives an introduction to discrete differential geometry including the definitions of a mesh and the discrete Laplace-Beltrami operator which will then be used throughout.

2.1 Matrix Norms

The following matrix norms appear in various places.

Definition 2.1.1: The **Frobenius norm** of a matrix $X \in \mathbb{R}^{n \times k}$ is a function $\|\cdot\|_F : \mathbb{R}^{n \times k} \rightarrow \mathbb{R}$ defined by

$$\|X\|_F = \left(\sum_{ij} |X_{ij}|^2 \right)^{\frac{1}{2}} = \text{tr}(X^T X)^{\frac{1}{2}}.$$

Definition 2.1.2: The ℓ_1 **norm** of a matrix $X \in \mathbb{R}^{n \times k}$ is a function $\|\cdot\|_1 : \mathbb{R}^{n \times k} \rightarrow \mathbb{R}$ defined by

$$\|X\|_1 = \sum_{ij} |X_{ij}|.$$

Definition 2.1.3: The $\ell_{2,1}$ **norm** of a matrix $X \in \mathbb{R}^{n \times k}$ is a function $\|\cdot\|_{2,1} : \mathbb{R}^{n \times k} \rightarrow \mathbb{R}$ defined by

$$\|X\|_{2,1} = \sum_i \|x_i^T\|_2.$$

This is the ℓ_1 norm of the vector of ℓ_2 norms of columns x_i of X .

Definition 2.1.4: Given symmetric positive definite matrix $M \in \mathbb{R}^{n \times n}$ the **M norm** of a matrix $X \in \mathbb{R}^{n \times k}$ is a function $\|\cdot\|_{F,M} : \mathbb{R}^{n \times k} \rightarrow \mathbb{R}$ defined by

$$\|X\|_{F,M} = \text{tr}(X^T M X)^{\frac{1}{2}}.$$

When there is no chance for confusion the shorthand $\|\cdot\|_M$ is used for the M norm. The M norm arises as the induced norm of the M inner product.

Definition 2.1.5: Let $X, Y \in \mathbb{R}^{n \times k}$. The **M inner product** is an inner product $\langle \cdot, \cdot \rangle_M : \mathbb{R}^{n \times k} \times \mathbb{R}^{n \times k} \rightarrow \mathbb{R}$ defined by

$$\langle X, Y \rangle_M := \text{tr}(X^T M Y),$$

where M is an $n \times n$ symmetric positive definite matrix.

2.2 Matrix Decomposition

This section details some methods of decomposing matrices into products of matrices with specific properties.

The Singular Value Decomposition

Theorem 2.2.1: [15, 7.3.5],[16, chapter 7] Let M be any $n \times k$ real matrix. Then $M = U \Sigma V^T$ where $U \in \mathbb{R}^{n \times n}$ such that $U^T U = I$, $V \in \mathbb{R}^{k \times k}$ such that $V^T V = I$ and $\Sigma \in \mathbb{R}^{n \times k}$ such that Σ is formed by a $l \times l$ diagonal matrix and a rectangular block of zeros, where $l = \min\{n, k\}$.

Definition 2.2.2: The decomposition of a matrix M via theorem 2.2.1 is called the **singular value decomposition (svd)**. The diagonal entries σ_i of Σ are called the **singular values** of M .

Usually the svd is chosen such that $\sigma_1 \geq \dots \geq \sigma_l$. Note that any singular value decomposition $M = U\Sigma V^T$ can be written in this way. Let Π_n be a permutation matrix which reorders the rows of Σ in an appropriate way, and let Π_k be a permutation matrix which reorders the columns of Σ in an appropriate way. Then $M = \bar{U}\bar{\Sigma}\bar{V}^T$ with $\bar{U} = U\Pi_n^T$, $\bar{\Sigma} = \Pi_n\Sigma\Pi_k$ and $\bar{V} = V\Pi_k^T$.

The QR Decomposition

Theorem 2.2.3: [15, 2.6.1] Let M be an $n \times k$ matrix with $n \geq k$. Then there is an $n \times k$ matrix Q such that $Q^T Q = I$ and a $k \times k$ upper triangular matrix R such that $M = QR$.

Definition 2.2.4: The decomposition of a matrix M via theorem 2.2.3 is called the **QR decomposition** of M .

The Cholesky Decomposition

Theorem 2.2.5: [15, 7.2.9] Let M be an $n \times n$ symmetric positive definite real matrix. Then there exists a unique lower triangular matrix L with positive diagonal values such that $M = LL^T$.

Definition 2.2.6: The **Cholesky decomposition** (or Cholesky matrix) of a symmetric positive definite matrix M is the unique lower triangular matrix L with positive diagonal values such that $M = LL^T$. Let $\text{Chol} : \mathbb{P} \rightarrow \mathbb{L}$ denote the map defined by $\text{Chol}(M) = L$.

Proposition 2.2.7: Let M be a real symmetric matrix, then the entries of the Cholesky matrix L can be defined via the following formulae:

- $L_{jj} = \sqrt{M_{jj} - \sum_{k=1}^{j-1} (L_{jk})^2}$,
- $L_{ij} = \frac{1}{L_{jj}} \left(M_{ij} - \sum_{k=1}^{j-1} L_{ik} L_{jk} \right)$ for $i > j$,
- $L_{ij} = 0$ for $i < j$.

An algorithm for calculating a Cholesky decomposition is given in algorithm 1.

Algorithm 1 The Cholesky-Crout algorithm [17, section 51.4],[18]

- 1: Given $n \times n$ real symmetric positive definite M
 - 2: Set L as an $n \times n$ zero matrix then
 - 3: **for** $j = 1, \dots, n$ **do**
 - 4: $L_{jj} = \sqrt{M_{jj} - \sum_{k=1}^{j-1} (L_{jk})^2}$
 - 5: **for** $i = j + 1, \dots, n$ **do**
 - 6: $L_{ij} = \frac{1}{L_{jj}} \left(M_{ij} - \sum_{k=1}^{j-1} L_{ik} L_{jk} \right)$
 - 7: **end for**
 - 8: **end for**
-

Example

$$\begin{aligned}
 M &= \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \\
 &= \begin{bmatrix} \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}, \text{ the eigendecomposition,} \\
 &= \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix}, \text{ the svd,} \\
 &= \begin{bmatrix} \frac{2}{\sqrt{5}} & \frac{-1}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} & \frac{2}{\sqrt{5}} \end{bmatrix} \begin{bmatrix} \sqrt{5} & \frac{4}{\sqrt{5}} \\ 0 & \frac{3}{\sqrt{5}} \end{bmatrix}, \text{ the QR decomposition,} \\
 &= \begin{bmatrix} \sqrt{2} & 0 \\ \frac{1}{\sqrt{2}} & \frac{\sqrt{3}}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \sqrt{2} & \frac{1}{\sqrt{2}} \\ 0 & \frac{\sqrt{3}}{\sqrt{2}} \end{bmatrix}, \text{ via the Cholesky decomposition.}
 \end{aligned}$$

2.3 Matrix Calculus

Often the aim is to minimise a function $F : \mathbb{R}^{n \times k} \rightarrow \mathbb{R}$. Let $X \in \mathbb{R}^{n \times k}$ then when $F(X)$ can be written as a polynomial in the elements of X , the derivative of F is given by

$$\frac{\partial F(X)}{\partial X} = \begin{bmatrix} \frac{\partial F(X)}{\partial X_{11}} & \cdots & \frac{\partial F(X)}{\partial X_{1k}} \\ \vdots & \ddots & \vdots \\ \frac{\partial F(X)}{\partial X_{n1}} & \cdots & \frac{\partial F(X)}{\partial X_{nk}} \end{bmatrix}.$$

The following derivatives of the trace function can be found listed in [19, section 2.5] and are used without reference in later chapters:

$$\begin{aligned} \frac{\partial}{\partial X} \operatorname{tr}(AXB) &= A^T B^T, [19, (101)], \\ \frac{\partial}{\partial X} \operatorname{tr}(AXBX^T C) &= A^T C^T X B^T + C A X B, [19, (118)]. \end{aligned}$$

2.4 Optimisation methods

Various optimisation methods are used to tackle problems, primarily the method of Lagrange multipliers as referenced in the proof of proposition 2.5.33. Some of the other frequently used optimisation methods and results are summarised here.

2.4.1 Least Squares Minimisation

Let $A \in \mathbb{R}^{n \times a}$ and let $B \in \mathbb{R}^{n \times b}$. Least squares minimisation gives a way of finding a solution $X \in \mathbb{R}^{a \times b}$ to the over-determined system $AX = B$ which minimises the difference between AX and B .

Theorem 2.4.1: [16, via Theorem 4.1] Let $A \in \mathbb{R}^{n \times a}$ with $n > a$, A full rank, and let $B \in \mathbb{R}^{n \times b}$. Then the solution to

$$\arg \min_{X \in \mathbb{R}^{a \times b}} \|B - AX\|_F$$

is given by

$$X = (A^T A)^{-1} A^T B.$$

Proof. First note that

$$\begin{aligned} & \arg \min_X \|B - AX\|_F \\ &= \arg \min_X \|B - AX\|_F^2 \\ &= \arg \min_X \operatorname{tr}(B^T B) + \operatorname{tr}(X^T A^T A X) - 2 \operatorname{tr}(B^T A X). \end{aligned}$$

Differentiating with respect to X and equating with zero gives

$$0 = 2A^T A X - 2A^T B$$

and so

$$A^T A X = A^T B.$$

When A is full rank $A^T A$ is invertible (see lemma A.1) and hence

$$X = (A^T A)^{-1} A^T B.$$

□

2.4.2 ADMM

The alternating direction method of multipliers (ADMM)[20] can be used to solve problems of the form

$$\min f(X) + g(Z) \quad \text{subject to } AX + BZ = C, \quad (2.4.1)$$

where $X \in \mathbb{R}^{a \times m}$, $Z \in \mathbb{R}^{b \times m}$, $A \in \mathbb{R}^{n \times a}$, $B \in \mathbb{R}^{n \times b}$, $C \in \mathbb{R}^{n \times m}$, $f : \mathbb{R}^{a \times m} \rightarrow \mathbb{R}$ and $g : \mathbb{R}^{b \times m} \rightarrow \mathbb{R}$. The ADMM algorithm splits the problem into two separate optimisation problems which may have closed forms – solutions which can be evaluated in a finite number of calculations. The subproblems are iterated between until some convergence tolerance is reached. Algorithm 2 provides the general (scaled) ADMM algorithm.

Algorithm 2 ADMM

- 1: Given functions $f : \mathbb{R}^{a \times m} \rightarrow \mathbb{R}$, $g : \mathbb{R}^{b \times m} \rightarrow \mathbb{R}$, linear constraint $AX + BZ = C$
- 2: Set initial values $X_0 \in \mathbb{R}^{a \times m}$, $Z_0 \in \mathbb{R}^{b \times m}$, $U_0 = 0 \in \mathbb{R}^{n \times m}$
- 3: Set regularisation parameter $\rho > 0$
- 4: **repeat**

$$\begin{aligned}
 X_{k+1} &\leftarrow \arg \min_X f(X) + \frac{\rho}{2} \|AX + BZ_k - C + U_k\|_F^2 & (2.4.2) \\
 Z_{k+1} &\leftarrow \arg \min_Z g(Z) + \frac{\rho}{2} \|AX_{k+1} + BZ - C + U_k\|_F^2 \\
 U_{k+1} &\leftarrow U_k + AX_{k+1} + BZ_{k+1} - C
 \end{aligned}$$

- 5: **until** convergence
-

The number of iterations required for the algorithm to reach convergence can be improved by changing the regularisation parameter ρ in each iteration (see [20, 3.4.1],[21]). The convergence conditions are constructed by evaluating the following quantities:

$$\begin{aligned}
 r_k &:= AX_k + BZ_k - C \\
 s_k &:= \rho A^T B(Z_k - Z_{k-1}).
 \end{aligned}$$

The value $\|r_k\|_F$ measures how well the linear condition is met and is minimised when the linear condition is satisfied, i.e. when $\|r_k\|_F = 0$. Given that X^{k+1} minimises equation (2.4.2), s_k is the value of the derivative of the objective function evaluated at X^{k+1} . A minimum of the objective function is found when the derivative is equal to zero, and so $\|s_k\|_F$ must also be very small for convergence. Therefore, algorithm 2 converges when

$$\|r_k\|_F \leq \varepsilon_{\text{pri}} \quad \text{and} \quad \|s_k\|_F \leq \varepsilon_{\text{dual}},$$

where

$$\begin{aligned}\varepsilon_{\text{pri}} &:= \sqrt{nm}\varepsilon_{\text{abs}} + \varepsilon_{\text{rel}} \max\{\|AX_k\|_2, \|BZ_k\|_2, \|C\|_2\}, \\ \varepsilon_{\text{dual}} &:= \sqrt{am}\varepsilon_{\text{abs}} + \varepsilon_{\text{rel}}\|A^T \rho U_k\|_2, \\ \varepsilon_{\text{abs}} &> 0 \text{ is an absolute tolerance,} \\ \varepsilon_{\text{rel}} &> 0 \text{ is a relative tolerance, [20, 3.3.1].}\end{aligned}$$

A convergence guarantee for a nonconvex and nonsmooth objective function is given in [22].

2.5 An Introduction to Discrete Differential Geometry

Discrete differential geometry aims to use the well-understood foundation of smooth differential geometry and construct analogies to the discrete case. The major area of application is computer science – animation, computer vision, medical imaging software etc. – as computers cannot cope with the infinite nature of smooth results. The research seeks to provide mathematical solutions to the problems computers have with identifying, transforming and reconstructing shapes and images. Much work focuses on discretised surfaces, or *meshes*, as motivated by prevalence of the computer-based applications mentioned above, and is typically referred to as *geometry processing*.

An introduction to geometry processing, beginning with closed simple planar curves, is given in [23]. A more detailed introduction, focusing on handling individual meshes can be found in [24].

Definition 2.5.1: A **vertex** v is a point $v \in \mathbb{R}^3$. A **face** f in \mathbb{R}^3 is a 3-tuple of vertices, $f = (v_1, v_2, v_3)$. An **edge** e in \mathbb{R}^3 is a straight-line segment between two vertices of a face. That is, for a face $f = (v_1, v_2, v_3)$ there exists an edge between every pair of vertices.

A **mesh** $N(\mathcal{V}_N, \mathcal{T}_N)$ in \mathbb{R}^3 is a collection of vertices and faces, such that all vertices are contained in the set \mathcal{V}_N , and all faces are contained in the set \mathcal{T}_N (where faces are constructed from the vertices of \mathcal{V}_N). Typically the mesh is denoted by just N and the subscript on vertex and face sets suppressed unless necessary for distinguishing between meshes. That is, for ease of notation, when meshes are denoted N , a vertex $v \in N$ means $v \in \mathcal{V}_N$ and a face $f \in N$ means $f \in \mathcal{T}_N$.

Definition 2.5.2: A mesh $N(\mathcal{V}_N, \mathcal{T}_N)$ is **connected** if all vertices are contained in at least one face and there does not exist a partition $N(\mathcal{V}_N, \mathcal{T}_N) = N_1(\mathcal{V}_{N_1}, \mathcal{T}_{N_1}) \sqcup N_2(\mathcal{V}_{N_2}, \mathcal{T}_{N_2})$. Note that a set of edges can be constructed from the vertex and face sets via a simple searching and listing process.

Definition 2.5.3: The **one-ring neighbourhood** of a vertex v_i , denoted by $\mathcal{V}_1(v_i)$ is the set of vertices v_j such that there is an edge between v_i and v_j .

Definition 2.5.4: A **path** is a sequence of edges which join a sequence of distinct vertices. Such a sequence of vertices is called the **vertex sequence** of the path.

Definition 2.5.5: A vertex v_i is **non-singular** if for every pair vertices $x, y \in \mathcal{V}_1(v_i)$ there exists a path between x and y such that all vertices within the vertex sequence of the path lie in $\mathcal{V}_1(v_i)$.

Definition 2.5.6: A mesh is **simple** if

- (i) an edge exists between at most two faces (no non-manifold edges),
- (ii) no face intersects any other part of the mesh (no self intersection),
- (iii) all vertices are non-singular.

Recall that an edge exists between pairs of vertices in a face so the first condition means that no pair of vertices appears in more than two faces.

Definition 2.5.7: A mesh is **closed** if all edges exist between exactly two faces. That is, there is no boundary edge.

A connected simple closed mesh is homeomorphic to a triangulation of a sphere with handlebodies. Note that by the above definition meshes must have triangular faces, and this is assumed throughout, however there are occasions when alternative polygonal faces are preferred. For example, meshes with quadrilateral faces are used in animation. This is because they can be aligned with principal curvature directions and angles in quadrilaterals are less adversely affected by stretching in certain directions [25].

There are a variety of lines of research into meshes and their construction:

- **SMOOTHING:** reducing noise or minimising change in some energy on the mesh [26],[27],[28].
- **PARAMETRISATION:** equipping a mesh with coordinates, e.g. for representation in specific software packages, for mapping texture (colour) to the mesh [29],[30],[31].
- **REMESHING:** improving quality for a specific application, e.g. quad-dominant meshes for animation, meshes with vertices of equal valence, [32],[33],[34].
- **SIMPLIFICATION/APPROXIMATION:** constructing a new mesh with fewer vertices, faces and edges than the original, but preserving specific properties as well as possible, [35],[36],[37]
- **MODEL REPAIR:** from a computer model (e.g. generated via CAD) producing a mesh which is closed, has no intersecting faces or overlaps etc [38],[39],[40].
- **DEFORMATION:** ensuring that methods of user-controlled deformation, e.g. via clicking and dragging, behave in a realistic or appropriate way, [41],[42],[43].

In applications, meshes can originate from CAD programs or as scans of physical objects but to allow the transfer of theory from differential geometry it is assumed that a mesh is constructed with reference to an underlying manifold (hence the conditions about self-intersection).

Let \mathcal{N} be a smooth 2-manifold without boundary, smoothly embedded into \mathbb{R}^3 via isometric embedding $\gamma : \mathcal{N} \rightarrow \mathbb{R}^3$. Then the manifold \mathcal{N} can be discretised to a mesh N by defining \mathcal{V}_N as a set of n points $v_i \in \mathbb{R}^3$ from a set of n points $p_i \in \mathcal{N}$

such that $v_i = \gamma(p_i)$. Edges and faces can be defined as above, with reference to the vertex set.

In practice, the vertices of a mesh N constructed from an underlying manifold \mathcal{N} may not lie exactly on \mathcal{N} . This is in part due to the choice of vertex set depending on the application. Some applications only require approximations of the underlying shape, e.g. shape packing problems which given an initial mesh then use a mesh with a reduced number of vertices to accelerate algorithms [44]. Also, depending on the application, care must be taken to preserve the topology.

Many meshes are constructed by sampling a surface and it may be that a mesh constructed via a small number of vertices and faces does not have the same genus as a mesh constructed from the same underlying surface but with a much larger number of vertices and faces. Figure 2.1 shows this via a re-meshing of the victoria17 mesh. The original mesh is shown on the left and a re-meshing via poisson surface reconstruction [45] (executed in MeshLab) is shown on the right. The number of vertices and faces are listed in the figure. Note the loss of the hole formed by the left arm in the re-meshing.

Remark 2.5.8: From here it is assumed that meshes are connected, closed and simple.

Let N be a mesh with vertices v_1, \dots, v_n , then a real-valued function $f : N \rightarrow \mathbb{R}$ can be represented as a vector $\mathbf{f} \in \mathbb{R}^n$ such that the elements of \mathbf{f} are defined by

$$\mathbf{f}_i = f(v_i).$$

The space of all real-valued functions on N is denoted by $\mathcal{F}(N, \mathbb{R})$. Since functions can be described by n -dimensional vectors the function space $\mathcal{F}(N, \mathbb{R})$ is isomorphic to \mathbb{R}^n . Therefore, any basis for \mathbb{R}^n also provides a basis for $\mathcal{F}(N, \mathbb{R})$.

2.5.1 Function space approximation

Let $\{\phi_i\}$ be an ordered basis for the function space $\mathcal{F}(N, \mathbb{R})$, represented as columns of an $n \times n$ matrix Φ . Denote by Φ_k the matrix with the first k basis functions as

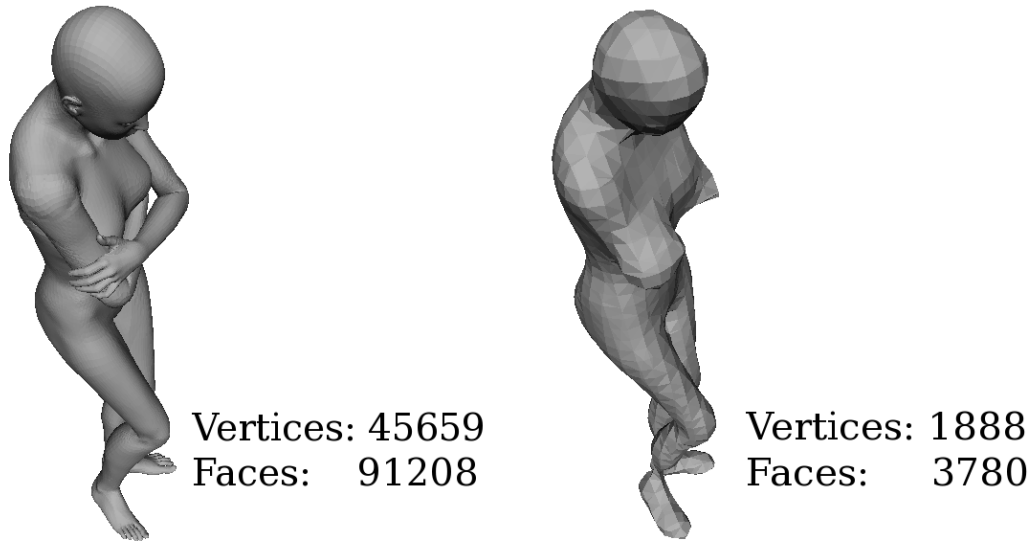


Figure 2.1: The effect of the number of vertices on the resulting mesh.

columns. This represents some truncation of the basis, and

$$\mathcal{F}_k(N, \mathbb{R}) := \text{span}\{\phi_1, \dots, \phi_k\}$$

is a linear subspace of $\mathcal{F}(N, \mathbb{R})$.

Any function $f \in \mathcal{F}(N, \mathbb{R})$ written in vector form as $\mathbf{f} = \Phi a$ can be projected into $\mathcal{F}_k(N, \mathbb{R})$, where $a \in \mathbb{R}^n$ is a vector of coefficients. In vector form the projected function $\bar{\mathbf{f}}$ is given by

$$\bar{\mathbf{f}} = \Phi \begin{bmatrix} I_k & 0 \\ 0 & 0 \end{bmatrix} a.$$

That is, $\bar{\mathbf{f}} = \Phi_k \bar{a}$ where $\bar{a} \in \mathbb{R}^k$ with elements $\bar{a}_i = a_i$.

2.5.2 The discrete Laplace-Beltrami operator

An important operator in geometry processing is a discretisation of the Laplace-Beltrami operator. A list of desirable properties for a discrete Laplace-Beltrami operator is given in [46], along with a proof that there does not exist an operator

which satisfies them all simultaneously for all meshes. This means that there is great flexibility in the definition and construction of a discrete Laplace-Beltrami operator. Hence, here definition is via the product of two symmetric matrices, generated via two functions, associating numbers to each vertex and to each pair of vertices respectively.

A mesh N with n vertices can be considered as a graph with vertex set \mathcal{V}_N and edge set defined via the faces of N . Recall the following definitions from graph theory:

Definition 2.5.9: The **degree matrix** $\text{Deg}(N)$ is the $n \times n$ diagonal matrix where the i -th diagonal element is equal to the number of neighbours of the vertex v_i .

Definition 2.5.10: The **adjacency matrix** $\text{Adj}(N)$ is the $n \times n$ matrix with entries given by

$$\text{Adj}(N)_{ij} = \begin{cases} 0, & \text{if } j = i, \\ 1, & \text{if } j \neq i \text{ and vertices } v_i, v_j \text{ share an edge,} \\ 1, & \text{if } j \neq i \text{ and vertices } v_i, v_j \text{ do not share an edge.} \end{cases}$$

Each vertex in a mesh has an area associated to it. This area can be chosen in a variety of ways, and allows a discrete integral to be defined. The areas are stored as elements of a matrix, for ease of calculation. One way of associating an area to a vertex is construct a cell via properties of the adjacent faces.

Definition 2.5.11: The **barycentre** of a triangle is the point of intersection of the straight lines connecting vertices and mid-points of opposing edges. (See figure 2.2.)

Definition 2.5.12: The **circumcentre** of a triangle is the point of intersection of the perpendicular bisectors of each edge. (See figure 2.3.)

Note that the barycentre of a triangle always lies in the interior of the triangle, but the circumcentre lies outside of the triangle for triangles with an obtuse angle.

Definition 2.5.13: The **barycentric cell** of a vertex v_i is the set of points bounded by the straight lines connecting the barycentres and edge mid-points of the triangle with v_i as a vertex. (See figure 2.4.)

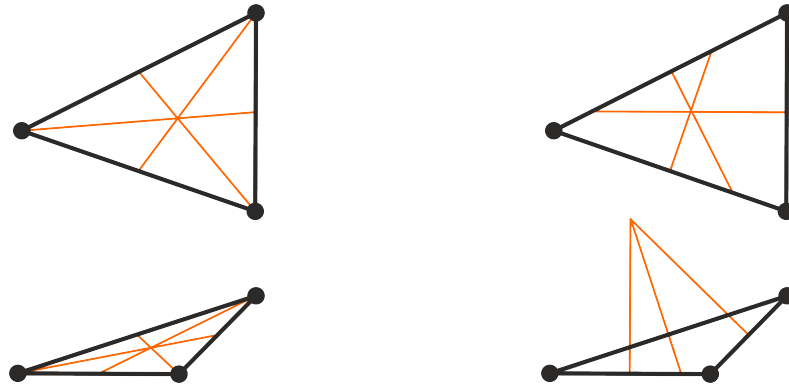


Figure 2.2: Barycentres of triangles **Figure 2.3:** Circumcentres of triangles

Definition 2.5.14: The **Voronoi cell** of a vertex v_i is the set of points bounded by the straight lines connecting the circumcentres and edge mid-points of the triangle with v_i as a vertex. (See figure 2.5.)

Definition 2.5.15: The **mixed Voronoi cell** of a vertex v_i is a Voronoi cell constructed by replacing circumcentres which lie outside of obtuse-angled triangles with the mid-point of the edge opposing the vertex v_i . (See figure 2.6.)



Figure 2.4: A barycentric cell **Figure 2.5:** A Voronoi cell **Figure 2.6:** A mixed Voronoi cell

Definition 2.5.16: Let N be a mesh with vertex set \mathcal{V} and face set \mathcal{T} . An **area matrix** A is an $n \times n$ symmetric positive-definite matrix with entries defined via a function $\mathcal{A} : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}$, such that

$$A_{ij} = \mathcal{A}(v_i, v_j).$$

Examples 2.5.17:

(i) Let $\mathcal{A} : \mathcal{V} \times \mathcal{V} \rightarrow [0, \infty)$ be defined via $\mathcal{A}(v_i, v_j) = r\delta_{ij}$ with $r \in \mathbb{R}^+$. Then $A = rI_n$. That is, each vertex is weighted with an equal area. This is called the **uniform area matrix**.

(ii) Let $\mathcal{A} : \mathcal{V} \times \mathcal{V} \rightarrow [0, \infty)$ be defined via

$$\mathcal{A}(v_i, v_j) = \begin{cases} \frac{1}{3} \sum_{t \in \mathcal{T}(i)} \text{area}(t), & \text{if } i = j, \\ 0, & \text{otherwise} \end{cases}$$

where $\mathcal{T}(i)$ denotes the set of faces with v_i as a vertex and $\text{area}(t)$ denotes the area of the face t . Then A is the diagonal matrix where the i -th diagonal entry is an area associated to the vertex v_i .

(iii) Let $\mathcal{A} : \mathcal{V} \times \mathcal{V} \rightarrow [0, \infty)$ be defined via

$$\mathcal{A}(v_i, v_j) = \begin{cases} \text{area}(\text{cell}(v_i)), & \text{if } i = j, \\ 0, & \text{otherwise} \end{cases}$$

where $\text{cell}(v_i)$ denotes any one of the cells defined in definitions 2.5.13 to 2.5.15. Then A is the diagonal matrix where the i -th diagonal entry is an area associated to the vertex v_i , as in example (ii) above.

(iv) Let $\mathcal{A} : \mathcal{V} \times \mathcal{V} \rightarrow [0, \infty)$ be defined via

$$\mathcal{A}(v_i, v_j) = \begin{cases} \text{the number of neighbours of } v_i, & \text{if } i = j, \\ 0, & \text{otherwise.} \end{cases}$$

That is, $A = \text{Deg}(M)$.

(v) Let $\mathcal{A} : \mathcal{V} \times \mathcal{V} \rightarrow [0, \infty)$ be defined via

$$\mathcal{A}(v_i, v_j) = \begin{cases} \frac{1}{6} \sum_{t \in \mathcal{T}(i)} \text{area}(t), & \text{if } i = j, \\ \frac{1}{12} (\text{area}(t_{ij}) + \text{area}(\bar{t}_{ij})), & \text{otherwise,} \end{cases}$$

where t_{ij}, \bar{t}_{ij} are the faces which share the edge with endpoints v_i and v_j .

Note that the final example provides an area matrix which is not diagonal. The property of being diagonal is often desirable as diagonal matrices are easy to invert, are very sparse and are easy to use in calculations. To avoid the problem of non-diagonal area matrices a sum can be used to obtain a diagonal matrix.

Definition 2.5.18: Let M be a square matrix. The **lumped matrix** \bar{M} obtained from M is given by the diagonal matrix with entries $\bar{M}_{ii} = \sum_j M_{ij}$.

A diagonal area matrix gives a quick and simple approximation of an integral of a real-valued function over the mesh. The integral can be approximated by summing the value of the function multiplied by the associated area at each vertex. That is, for 2-manifold \mathcal{N} ,

$$\int_{\mathcal{N}} f \approx \sum_{v_i \in N} f(v_i) A_i \quad (2.5.3)$$

where N is a mesh constructed from \mathcal{N} and A_i is an area associated to the vertex v_i . (For convenience it is assumed that the vertices of N lie on the manifold \mathcal{N} , so that the function f can be evaluated on the vertices.) Note that the diagonal entry A_{ii} of the (lumped) area matrix A and the area A_i associated to the vertex v_i for the purpose of approximating the integral are assumed to coincide.

Remark 2.5.19: From here it is assumed that area matrices are diagonal (or lumped).

Definition 2.5.20: Let N be a mesh with vertex set \mathcal{V} and face set \mathcal{T} . A **partial weight matrix** \widehat{W} is an $n \times n$ symmetric matrix with entries defined via a function $\mathcal{W} : \mathcal{V} \times \mathcal{V} \rightarrow (-\infty, 0]$, such that

$$\widehat{W}_{ij} = \begin{cases} \mathcal{W}(v_i, v_j), & \text{when } i \neq j \\ 0, & i = j. \end{cases}$$

Let \bar{W} denote the lumped partial weight matrix, then a **weight matrix** W is defined to be

$$W := \widehat{W} - \bar{W}.$$

That is, W has entries

$$\begin{aligned} W_{ij} &= \widehat{W}_{ij}, \quad i \neq j \\ W_{ii} &= - \sum_j \widehat{W}_{ij} \\ &= - \sum_{j \neq i} W_{ij}. \end{aligned}$$

Proposition 2.5.21: If W is constructed as above then W is symmetric and positive semi-definite.

Proof. Symmetry is clear from the definition of W . To show that W is positive semi-definite first note that W is real and symmetric so has real eigenvalues.

Hence, via Geršgorin's theorem [15, theorem 6.1.1] all eigenvalues λ of W are such that there exists $1 \leq i \leq n$ with

$$|\lambda - W_{ii}| \leq \sum_{\substack{i=1 \\ i \neq j}}^n |W_{ij}|.$$

That is, since $W_{ij} \leq 0$,

$$\begin{aligned} |\lambda - W_{ii}| &\leq \sum_{i \neq j} -W_{ij} \\ &\leq - \sum_{i \neq j} W_{ij} \\ &\leq W_{ii}. \end{aligned}$$

Therefore

$$-W_{ii} \leq \lambda - W_{ii}$$

and so

$$0 \leq \lambda.$$

That is, all eigenvalues of W are non-negative, and hence W is positive semi-definite. \square

Examples 2.5.22:

(i) Let $\mathcal{W} : \mathcal{V} \times \mathcal{V} \rightarrow (-\infty, 0]$ be defined via

$$\mathcal{W}(v_i, v_j) = \begin{cases} 0, & \text{if } v_i \text{ is not a neighbour of } v_j, \\ -\left(\frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2}\right), & \text{if } v_i \text{ neighbour of } v_j \text{ and } i \neq j, \\ 0, & \text{if } i = j. \end{cases}$$

where α_{ij}, β_{ij} are the angles of the triangles with edge $v_i v_j$, as shown in figure 2.7. Then W has entries

$$W_{ij} = -\left(\frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2}\right), \text{ when } i \neq j,$$

$$W_{ii} = -\sum_{i \neq j} W_{ij}.$$

This is the weight matrix for the frequently used cot Laplacian (see example 2.5.24.(iii)). The function \mathcal{W} only maps to $(-\infty, 0]$ if the sum $\cot \alpha_{ij} + \cot \beta_{ij}$ is positive. This is not true in general but can be guaranteed by ensuring that a mesh has high isotropy, that is, triangles are close to being equilaterals (see [24, section 6.1]). From here it assumed that meshes meet this condition.

(ii) Let $\mathcal{W} : \mathcal{V} \times \mathcal{V} \rightarrow (-\infty, 0]$ be defined via

$$\mathcal{W}(v_i, v_j) = \begin{cases} -1, & \text{if } v_i \text{ and } v_j \text{ are neighbours,} \\ 0, & \text{if } v_i \text{ and } v_j \text{ are not neighbours,} \\ 0, & \text{if } i = j. \end{cases}$$

Then $W = \text{Deg } N - \text{Adj}(N)$.

Definition 2.5.23: A discrete **Laplace-Beltrami operator** (LBO) L is given by

$$L = A^{-1}W, \tag{2.5.4}$$

where A is an area matrix and W is a weight matrix. It is common to refer to such an operator as a **Laplacian**.

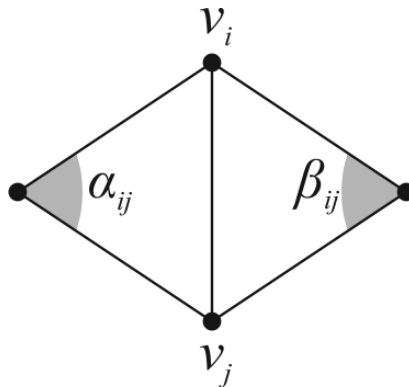


Figure 2.7: Construction of $\mathcal{W}(v_i, v_j)$

Discrete Laplace-Beltrami operators can be used to emulate the behaviour of the Laplace-Beltrami operator in the smooth case, but the above definition is extremely flexible. By combining area and weight matrices from the examples above various discrete Laplacians can be constructed.

Examples 2.5.24:

- (i) The **graph Laplacian** is given by $L = A^{-1}W$ where $A = I$ and $W = \text{Deg}(N) - \text{Adj}(N)$, [47, p.4].
- (ii) The **uniform Laplacian** is given by $L = A^{-1}W$ where $A = \text{Deg}(N)$ and $W = \text{Deg}(N) - \text{Adj}(N)$, [26],[24, 3.3.4].
- (iii) The **cot Laplacian** is given by $L = A^{-1}W$ where A is given by some diagonal area matrix and W is given by the W defined above in example 2.5.22.(i), [24, 3.3.4], [48].
- (iv) The **FEM Laplacian** is given by $L = A^{-1}W$ where A is given by the non-diagonal A defined above in example 2.5.17.(v) and W is given by the W defined above in example 2.5.22.(i), [23, 3.1].

Note that the definition of a discrete Laplace-Beltrami operator does not include the Tutte Laplacian [49] as it is not symmetric, but does include the symmetric quasi-Laplacian of [50].

Remark 2.5.25: When A is diagonal, the entries of L are given by

$$L_{ij} = \frac{W_{ij}}{A_{ii}}.$$

Figure 2.8 displays a small sub-matrix of the area, weight and resulting Laplacian matrices defined in the above examples, calculated for the homer mesh. Note that the area matrices for the graph and uniform Laplacians have not been scaled, and the area matrix for the FEM Laplacian has not been lumped.

The discrete Laplace-Beltrami operator has some important properties. Let $L = A^{-1}W$ be an $n \times n$ Laplacian and let $x, y \in \mathbb{R}^n$.

Proposition 2.5.26: L is self-adjoint with respect to the A inner product. That is, $\langle Lx, y \rangle_A = \langle x, Ly \rangle_A$.

Proof. See lemma A.2. □

The Laplacian eigenproblem is given by

$$L\phi = \lambda\phi$$

or equivalently

$$W\phi = \lambda A\phi. \tag{2.5.5}$$

Proposition 2.5.27: Eigenvalues of L are real and non-negative.

Proof. See lemma A.3. □

The constant function has eigenvalue $\lambda = 0$. This is a consequence of the construction of W . Figure 2.9 displays the first 7 Laplacian eigenfunctions constructed for the homer mesh, using the Laplacians listed in examples 2.5.24.

Proposition 2.5.28: Eigenfunctions of L with distinct eigenvalues are orthogonal with respect to the A inner product.

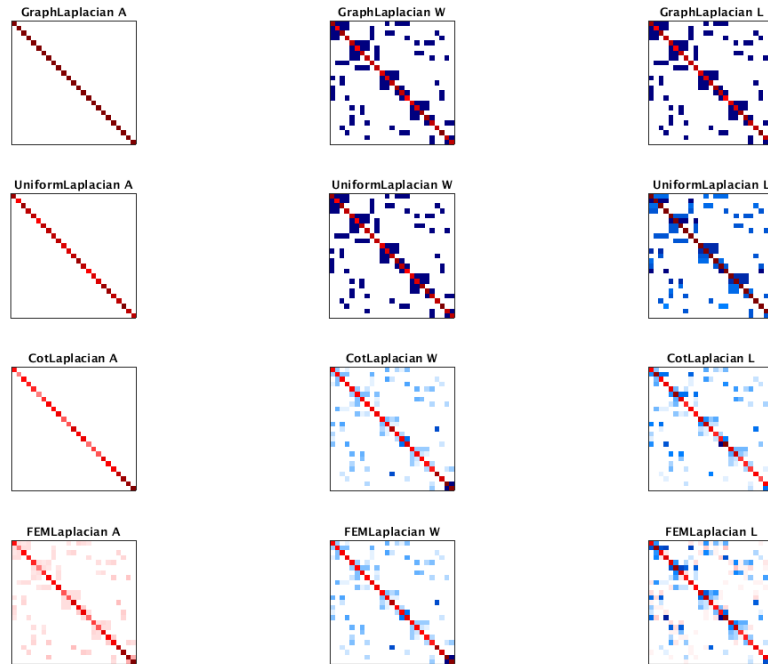


Figure 2.8: Area and weight matrices, and the resulting Laplacian, constructed via the examples listed in 2.5.24. The matrices display only the upper left 25×25 square of elements. The entries are coloured such that negative values are blue and positive values are red. Paler shades indicate that the values are close to zero (coloured white).

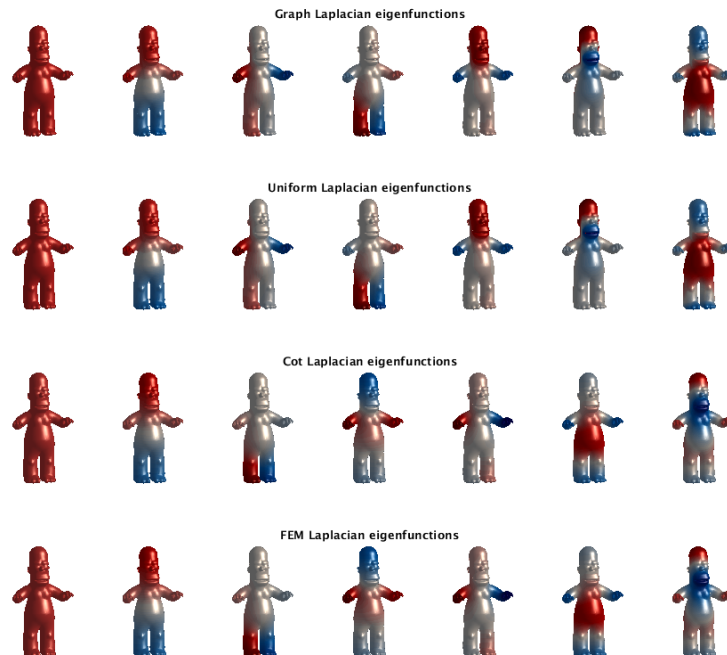


Figure 2.9: The first 7 eigenfunctions for Laplacians constructed as in examples 2.5.24.

(For proof see lemma A.3.)

Remark 2.5.29: From here it is assumed that the eigenfunctions are sorted according to the natural ordering such that $\lambda_1 \leq \dots \leq \lambda_n$.

Proposition 2.5.30: The Laplacian eigenfunctions form a basis for $\mathcal{F}(N, \mathbb{R})$.

Proof. The result follows from the fact that $\mathcal{F}(N, \mathbb{R}) \cong \mathbb{R}^n$ and that since the eigenvectors are orthogonal and there are n of them then they form a basis for \mathbb{R}^n . \square

Denote by Λ the $n \times n$ diagonal matrix with entries $\Lambda_{ii} = \lambda_i$ then the Laplacian eigenproblem, as in equation (2.5.5), can be stated in matrix form as

$$W\Phi = A\Phi\Lambda$$

or, for a truncated set of eigenfunctions,

$$W\Phi_k = A\Phi\Lambda_k \tag{2.5.6}$$

where Λ_k denotes the $k \times k$ upper-left submatrix of Λ . Note that this leads to the equality

$$\Phi_k^T W\Phi_k = \Lambda_k. \tag{2.5.7}$$

Let N be a mesh with Laplacian $L_N = A_N^{-1}W_N$ and eigenpairs (ϕ_i, λ_i) . Let P be a scaling of N such that $L_P = A_P^{-1}W_P$ with $W_P = W_N$ and $A_P = \mu A_N$, $\mu \in \mathbb{R}^+$. (That is, the vertex positions of P have changed so the face areas have been scaled, but the weight matrix does not change.) Consider the L_P eigenproblem,

$$L_P\varphi_i = \sigma_i\varphi_i,$$

then since $L_P = \frac{1}{\mu}A_N^{-1}W_N$,

$$\frac{1}{\mu}L_N\varphi_i = \sigma_i\varphi_i$$

$$L_N\varphi_i = \mu\sigma_i\varphi_i.$$

Since the eigenvectors of L_N are ϕ_i it must be that $\mu\sigma_i = \lambda_i$ and hence $\sigma_i = \frac{\lambda_i}{\mu}$ is an eigenvalue of L_P for eigenvector ϕ_i .

Remark 2.5.31: To allow comparison between meshes all area matrices are scaled such that $\sum_i A_{ii} = 1$.

An important property of the Laplacian eigenfunctions is that they are critical points of the Dirichlet energy. First consider the manifold case.

The Dirichlet energy for a real-valued function $f : \mathcal{M} \rightarrow \mathbb{R}$ is given by

$$E(f) = \int_{\mathcal{M}} \langle \nabla f, \nabla f \rangle_{\ell_2} [51], \quad (2.5.8)$$

where the ℓ_2 inner product is defined by

$$\langle p, q \rangle_{\ell_2} := p \cdot q \quad (2.5.9)$$

for $p, q \in \mathbb{R}^n$.

Green's identity for manifolds without boundary states

$$\int_{\mathcal{M}} \langle \nabla f, \nabla g \rangle_{\ell_2} - \int_{\mathcal{M}} f \Delta g = 0 [52, \text{chapter 2, theorem 5.13}]$$

for functions $f, g : \mathcal{M} \rightarrow \mathbb{R}$. Using this as an alternative definition for the Dirichlet energy,

$$E(f) = \int_{\mathcal{M}} \langle \nabla f, \nabla f \rangle_{\ell_2} = \int_{\mathcal{M}} f \Delta f.$$

Then, discretising to a mesh M constructed from \mathcal{M} with discrete Laplacian $L = A^{-1}W$ gives, via equation (2.5.3),

$$\begin{aligned} E(f) &\approx \sum_{v_i \in M} f(v_i) (Lf(v_i)) A_{ii} \\ &= \sum_{i=1}^n \mathbf{f}_i (L\mathbf{f})_i A_{ii} \\ &= \mathbf{f}^T A(L\mathbf{f}) \\ &= \mathbf{f}^T A A^{-1} W \mathbf{f} \\ &= \mathbf{f}^T W \mathbf{f}. \end{aligned} \quad (2.5.10)$$

Definition 2.5.32: Let $f : M \rightarrow \mathbb{R}$ be a function defined on mesh M , represented by the vector $\mathbf{f} \in \mathbb{R}^n$. Then the **discrete Dirichlet energy** $E(f)$ is defined to be

$$E(f) := \mathbf{f}^T W \mathbf{f}$$

where W is the weight matrix of the Laplacian associated to M .

Proposition 2.5.33: The Laplacian eigenfunctions are critical points of the problem

$$\min_{x \in \mathbb{R}^n} x^T W x \text{ subject to } x^T A x = 1.$$

Proof. The Lagrange multiplier function is given by

$$\mathcal{L}(x, \lambda) = x^T W x - \lambda x^T A x.$$

Differentiating with respect to x and equating with zero gives

$$2Wx - 2\lambda Ax = 0.$$

Rearranging and simplifying leads to the Laplacian eigenproblem

$$Wx = \lambda Ax,$$

and hence, Laplacian eigenfunctions are critical points of the Dirichlet energy. \square

Corollary 2.5.34: The matrix Φ_k minimises

$$\min_{X \in \mathbb{R}^{n \times k}} \text{tr}(X^T W X) \text{ subject to } X^T A X = I.$$

Proof. This follows from proposition 2.5.28. \square

Corollary 2.5.35: $\text{tr}(\Phi_k^T W \Phi_k) = \sum_{i=1}^k \lambda_i$.

Chapter 3

Generalised Manifold Harmonics

The Laplacian eigenfunctions are widely used in geometry processing [2],[3],[4],[5]. Some related functions include the recent works of localised manifold harmonics [6] and Hamiltonian eigenfunctions [7], which aim to construct additional functions to combat the loss of information due to basis truncation, and compressed manifold modes which include a sparsity inducing condition. These functions all have a similar form, minimising a trace subject to an orthogonality condition. Background to these alternative basis methods is provided in the first two sections. The section on compressed manifold modes includes the addition of scaling by the area matrix into the ℓ_1 norm, and comments on the errors in the method presented in [9].

Here a general problem for finding orthogonal functions is posed, with solutions referred to as *generalised manifold harmonics*. The formulation has Laplacian eigenfunctions, localised manifold harmonics, Hamiltonian eigenfunctions and compressed manifold modes as specific cases.

An important property of the Laplacian eigenfunctions is that, in the smooth case, they commute with isometry. This is a well-exploited idea in geometry processing, particularly in shape matching (see section 5.1) but a definition of isometry in the discrete case is brushed over. Here a definition of a discrete isometry is presented, constructed via an analogy to necessary and sufficient conditions for isometry between Riemannian manifolds. It is then proved that Laplacian eigenfunctions,

localised manifold harmonics, Hamiltonian eigenfunctions and compressed manifold modes commute with discrete isometry.

Finally, the conditions required for generalised manifold harmonics to commute with discrete isometry are discussed.

3.1 Background: Supplementary Basis Functions

The eigenfunctions of a Laplacian constructed for a mesh N provide a basis for the function space $\mathcal{F}(N, \mathbb{R})$. Any function can be projected in to a truncation of this basis, but high frequency information is lost. This loss of information is visualised well by considering the effect of the reconstruction of functions which have small local support (such as a highly peaked delta function) or which have unique values which are very close together (such as the vertex position functions). Figure 3.1 shows the effect of basis truncation on a delta function, made visible on the original mesh by a marker (in red) and on the vertex positions. The red patches show the dispersion of the reconstructed delta function and the error in in vertex reconstruction respectively.



Figure 3.1: The effect of basis truncation on reconstruction of (a) a delta function, (b) vertex positions.

There has been some recent work which aims to combat the problem of loss of detail due to truncation of the spectral basis; a consequence of removing high frequency eigenfunctions. This section combines the methods presented at SGP

2017 in the posters ‘Localized Manifold Harmonics for Spectral Shape Analysis’ (Melzi et al) [6] (section 3.1.1) and ‘Schrödinger Operator for Sparse Approximation of 3D Meshes’ (Choukron et al) [7] (section 3.1.2). The associated papers are [53] and [54] respectively. The methods are very similar, and section 3.3 provides a generalisation.

The driving idea behind the aforementioned methods is to construct an additional set of basis functions, based on the eigenfunctions of a Laplacian, which in some way compensate for the loss in high frequency eigenfunctions following basis truncation.

Throughout this section L denotes a discrete Laplacian matrix, formed by area matrix A and weight matrix W , such that $L = A^{-1}W$. The matrix Φ_k is the matrix of the first k eigenfunctions of L , where columns correspond to eigenfunctions and rows correspond to the value of the functions at a specific vertex. In [6] the number of additional functions l is taken to be $\frac{k}{2}$.

3.1.1 Localised Manifold Harmonics

In [6], given a mesh N , the aim is to construct a set of l functions ψ_i , stored as columns in a matrix Ψ , which are

- (i) eigenfunction-like, i.e. critical points of the Dirichlet energy; (The Dirichlet condition)
- (ii) A -orthonormal, with respect to the first k Laplacian eigenfunctions ϕ_j and with respect to each other; (The orthogonality condition)
- (iii) localised to a specified region, R . (The localisation condition)

To construct a minimisation problem which has the ψ_i as solutions consider each condition separately.

The Localisation Condition

Let R be a subset of the vertices of M .

Definition 3.1.1: A function f is **localised to a specific region** R if it has local support, i.e. $f(x) = 0$ if $x \notin R \subseteq M$.

The region R can be described via an indicator function on the vertices $x \in M$,

$$u(x) = \begin{cases} 1, & x \in R, \\ 0, & x \notin R. \end{cases}$$

When searching for functions ψ_i which are localised to a region R , described by indicator function u , consider a quadratic penalty which must be minimised:

$$\min_{\psi_i} \sum_x (\psi_i(x) (1 - u(x)))^2 A_{xx} \quad (3.1.1)$$

where A_{xx} associates the penalty to the area around the vertex x as defined the (lumped) area matrix of the Laplacian calculated for M . Define

$$v(x) := (1 - u(x))^2 \quad (3.1.2)$$

and note that $v(x) = 1$ if $u(x) = 0$ and $v(x) = 0$ if $u(x) = 1$. Then equation (3.1.1) can be rewritten as

$$\begin{aligned} & \min_{\psi_i} \sum_x \psi_i(x) (A \operatorname{diag}(v(x)))_{xx} \psi_i(x) \\ &= \min_{\psi_i} \psi_i^T A \operatorname{diag}(v(x)) \psi_i. \end{aligned}$$

For a set of l functions $\{\psi_i\}$ expressed as a matrix Ψ this becomes the minimisation of a trace,

$$\min_{\Psi \in \mathbb{R}^{n \times l}} \operatorname{tr} (\Psi^T A \operatorname{diag}(v(x)) \Psi). \quad (3.1.3)$$

A major advantage of this method is the ability to choose exactly the area of focus for the localised functions, unlike compressed manifold modes where the local support

depends on many variables, including mesh density and parameters within the optimisation problem (see section 3.2). In [6], R is chosen by reconstructing vertex positions in the truncated basis Φ_k , and selecting vertices which do not lie within a certain distance from their original position. (Note, R may not be connected, or consistent across pairs of meshes when applied to the standard matching problem.)

The Orthogonality Condition

The functions ψ_i must be A -orthogonal to the functions ϕ_j . That is,

$$\psi_i^T A \phi_j = 0 \quad \forall \psi_i, \phi_j \text{ where } 1 \leq i \leq k \text{ and } 1 \leq j \leq n.$$

In matrix form this can be written as $\Psi^T A \Phi_k = 0_{l \times k}$, and the aim is to minimise (with respect to Ψ)

$$\|\Psi^T A \Phi_k\|_F^2 = \text{tr}(\Psi^T A \Phi_k \Phi_k^T A \Psi). \quad (3.1.4)$$

The functions ψ_i must also be A -orthogonal to one another: $\psi_i^T A \psi_j = \delta_{ij}$. This will be enforced by adding the condition that $\Psi^T A \Psi = I_{l \times l}$ to the minimisation problem.

The Dirichlet Condition

The Dirichlet condition aims to find ψ_i which minimise $L\psi_i$. Recall from proposition 2.5.33 and corollary 2.5.34 that when $\Psi^T A \Psi = I$ the Dirichlet energy is minimised by Laplacian eigenfunctions, and that in matrix form the condition can be written as

$$\min_{\Psi \in \mathbb{R}^{n \times l}} \text{tr}(\Psi^T W \Psi) \text{ subject to } \Psi^T A \Psi = I.$$

Combining the Conditions

The localisation, orthogonality and Dirichlet conditions can be combined into the following minimisation problem:

$$\min_{\Psi} \text{tr}(\Psi^T (W + \mu_R A \text{diag}(v) + \mu_{\perp} A \Phi_k \Phi_k^T A) \Psi) \text{ subject to } \Psi^T A \Psi = I_{l \times l}, \quad (3.1.5)$$

where μ_R and μ_\perp are scalar parameters which control the weighting of the localisation and orthogonality terms respectively.

This can be solved as an eigenproblem, another significant feature of this method of basis construction. As a consequence, it is simple to define an ordering on the LMHs, achieved by sorting the eigenfunctions ψ_i in relation to their eigenvalues λ_i . The eigenvalues λ_i are given by

$$\lambda_i = \psi_i^T (W + \mu_R A \text{diag}(v) + \mu_\perp A \Phi_k \Phi_k^T A) \psi_i.$$

Definition 3.1.2: Solutions to the optimisation problem (3.1.5) are called **localised manifold harmonics** (LMHs).

An alternative set of basis functions for a $(k + l)$ -dimensional linear subspace of $\mathcal{F}(N, \mathbb{R})$ is given by the columns of $X = [\Phi_k \ \Psi]$, formed by concatenation of the Φ_k and Ψ matrices. Figure 3.2 shows the reconstruction of the vertex positions using a truncated basis of 100 Laplacian eigenfunctions, 150 eigenfunctions and the first 100 eigenfunctions supplemented by 50 localised manifold harmonics. The local area R was located by measuring the failure to reconstruct the vertex positions. Red patches show areas with greatest reconstruction error. Figure 3.3 shows the failure to meet the $\Phi_k^T A \Psi = I$ orthogonality condition. The left-hand matrix shows the matrix $X^T A X$ where X is the matrix of basis functions. The right-hand matrix shows $I - X^T A X$, which would be the zero matrix if X were truly A -orthonormal. The set of LMH functions is close to being orthogonal, but there is some error. This error lies between the Laplacian eigenfunctions and the additional LMHs. (For more on this failure to meet the orthogonality condition see section 4.3.1.)

3.1.2 Hamiltonian Eigenfunctions

In [7], given a mesh M with truncated eigenbasis Φ_k and a potential function represented by an $n \times n$ diagonal matrix V , the aim is to find a new orthonormal basis for $\mathcal{F}(N, \mathbb{R})$ which improves reconstruction of areas with fine detail. Clearly the idea is very similar to that of the localised manifold harmonics, but the formulation of the minimisation problem is slightly different.

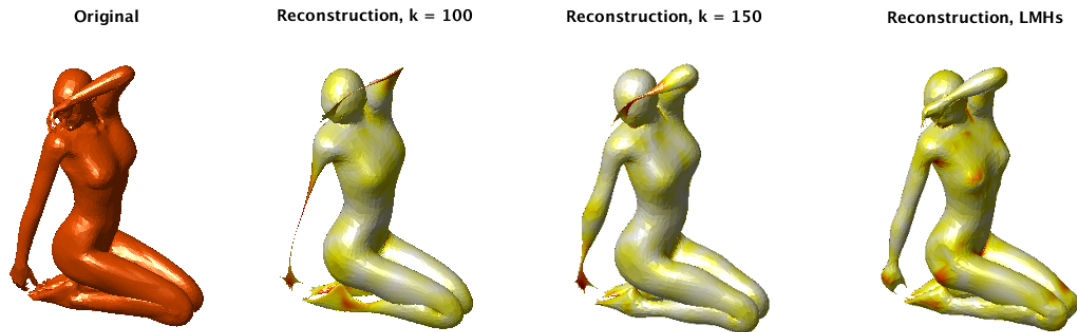


Figure 3.2: The effect of basis truncation on reconstruction of vertex positions, improved by the addition of a set of localised manifold harmonics.

Definition 3.1.3: Let L be a Laplacian for a mesh N with n vertices and let $V \in \mathbb{R}^{n \times n}$ be diagonal. A **Hamiltonian operator** H is defined to be

$$H := L + \mu V, \quad (3.1.6)$$

where μ is a scalar. The matrix V is called the **potential**.

The parameter μ controls the impact of the potential V : small μ leads to solutions which minimise the total energy; $\mu = 0$ will result in Laplacian eigenfunctions ϕ ; large μ will promote solutions which are close to being zero on areas with a high potential. Note that the usual definition of H is given by $H = -L + \mu V$. Since the sign of L depends on a choice of definition for L ($L = -A^{-1}W$ or $L = A^{-1}W$), removing the minus sign allows a straightforward combination of the method of Hamiltonian eigenfunctions with LMHs.

Similar to the choice of region in the work on LMHs, V is chosen to be a diagonal matrix which weights vertices, according to the distance between the original position of the vertex and the position of the vertex when reconstructed in a truncated Laplacian eigenbasis. Vertices with a large reconstruction error have a small weight and vertices with a low reconstruction error have a large weight. Then, setting μ to be large will result in functions ψ which are close to being zero on areas which are well reconstructed in the Laplacian eigenbasis Φ_k .

The eigenproblem $H\psi = \lambda\psi$ can be reformulated by multiplying on both sides by

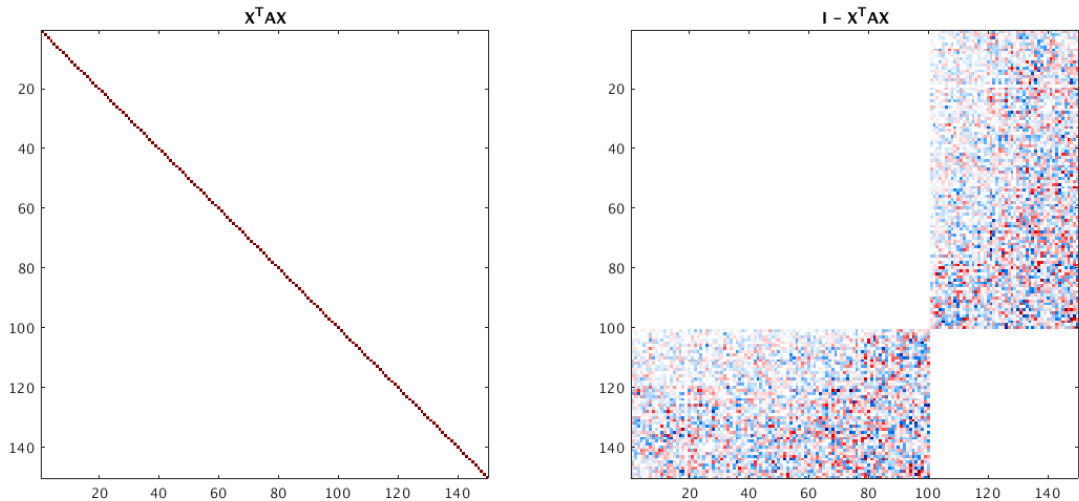


Figure 3.3: Orthogonality of LMHs, the maximum absolute value of any non-zero entry in the right-hand matrix is of order 10^{-1} .

A :

$$(W + \mu AV) \psi = A \lambda \psi. \quad (3.1.7)$$

Lemma 3.1.4: Let $L = A^{-1}W$ be a discrete Laplacian. Then, the operator $H := L + \mu V$ is self-adjoint, with respect to A .

Proof. Assume A, V are diagonal matrices and assume W is a symmetric matrix. Let f and g be vectors, then

$$\begin{aligned} \langle Hf, g \rangle_A &= f^T H^T A g \\ &= f^T (L^T + \mu V^T) A g \\ &= f^t (W A^{-1} + \mu V) A g, \text{ since } A, V, W \text{ are symmetric} \\ &= f^t (W + \mu AV) g, \text{ since } AV = VA \\ &= f^T A (L + \mu V) g \\ &= f^T A H g \\ &= \langle f, Hg \rangle_A. \end{aligned}$$

□

This means that the normalised eigenfunctions of a self-adjoint H will form an A -orthonormal basis for $\mathcal{F}(M, \mathbb{R})$.

Solutions ψ_i to the eigenproblem (3.1.7) satisfy $(W + \mu AV)\psi_i = \lambda_i A\psi_i$ and are also solutions to the minimisation problem

$$\min_f \operatorname{tr} (f^T (W + \mu AV) f) \quad \text{subject to } f^T A f = 1. \quad (3.1.8)$$

For a set of l solutions the problem can be given in matrix form as

$$\min_{\Psi \in \mathbb{R}^{n \times l}} \operatorname{tr} (\Psi^T (W + \mu AV) \Psi) \quad \text{subject to } \Psi^T A \Psi = \Psi, \quad \mu \in \mathbb{R}, \quad (3.1.9)$$

which can be solved as a generalised eigenproblem.

Definition 3.1.5: The solutions to equation (3.1.9) are called **Hamiltonian eigenfunctions**.

In [7] the Hamiltonian eigenfunctions Ψ are used to complement the Laplacian eigenfunctions and orthogonality between the two sets of functions is promoted via an iterative method of solution (SOMP [55]) which also solves for an optimum value for μ . In section 3.3 Hamiltonian eigenfunctions and localised manifold harmonics are combined into a generalised equation, to which they are specific solutions.

3.2 Background: Compressed Manifold Modes

Compressed modes, introduced by Ozolinš et al. [8] and applied to geometry processing by Neumann et al. [9], provide an alternative to Laplacian eigenfunctions. A sparsity inducing term is used to obtain functions which are sparse minimisers of the Dirichlet energy. Sparsity is induced by the addition of an ℓ_1 term to the usual eigenproblem and leads to functions which have localised support. This section begins with a discussion of the ℓ_1 norm and sparsity, followed by a description of the compressed mode problem and an algorithm for finding solutions.

3.2.1 Sparsity and the ℓ_1 Norm

The ℓ_1 norm, also known as the *taxicab distance* is used in regression analysis [56],[57], popularised in LASSO [58], to prevent overfitting – the fitting of model too closely to known data, which prevents reliable forecasting. For the same reason, it is important in machine learning [59]. Applications in geometry processing include splines (piecewise polynomial curves used by CAD packages) [60],[61]; surface reconstruction [62]; shape matching [63]; and compressed manifold modes.

Definition 3.2.1: Let X be an $m \times n$ matrix. Let z be the number of elements equal to zero. The **sparsity** of X is the percentage of zero elements:

$$\text{Sparsity}(X) = \frac{z}{mn} \times 100.$$

A matrix with sparsity greater than a specific value ε (say $\varepsilon = 50$) is called **sparse** and a matrix with sparsity less than or equal to ε is called **dense**.

The higher the sparsity, the less storage is required on a computer. Using sparse matrices in calculations can also have significant benefit to run times [64]. This makes sparse matrices desirable. Unfortunately, although the Laplacian eigenfunctions are easy to calculate (as solutions to a generalised eigenvalue problem), a matrix of eigenfunctions will be dense. This is because, in general, eigenfunctions are supported globally across the mesh.

Recall from definition 2.1.2 that the ℓ_1 norm of a matrix X is defined to be

$$\|X\|_1 = \sum_{ij} |X_{ij}|.$$

To see how the ℓ_1 norm induces sparsity consider a comparison with the Euclidean norm the two-dimensional case. (This becomes Frobenius norm for matrices.) Let $v = (x, y) \in \mathbb{R}^2$ be such that $\|v\|_1 = |x| + |y| = r$ and let $w = (a, b) \in \mathbb{R}^2$ such that $\|w\|_2 = (a^2 + b^2)^{\frac{1}{2}} = r$. Some values for v and w for different r are plotted in figure 3.4.

Proposition 3.2.2: Let $v \in \mathbb{R}^n$, then $\|v\|_2 \leq \|v\|_1 \leq \sqrt{n}\|v\|_2$.

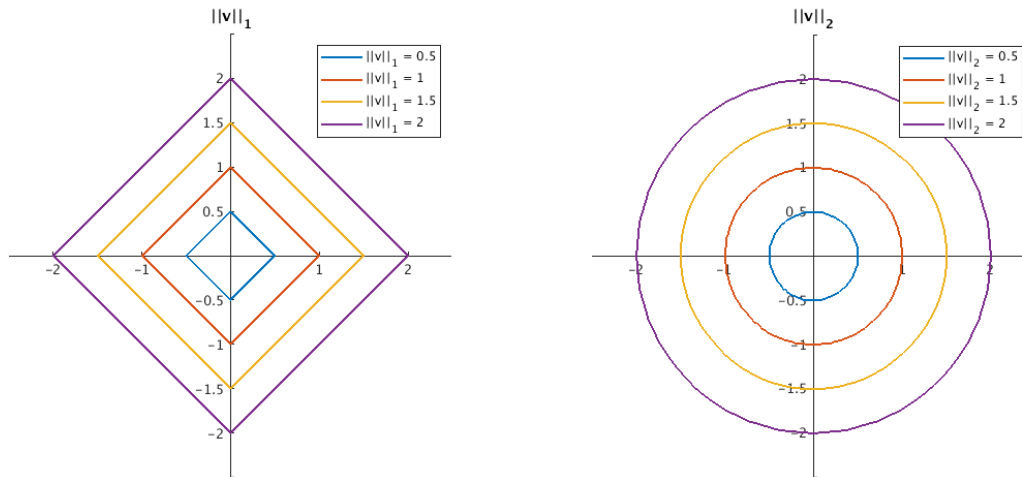


Figure 3.4: The l_1 norm in comparison to the l_2 norm.

Proof. To show the first inequality

$$\begin{aligned}
 \|v\|_2^2 &= \sum_i v_i^2 \\
 &= \sum_i |v_i|^2 \\
 &\leq \left(\sum_i |v_i|\right)^2 \\
 &\leq \|v\|_1^2.
 \end{aligned}$$

Therefore $\|v\|_2 \leq \|v\|_1$. For the second inequality denote the n -dimensional vector with all entries equal to one by $\mathbf{1}_n$. Then

$$\begin{aligned}
 \|v\|_1 &= |v|^T \mathbf{1}_n \\
 &\leq (|v|^T |v|)^{\frac{1}{2}} (\mathbf{1}_n^T \mathbf{1}_n)^{\frac{1}{2}}, \text{ by the Cauchy-Schwartz inequality,} \\
 &\leq \|v\|_2 \sqrt{n}.
 \end{aligned}$$

□

Proposition 3.2.2 generalises to matrices via the Frobenius norm.

Proposition 3.2.3: Let $X \in \mathbb{R}^{n \times k}$, then $\|X\|_F \leq \|X\|_1 \leq \sqrt{nk} \|X\|_F$.

Proof. To show the first inequality

$$\begin{aligned}
\|X\|_F^2 &= \sum_{ij} X_{ij}^2 \\
&= \sum_{ij} |X_{ij}|^2 \\
&\leq \left(\sum_{ij} |X_{ij}| \right)^2 \\
&\leq \|X\|_1^2.
\end{aligned}$$

Therefore $\|X\|_F \leq \|X\|_1$. For the second inequality denote the nk -dimensional vector with all entries equal to one by 1_{nk} . Then, recalling the vec operator defined in definition 5.1.4,

$$\begin{aligned}
\|X\|_1 &= \text{vec}(|X|)^T 1_{nk} \\
&\leq (\text{vec}(|X|)^T \text{vec}(|X|))^{1/2} (1_{nk}^T 1_{nk})^{1/2}, \text{ by the Cauchy-Schwartz inequality,} \\
&\leq \|X\|_F \sqrt{nk}.
\end{aligned}$$

□

Define the n -dimensional ball of radius r by

$$B(n, r) := \{v \in \mathbb{R}^n : \|v\|_2 \leq r\};$$

the $(n - 1)$ -dimensional sphere of radius r by

$$\mathbb{S}_r^{(n-1)} := \{v \in \mathbb{R}^n : \|v\|_2 = r\};$$

the n -dimensional l_1 ball of radius r by

$$L(n, r) := \{v \in \mathbb{R}^n : \|v\|_1 \leq r\}.$$

Proposition 3.2.4: For $n > 1$, $L(n, r) \subset B(n, r)$.

Proof. Let $s \in L(n, r)$, then $s = (s_1, \dots, s_n)$ such that $\sum_i |s_i| \leq r$. The point s has one of two forms:

- (i) $s_i = \pm r$ for one i , $s_j = 0$ for all $j \neq i$,
- (ii) $|s_i| < r$ for all i .

Points of form (i) clearly lie in $B(n, r)$. Let s be a point of form (ii) and consider $\|s\|_2^2$:

$$\begin{aligned} \|s\|_2^2 &\leq \|s\|_1^2, \text{ via proposition 3.2.2,} \\ &\leq r^2, \text{ since } \sum_i |s_i| \leq r \text{ as } s \in L(n, r). \end{aligned} \quad (3.2.10)$$

It follows that $\|s\|_2 \leq r$ so $s \in B(n, r)$ and $L(n, r) \subseteq B(n, r)$.

To see that $L(n, r) \subset B(n, r)$ consider the point $b \in \mathbb{R}^n$ with all entries $b_i = \frac{r}{\sqrt{n}}$. Then

$$\begin{aligned} \|b\|_2 &= \left(n \frac{r^2}{n} \right)^{\frac{1}{2}} = r \quad \text{so } b \in B(n, r), \text{ but} \\ \|b\|_1 &= n \left| \frac{r}{\sqrt{n}} \right| = r\sqrt{n} \quad \text{so } b \notin L(n, r). \end{aligned}$$

□

Corollary 3.2.5: The intersection $L(n, r) \cap \mathbb{S}_r^{(n-1)}$ is given by the set of all points of the form $s = (s_1, \dots, s_n)$ such that $s_i = \pm r$ for one i , $s_j = 0$ for all $j \neq i$.

Proof. Let $s \in L(n, r) \cap \mathbb{S}_r^{(n-1)}$. Then, since $s \in L(n, r)$,

$$\begin{aligned} \left(\sum_i |s_i| \right)^2 &\leq r^2, \text{ via (3.2.10),} \\ \sum_i |s_i|^2 + \sum_{i \neq j} |s_i| |s_j| &\leq \sum_i s_i^2, \text{ since } s \in \mathbb{S}_r^{(n-1)}. \end{aligned}$$

Therefore, since $\sum_i |s_i|^2 = \sum_i s_i^2$, the sum $\sum_{i \neq j} |s_i| |s_j|$ is equal to zero. This is only possible when at most one of the s_i is non-zero, therefore s must be of the form $s = (s_1, \dots, s_n)$ such that $s_i = \pm r$ for one i , $s_j = 0$ for all $j \neq i$. □

Corollary 3.2.6: Minimisers of the problem

$$\arg \min_v \|v\|_1 \text{ subject to } v^T v = r^2 \quad (3.2.11)$$

are given by v of the form $s = (s_1, \dots, s_n)$ such that $s_i = \pm r$ for one i , $s_j = 0$ for all $j \neq i$.

Proposition 3.2.7: Let D be a $k \times k$ diagonal matrix with $D_{ii} > 0$ for all i . Then the minimiser $X^* \in \mathbb{R}^{n \times k}$ of $\|X\|_1$ such that $X^T X = D$ is given by

$$X^* = \Pi \begin{bmatrix} D^{\frac{1}{2}} \\ 0 \end{bmatrix},$$

where Π is an $n \times n$ permutation matrix.

Proof. First note that the condition $X^T X = D$ is equivalent $X_i^T X_j = D_{ii} \delta_{ij}$ for all (i, j) pairs, where X_i denotes the i -th column of X , and that $\|X\|_1 = \sum_i \|X_i\|_1$. This means that $\|X\|_1$ can be minimised by minimising $\|X_i\|_1$ for each i , subject to the conditions.

By corollary 3.2.6 $\|X_i\|_1$ is minimised by a column of form (i), with $r = \sqrt{D_{ii}}$. To enforce the $X_i^T X_j = 0$ condition for $i \neq j$ it must be that the index of the non-zero element of X_i is distinct for each i . The non-zero element can appear in any row of the $n \times k$ matrix solution but must correspond to the i -th element of the diagonal of D . This can be expressed as a permutation of the rows of the $n \times k$ block matrix constructed by $D^{\frac{1}{2}}$ and an $(n - k) \times k$ zero matrix. \square

Let M be a symmetric positive definite matrix. Then the points $x \in \mathbb{R}^n$ such that $x^T M x = 1$ form an $(n - 1)$ -dimensional ellipsoid. This can be used to generalise proposition 3.2.4.

Definition 3.2.8: An n -dimensional ellipsoid with semi-axes lengths $\{\frac{1}{\sqrt{\lambda_1}}, \dots, \frac{1}{\sqrt{\lambda_n}}\}$, is defined to be

$$\mathbb{E}_M = \{x \in \mathbb{R}^{n+1} : x^T M x = 1\}, \quad (3.2.12)$$

where M is a symmetric positive definite matrix with (non-negative) eigenvalues λ_i .

To see that this definition describes ellipsoids consider the diagonalisation of M : $M = Q D^T Q^T$ where Q is an $n \times n$ orthogonal matrix and D is an $n \times n$ diagonal

matrix with entries given by the eigenvalues λ_i of M . Any $x \in \mathbb{R}^n$ can be written as $x = Q^T y$ for $y \in \mathbb{R}^n$ and orthogonal Q . Then $\mathbb{E}_M = \{y \in \mathbb{R}^n : y^T Q D Q^T y = 1\}$ where $M = Q D Q^T$ and it follows that \mathbb{E}_M is $(n - 1)$ -dimensional ellipsoid.

Define the n -dimensional ellipsoidal ball by

$$E_M := \{v \in \mathbb{R}^n : v^T M v \leq 1\},$$

and define

$$b := \max_i \{M_{ii}\}.$$

Note that since M is positive definite b is the largest element in M . (For proof see A.4.)

Proposition 3.2.9: for $n > 1$, $L(n, \frac{1}{\sqrt{b}}) \subset E_M$.

Proof. Let $s \in L(n, \frac{1}{\sqrt{b}})$ then $s = (s_1, \dots, s_n)$ such that $\sum_i |s_i| \leq \frac{1}{\sqrt{b}}$. Then s has one of two forms:

1. $s_i = \pm \frac{1}{\sqrt{b}}$ for one i , $s_j = 0$ for all $j \neq i$,
2. $|s_i| < \frac{1}{\sqrt{b}}$ for all i .

Point of form 1 lie in E_M since

$$s^T B s = \sum_{ij} s_i M_{ij} s_j = s_i^2 M_{ii} = \frac{M_{ii}}{b} \leq 1,$$

since $M_{ii} \leq b = \max_i \{M_{ii}\}$ for all i .

Let s be of form 2, then

$$\begin{aligned} s^T M s &= \sum_{ij} s_i M_{ij} s_j \\ &\leq \sum_{ij} |s_i| |s_j| M_{ij} \\ &\leq b \sum_{ij} |s_i| |s_j|, \text{ since } \max_{ij} \{M_{ij}\} = \max_i \{M_{ii}\} = b, \\ &< b \cdot \frac{1}{b}, \text{ since } |s_i| < \frac{1}{\sqrt{b}} \text{ for all } i, \end{aligned}$$

and hence, points of form 2 lie in E_M .

To show that $L(n, \frac{1}{\sqrt{b}})$ is not equal to E_M define for ease of notation

$$\alpha = \sqrt{\sum_{ij} M_{ij}}$$

and define the vector v to be the n -dimensional vector with every element equal to $\frac{1}{\alpha}$. Then $v \in E_M$ since

$$\begin{aligned} v^T M v &= \sum_{ij} \frac{1}{\alpha^2} M_{ij} \\ &= \frac{1}{\alpha^2} \sum_{ij} M_{ij} \\ &= \frac{\alpha^2}{\alpha^2} \\ &= 1. \end{aligned}$$

Then, since M is positive definite not all entries of M are equal (see A.5 for proof) and so

$$\sum_{ij} M_{ij} < \sum_{ij} b.$$

That is

$$\alpha^2 < n^2 b$$

therefore

$$\alpha < n\sqrt{b}$$

and, as $\alpha, b > 0$,

$$\frac{1}{\sqrt{b}} < \frac{n}{\alpha}.$$

The ℓ_1 norm of v is given by

$$\begin{aligned} \|v\|_1 &= \sum_{i=1}^n \frac{1}{\alpha} \\ &= \frac{n}{\alpha} \\ &> \frac{1}{\sqrt{b}}. \end{aligned}$$

Hence $v \notin L\left(n, \frac{1}{\sqrt{b}}\right)$ and $L\left(n, \frac{1}{\sqrt{b}}\right) \subset E_M$. \square

3.2.2 Compressed Manifold Modes via ADMM

The usual Laplacian eigenproblem (via equation (2.5.34)) can be altered to produce functions which behave like eigenfunctions (minimising the Dirichlet energy) but which are also sparse. This is achieved by adding an ℓ_1 term to the function which is to be minimised.

Definition 3.2.10: Let $A, W \in \mathbb{R}^{n \times n}$ with A symmetric positive definite and W symmetric semi-positive definite. Let $\Psi \in \mathbb{R}^{n \times k}$ with $k \leq n$. Solutions to the problem

$$\arg \min_{\Psi} \text{tr}(\Psi^T W \Psi) + \mu \|A\Psi\|_1 \text{ subject to } \Psi^T A \Psi = I, \quad (3.2.13)$$

are called **compressed manifold modes** (compressed modes/CMMs). The parameter μ is called the **sparsity parameter**.

Note that the ℓ_1 term differs from [9], using $\|A\Phi\|_1$ instead of $\|\Phi\|_1$. The weighting by area more accurately represents the smooth case: Let \mathcal{M} be a smooth manifold and let $f : \mathcal{M} \rightarrow \mathbb{R}$, then the continuous ℓ_1 norm of f is defined to be

$$\|f\|_{L_1} := \int_{\mathcal{M}} |f(x)| da$$

where da is the standard area element on \mathcal{M} [65]. To discretise this let M be a mesh obtained from \mathcal{M} , then, via the discretisation of the integral,

$$\|f\|_{L_1} \approx \sum_i |f(v_i)| A_{ii}$$

where A_i denotes an area associated to the vertex v_i . For a diagonal area matrix A this is equal to $\|Af\|_1$. This is mentioned in [66],[63] and discussed in [65], which also presents a weighting based on one-ring neighbourhoods of vertices. However, it is shown that the alternative weighting is not significantly different [65, figure 2], and hence the simpler area-based weighting is used here.

The sparsity parameter controls the weighting given to the ℓ_1 term. When the sparsity parameter $\mu = 0$ the solutions to the CMM problem are the Laplacian eigenfunctions. A small μ prioritises the trace term. Recall that this is a discretisation of the Dirichlet energy and minimising it promotes smoothness which in the discrete case means the promotion of solutions which do not have extreme difference between values at neighbouring vertices. A large μ prioritises the ℓ_1 term, resulting in very sparse solutions, where there are extreme differences between the values at neighbouring vertices. Choosing a μ such that a solution is both sparse and Dirichlet energy-minimising results in a solution which focuses energy in local regions. (See [67] for a smooth-case discussion of compact support varying with μ , noting that their sparsity parameter is of the form $\frac{1}{\mu}$.) Selecting such a μ is not simple and usually involves some trial and error for each (N, k) pair, where N is a mesh and k is the desired number of modes. Figure 3.5 shows the effect of the sparsity parameter μ on the compressed manifold modes. Five modes were calculated via ADMM for the homer mesh. Note that in the $\frac{\mu}{n} = 1$ case the algorithm did not converge. For more on effect of the scaling of the ℓ_1 norm on the sparsity parameter see remark 3.2.14.

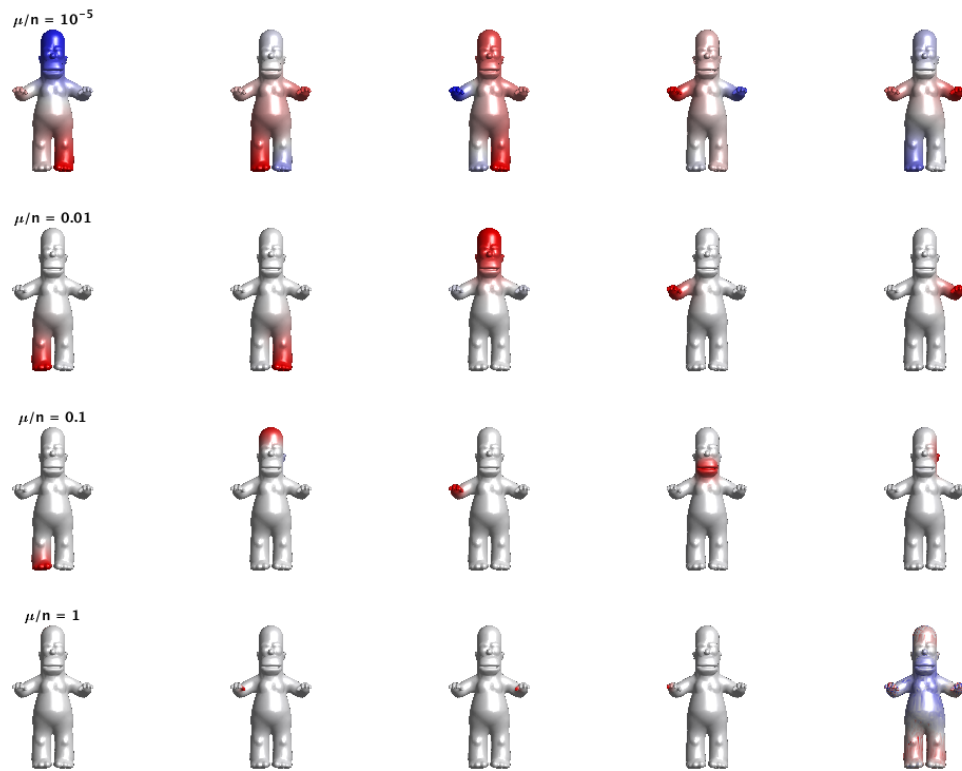


Figure 3.5: The effect of the sparsity parameter on compressed manifold modes, calculated via ADMM for the homer mesh.

The addition of the ℓ_1 term makes finding solutions more complicated. Unlike localised manifold harmonics there is no simple way to write $\|AX\|_1$ in the form $\text{tr}(X^T Q X)$ so the problem cannot be easily rearranged into an eigenproblem. Despite this, in [65] an eigenproblem-based formulation of the CMM problem was posed. Modes are calculated recursively; localisation is enforced via a potential matrix, much as in LMH or Hamilton eigenfunction problems; orthogonality with previously calculated modes is enforced via an ℓ_2 regularisation term. The paper is vague on the construction of the potential matrix used to control sparsity, so this method is not implemented here.

In [9] an algorithm for calculating compressed manifold modes via ADMM is given. The algorithm, including explicit solutions for the ADMM steps is given below. The algorithm has been generalised for the area-weighted ℓ_1 term.

To aid the splitting of the CMM problem, equation (3.2.13) is rewritten as

$$\arg \min_{\Psi} \text{tr}(\Psi^T W \Psi) + \mu \|A\Psi\|_1 + \iota(\Psi), \quad (3.2.14)$$

where $\iota : \mathbb{R}^{n \times k} \rightarrow \{0, \infty\}$ is the indicator function defined by

$$\iota(\Psi) = \begin{cases} 0, & \text{if } \Psi^T A \Psi = I, \\ \infty, & \text{otherwise.} \end{cases}$$

This can then be split as in the ADMM algorithm, with the $f(X)$ part given by $\iota(\Psi)$; the $g(Z)$ part is given by setting

$$Z = \begin{bmatrix} E \\ S \end{bmatrix}, \quad g(Z) = \begin{bmatrix} \text{tr}(E^T W E) \\ \mu \|AS\|_1 \end{bmatrix},$$

and the linear condition is given by

$$\begin{bmatrix} I \\ I \end{bmatrix} \Psi + \begin{bmatrix} -I & 0 \\ 0 & -I \end{bmatrix} \begin{bmatrix} E \\ S \end{bmatrix} = 0.$$

Given an initial Ψ_0 , set $E_0 = S_0 = \Psi_0$, $U_0^E = U_0^S = 0 \in \mathbb{R}^{n \times k}$. The iterative steps of

the algorithm are then given by

$$\text{The } \Psi \text{ step: } \Psi_{k+1} := \arg \min_{\Psi} \iota(\Psi) + \frac{\rho}{2} \left\| \begin{bmatrix} \Psi \\ \Psi \end{bmatrix} - \begin{bmatrix} E_k \\ S_k \end{bmatrix} + \begin{bmatrix} U_k^E \\ U_k^S \end{bmatrix} \right\|_F^2, \quad (3.2.15)$$

$$\text{The } E \text{ step: } E_{k+1} := \arg \min_E \text{tr}(E^T W E) + \frac{\rho}{2} \|\Psi_{k+1} - E + U_k^E\|_F^2, \quad (3.2.16)$$

$$\text{The } S \text{ step: } S_{k+1} := \arg \min_S \mu \|AS\|_1 + \frac{\rho}{2} \|\Psi_{k+1} - S + U_k^S\|_F^2, \quad (3.2.17)$$

$$\text{The } U \text{ step: } U_{k+1}^E := U_k^E + \Psi_{k+1} - E_{k+1},$$

$$U_{k+1}^S := U_k^S + \Psi_{k+1} - S_{k+1}.$$

The Ψ , E and S steps can all be solved via an explicit formula.

The Ψ Step: The optimisation in this can be rephrased as

$$\arg \min_{\Psi} \frac{\rho}{2} \left\| \begin{bmatrix} \Psi \\ \Psi \end{bmatrix} - \begin{bmatrix} E_k \\ S_k \end{bmatrix} + \begin{bmatrix} U_k^E \\ U_k^S \end{bmatrix} \right\|_F^2, \text{ subject to } \Psi^T A \Psi = I. \quad (3.2.18)$$

For ease of notation define $\hat{E} = E - U_k^E$, $\hat{S} = S - U_k^S$. First note that

$$\|\Psi - \frac{1}{2}(\hat{E} + \hat{S})\|_F^2 = \|\Psi\|_F^2 + \frac{1}{2}\|\hat{E} + \hat{S}\|_F^2 - \text{tr}(\Psi^T(\hat{E} + \hat{S})), \text{ by A.6.} \quad (3.2.19)$$

Then the norm can be rearranged:

$$\begin{aligned} \left\| \begin{bmatrix} \Psi \\ \Psi \end{bmatrix} - \begin{bmatrix} E_k \\ S_k \end{bmatrix} + \begin{bmatrix} U_k^E \\ U_k^S \end{bmatrix} \right\|_F^2 &= \|\Psi - \hat{E}\|_F^2 + \|\Psi - \hat{S}\|_F^2, \text{ by A.7,} \\ &= 2\|\Psi\|_F^2 + \|E\|_F^2 + \|S\|_F^2 - 2\text{tr}(\Psi^T(\hat{E} + \hat{S})), \text{ via A.6,} \\ &= 2\|\Psi - \frac{1}{2}(\hat{E} + \hat{S})\|_F^2 - \|\hat{E} + \hat{S}\|_F^2 + \|E\|_F^2 + \|S\|_F^2, \\ &\quad \text{via (3.2.19),} \\ &= 2\|\Psi - \frac{1}{2}(\hat{E} + \hat{S})\|_F^2 - 2\text{tr}(\hat{E}^T \hat{S}), \text{ again via A.6.} \end{aligned}$$

Therefore the Ψ step problem (3.2.18) can be rephrased as

$$\arg \min_{\Psi} \|\Psi - Y\|_F^2 \text{ subject to } \Psi^T A \Psi = I \quad (3.2.20)$$

by disregarding constant terms and scalars, and where $Y := \frac{1}{2}(E_k + S_k - U_k^E - U_k^S)$.

Let $A = I$ and Y be full rank. Then, problems of the form (3.2.20) are minimised by

$$\Psi = YVD^{-\frac{1}{2}}V^T \quad (3.2.21)$$

where VDV^T is the singular value decomposition of Y^TY [68, Theorem 1]. To generalise this for the $A \neq I$ case, let $LL^T = A$ be the Cholesky decomposition of A (see section 2.2) and substitute $X = L^T\Psi$ in to problem (3.2.20) to get

$$\arg \min_X \|(L^T)^{-1}X - Y\|_F^2 \text{ subject to } X^TX = I. \quad (3.2.22)$$

However the solution from [68] assumes $L = I$ and does not generalise for any diagonal matrix, as claimed in [9]. This error is noted in [66, section 7]. A similar error is made in [69, equation 5.18], assuming that $D\Phi = \Phi D$ for diagonal D .

A solution to equation (3.2.20) can be found in the $A = rI$ case. The Lagrangian equation is given by

$$\mathcal{L}(\Psi, \Lambda) = \text{tr}(\Psi^T\Psi) - 2\text{tr}(\Psi^TY) + \text{tr}(Y^TY) + \text{tr}(\Lambda\Psi^TA\Psi) - \text{tr}(\Lambda),$$

where Λ is a matrix of Lagrange multipliers. Differentiating with respect to Ψ and equating with zero gives

$$2\Psi + A\Psi\Lambda^T + A\Psi\Lambda = 2Y.$$

When $A = rI$, $r \in \mathbb{R}$, this can be rearranged to

$$\Psi(I + \frac{r}{2}(\Lambda + \Lambda^T)) = Y. \quad (3.2.23)$$

Then, since differentiating with respect to Λ gives $\Psi^TA\Psi$, the product Y^TAY is given by

$$\begin{aligned} rY^TY &= (I + \frac{r}{2}(\Lambda + \Lambda^T))^T\Psi^TA\Psi(I + \frac{r}{2}(\Lambda + \Lambda^T)) \\ &= (I + \frac{r}{2}(\Lambda + \Lambda^T))^T(I + \frac{r}{2}(\Lambda + \Lambda^T)) \\ &= (I + \frac{r}{2}(\Lambda + \Lambda^T))^2. \end{aligned}$$

That is,

$$\begin{aligned} \left(I + \frac{r}{2}(\Lambda + \Lambda^T)\right) &= \sqrt{r}(Y^T Y)^{\frac{1}{2}} \\ &= \sqrt{r}VD^{\frac{1}{2}}V^T \end{aligned}$$

where $VDV^T = Y^T Y$ is the singular value of $Y^T Y$. Then, returning to equation (3.2.23),

$$\begin{aligned} \Psi &= Y(\sqrt{r}VD^{\frac{1}{2}}V^T)^{-1} \\ &= \frac{1}{\sqrt{r}}YVD^{-\frac{1}{2}}V^T. \end{aligned} \tag{3.2.24}$$

Hence, it is possible to generalise for area matrices of the form rI , $r \in \mathbb{R}^+$, ($L = \sqrt{r}I$) as the scalar *will* commute with the matrices. This could be an intentional choice (as in example 2.5.17.(i)) but is unlikely to be the case from any other method of constructing the area matrix, e.g. example 2.5.17.(ii).

Let n be the number of vertices, and set $r = 1/n$. Then, when the variance of the elements of the area matrix is small (see figure E.4), the matrices $rY^T Y$ and $Y^T AY$ are almost equal. More formally, let $\varepsilon = \max_i\{|A_{ii} - r|\}$, then

$$\begin{aligned} (rY^T Y)_{ij} &= \frac{1}{n} \sum_l Y_{li} Y_{lj} \\ (Y^T AY)_{ij} &= \sum_l Y_{li} Y_{lj} A_{ii} \\ &\leq \sum_l Y_{li} Y_{lj} (r + \varepsilon) \\ &= \frac{1}{n} \sum_l Y_{li} Y_{lj} + \varepsilon n \sum_l Y_{li} Y_{lj} \\ &= (rY^T Y)_{ij} + n\varepsilon(rY^T Y)_{ij}. \end{aligned}$$

Therefore

$$\|rY^T Y - Y^T AY\|_F \leq \|\varepsilon n r Y^T Y\|_F = \varepsilon n \|rY^T Y\|_F, \tag{3.2.25}$$

which tends to zero as ε tends to zero. Let the singular values for $rY^T Y$ and $Y^T AY$ be denoted by σ_i and $\tilde{\sigma}_i$ respectively. Then combining equation (3.2.25) with

the application of results from matrix perturbation theory (Mirsky's theorem [70, theorem 4.7]) gives

$$\begin{aligned} \|\text{diag}(\sigma_1 - \tilde{\sigma}_1, \dots, \sigma_k - \tilde{\sigma}_k)\|_F &\leq \|rY^T Y - Y^T A Y\|_F \\ &\leq \varepsilon n \|rY^T Y\|_F. \end{aligned}$$

Therefore, in practice, the approximation $A = \frac{1}{n}I$ is used. The output matrix can be used to construct an A -orthogonal Ψ via application of Gram-Schmidt.

The E Step: The minimiser of (3.2.16) can be found by differentiating. Let $\hat{E} := \Psi_{k+1} + U_k^E$, then

$$\begin{aligned} E_{k+1} &= \arg \min_E \text{tr}(E^T W E) + \frac{\rho}{2} \|\hat{E} - E\|_F^2 \\ &= \arg \min_E \text{tr}(E^T W E) + \frac{\rho}{2} \left(\text{tr}(\hat{E}^T \hat{E}) + \text{tr}(E^T E) - 2 \text{tr}(\hat{E}^T E) \right). \end{aligned}$$

Differentiating and equating with zero gives

$$2WE + \frac{\rho}{2}(2E - 2\hat{E}) = 0$$

and so

$$\begin{aligned} (2W + \rho I)E &= \rho \hat{E} \\ E &= \rho(2W + \rho I)^{-1}(\Psi_{k+1} + U_k^E) \quad [9, \text{equation (17)}]. \end{aligned} \quad (3.2.26)$$

The S Step: The S step can be solved explicitly by using the soft thresholding operator, a specific case of proximal operator. This solution for $A = I$ is given in [9, (18)] and generalised here for the case where A is any diagonal matrix.

Definition 3.2.11: Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a real-valued function. The **proximal operator** $\text{prox}_f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is defined by

$$\text{prox}_f(v) = \arg \min_x f(x) + \frac{1}{2} \|x - v\|_2^2 \quad [71, \text{equation 1.1}]. \quad (3.2.27)$$

Definition 3.2.12: The **soft thresholding operator** $S_\kappa : \mathbb{R} \rightarrow \mathbb{R}$ is defined, for $a, \kappa \in \mathbb{R}$, to be [20, 4.4.3]

$$S_\kappa(a) = \begin{cases} a - \kappa & \text{if } a > \kappa \\ 0 & \text{if } |a| \leq \kappa \\ a + \kappa & \text{if } a < -\kappa. \end{cases} \quad (3.2.28)$$

The soft thresholding operator is the proximal operator of the absolute value function defined by $x \mapsto \kappa|x|$ for some $\kappa > 0$. Figure 3.6a shows the function $f(x) = \kappa|x| + \frac{1}{2}(x - a)^2$ for some values of a and figure 3.6b shows the value of the soft thresholding operator as a varies.

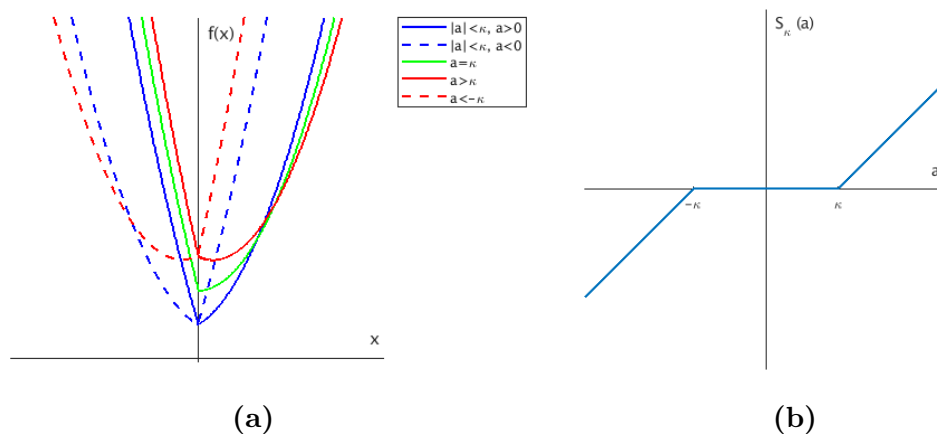


Figure 3.6: $S_\kappa(a) = \arg \min_x \kappa|x| + \frac{1}{2}(x - a)^2$

The following theorem generalises the soft thresholding operator for the matrix case with a (diagonally) weighted ℓ_1 term.

Theorem 3.2.13: Let B be an $n \times n$ diagonal matrix with $B_{ii} > 0$ and let $X, C \in \mathbb{R}^{n \times k}$. The minimisation problem

$$\arg \min_X \mu \|BX\|_1 + \sigma \|X - C\|_F^2$$

has solution X^* given element-wise by

$$X_{ij}^* = \text{sgn}(C_{ij}) \max \left\{ |C_{ij}| - \frac{\mu B_{ii}}{2\sigma}, 0 \right\}.$$

Proof. First consider a one-dimensional problem: let $x, c, b, \mu, \sigma \in \mathbb{R}$, with $b, \mu, \sigma > 0$ and consider the minimisation problem

$$\arg \min_x \mu|bx| + \sigma(x - c)^2. \quad (3.2.29)$$

Define $f(x) := \mu|bx| + \sigma(x - c)^2$. To find a minimiser, differentiate with respect to x and equate with zero.

Assume $bx > 0$ so $\text{sgn}(x) = 1$, resulting in

$$\begin{aligned} \frac{\partial f}{\partial x} &= \mu b + 2\sigma(x - c) = 0 \\ x &= -\frac{\mu b}{2\sigma} + c, \end{aligned}$$

therefore

$$\begin{aligned} 0 &< -\frac{\mu b}{2\sigma} + c, \text{ since } x > 0. \\ \frac{\mu b}{2\sigma} &< c. \end{aligned}$$

Assume $bx < 0$ so $\text{sgn}(x) = -1$, resulting in

$$\begin{aligned} \frac{\partial f}{\partial x} &= -\mu b + 2\sigma(x - c) = 0 \\ x &= \frac{\mu b}{2\sigma} + c, \end{aligned}$$

therefore

$$\begin{aligned} 0 &> \frac{\mu b}{2\sigma} + c, \text{ since } x < 0. \\ -\frac{\mu b}{2\sigma} &> c. \end{aligned}$$

That is, for $x \neq 0$ the value of c must be such that $|c| > \frac{\mu b}{2\sigma}$. Consider, then, the $x = 0$ case: when $|c| \leq \frac{\mu b}{2\sigma}$ it must be that $x = 0$. Note that, since $b, \mu, \sigma > 0$, $\text{sgn}(c) = \text{sgn}(x)$. In summary,

$$\begin{aligned} x &= \begin{cases} \text{sgn}(c) \left(|c| - \frac{\mu b}{2\sigma}\right), & |c| > \frac{\mu b}{2\sigma} \\ 0, & |c| \leq \frac{\mu b}{2\sigma} \end{cases} \\ &= \text{sgn}(c) \max \left\{ |c| - \frac{\mu b}{2\sigma}, 0 \right\}. \end{aligned} \quad (3.2.30)$$

Now consider the matrix case:

$$X^* = \arg \min_X \mu \|BX\|_1 + \sigma \|X - C\|_F^2$$

Expanding out the norms as sums of matrix elements gives

$$X^* = \arg \min_X \mu \sum_{ij} |(BX)_{ij}| + \sigma \operatorname{tr}((X - C)^T(X - C)).$$

Then, since B is diagonal and $\operatorname{tr}(A^T B) = \sum_{ij} A_{ij} B_{ij}$,

$$\begin{aligned} X^* &= \arg \min_X \mu \sum_{ij} |B_{ii} X_{ij}| + \sigma \sum_{ij} (X_{ij} - C_{ij})^2 \\ &= \arg \min_X \sum_{ij} (\mu |B_{ii} X_{ij}| + \sigma (X_{ij} - C_{ij})^2). \end{aligned}$$

Therefore, a minimiser X^* can be found by solving element-wise for X_{ij}^* by minimising $\mu |B_{ii} X_{ij}| + \sigma (X_{ij} - C_{ij})^2$, and so, via (3.2.30),

$$X_{ij}^* = \operatorname{sgn}(C_{ij}) \max \left\{ |C_{ij}| - \frac{\mu B_{ii}}{2\sigma}, 0 \right\}.$$

□

From this an explicit solution to the S step can be found, when A is diagonal. (If A is not diagonal, replace A with the lumped area matrix, see definition 2.5.18.) Let $\hat{S} := \Psi_{k+1} + U_k^S$. The minimiser of (3.2.17) is then given by the matrix S with elements defined by

$$S_{ij} = \operatorname{sgn}(\hat{S}_{ij}) \max \left\{ |\hat{S}_{ij}| - \frac{\mu A_{ii}}{\rho}, 0 \right\}. \quad (3.2.31)$$

To summarise, the algorithm for calculating compressed manifold modes via ADMM is given in algorithm 3.

Remark 3.2.14: Note that the only step of the ADMM algorithm influenced by the sparsity parameter μ is the S step. This is also the only step which is affected by the decision to include an area weighting in the ℓ_1 term. Hence, the choice to weight by area has a direct impact on the choice of sparsity parameter.

Algorithm 3 CMMS via ADMM

- 1: Given $\Psi_0 \in \mathbb{R}^{n \times k}$, $\mu \in \mathbb{R}$, area and weight matrices $A, W \in \mathbb{R}^{n \times n}$
- 2: Set $E_0 = S_0 = \Psi_0$, $U_0^E = U_0^S = 0 \in \mathbb{R}^{n \times k}$
- 3: Set regularisation parameter $\rho > 0$
- 4: **repeat**

$$Y \leftarrow \frac{1}{2}(E_k + S_k - U_k^E - U_k^S)$$

$VDV^T = Y^T Y$, the svd

$$\Psi_{k+1} \leftarrow \sqrt{n} Y V D^{-\frac{1}{2}} V^T$$

$$E_{k+1} \leftarrow \rho(2W + \rho I)^{-1}(\Psi_{k+1} + U_k^E)$$

$$(S_{k+1})_{ij} \leftarrow \text{sgn}((\Psi_{k+1} + U_k^S)_{ij}) \max\{ |(\Psi_{k+1} + U_k^S)_{ij}| - \frac{\mu A_{ii}}{\rho}, 0 \}$$

$$U_{k+1}^E \leftarrow U_k^E + \Psi_{k+1} - E_{k+1}$$

$$U_{k+1}^S \leftarrow U_k^S + \Psi_{k+1} - S_{k+1}$$

- 5: **until** convergence
-

Consider a set of modes constructed using sparsity parameter $\bar{\mu}$, and S step without area weighting – the S matrix produced without area weighting has entries

$$S_{ij} = \text{sgn}(\hat{S}_{ij}) \max\{|\hat{S}_{ij}| - \frac{\bar{\mu}}{\rho}, 0\}.$$

Compare this with equation (3.2.31) – to get the same S matrix it must be that

$$\frac{\mu}{\rho} A_{ii} = \frac{\bar{\mu}}{\rho}.$$

Then, using the $A_{ii} = \frac{1}{n}$ assumption as above,

$$\begin{aligned} \frac{\mu}{n} &= \bar{\mu} \\ \mu &= \bar{\mu}n. \end{aligned}$$

That is, to compare modes calculated with area weighting in the ℓ_1 term to modes calculated without area weighting the sparsity parameter should be scaled by the number of vertices.

At points, to aid comparisons, the value $\frac{\mu}{n}$ is fixed and μ calculated accordingly. (For example, see figure 3.5.)

Sequential ADMM for CMMs

In [72] a method for finding compressed manifold modes in a sequential way was proposed. The method is based on the above ADMM formulation, with an alteration in the Ψ step. In practice the algorithm is supplemented by an acceleration step.

The sequential CMM problem is as follows: Given k compressed manifold modes ψ_1, \dots, ψ_k , stored as columns of matrix Ψ_k , find the minimiser of

$$\arg \min_{\psi_{k+1}} \psi_{k+1}^T W \psi_{k+1} + \mu \|A\psi_{k+1}\|_1 \text{ subject to } \psi_{k+1}^T A \psi_{k+1} = 1, \quad \psi_{k+1}^T A \Psi_k = 0. \quad (3.2.32)$$

Following the method of [9] this can be rewritten as

$$\arg \min_{\psi_{k+1}} \psi_{k+1}^T W \psi_{k+1} + \mu \|A\psi_{k+1}\|_1 + \iota(\psi_{k+1}), \quad (3.2.33)$$

where $\iota : \mathbb{R}^n \rightarrow \{0, \infty\}$ is the indicator function defined by

$$\iota(\psi_{k+1}) = \begin{cases} 0, & \text{if } \psi_{k+1}^T A \psi_{k+1} = 1, \quad \psi_{k+1}^T A \Psi_k = 0, \\ \infty, & \text{otherwise.} \end{cases}$$

This can be split in the same way as above, with identical E and S step solutions (adjusted for vectors). The difference lies in the ψ_{k+1} step, as the orthogonality condition is made more complex by the condition that $\psi_{k+1}^T A \Psi_k = 0$.

There are several benefits to a sequential method: after constructing a set of k modes further modes can be calculated if required; the parameter μ can be calibrated using only one mode (or a small number of modes), a more reliable process than calculating an entire set only to reject them because of a poor μ choice; calculation times may be improved.

3.2.3 A canonical ordering for CMMs

As the solution to equation 3.2.10 can vary up to permutation of matrix columns it is desirable to have a method of ordering modes. To do this requires a set of ‘‘eigenvalues’’ – a number associated to each mode. A possible definition is provided in [66] for the case where the ℓ_1 norm does not include a scaling by the area matrix. A generalisation is provided here.

Recall that a matrix Ψ consisting of k CMMs satisfies

$$\arg \min_{\Psi} \text{tr}(\Psi^T W \Psi) + \mu \|A \Psi\|_1 \text{ subject to } \Psi^T A \Psi = I.$$

Then the Lagrange multiplier function is given by

$$\mathcal{L}(\Psi, \Lambda) = \text{tr}(\Psi^T W \Psi) + \mu \|A \Psi\|_1 - \text{tr}(\Lambda (X^T A X - I)).$$

(For proof see [66, lemma 4.1], generalised for A -orthogonality.) Differentiating with respect to Ψ and equating with zero gives

$$2W\Psi + \mu A \text{sgn}(\Psi) - 2A\Psi\Lambda = 0.$$

Then rearranging and multiplying on the left by Ψ gives

$$\Lambda = \Psi^T W \Psi + \frac{\mu}{2} \Psi^T A \operatorname{sgn}(\Psi).$$

Considering individual columns ψ_i of Ψ the number $\lambda_i := \Lambda_{ii}$ is given by

$$\lambda_i = \psi_i^T W \psi_i + \frac{\mu}{2} \psi_i^T A \operatorname{sgn}(\psi).$$

Note that $\psi_i^T A \operatorname{sgn}(\psi) = \|A\psi_i\|_1$ and so

$$\lambda_i = \psi_i^T W \psi_i + \frac{\mu}{2} \|A\psi_i\|_1,$$

and hence there is a number which can be associated to each mode, in a similar way to eigenvalues and eigenfunctions.

Definition 3.2.15: Let ψ be a compressed manifold mode calculated for a mesh M with Laplacian $L = A^{-1}W$ and sparsity parameter μ . The **compressed eigenvalue**, λ , associated to ψ is the number

$$\lambda = \psi^T W \psi + \frac{\mu}{2} \|A\psi\|_1.$$

Remark 3.2.16: From here it is assumed that compressed manifold modes $\{\psi_1, \dots, \psi_k\}$ are sorted such that the associated compressed eigenvalues $\lambda_1 \leq \dots \leq \lambda_k$.

Figure 3.7 shows a comparison of Laplacian eigenfunctions, localised manifold harmonics and compressed manifold modes on the horse0 mesh. The LMHs were calculated with reference to the reconstruction of the vertex positions.

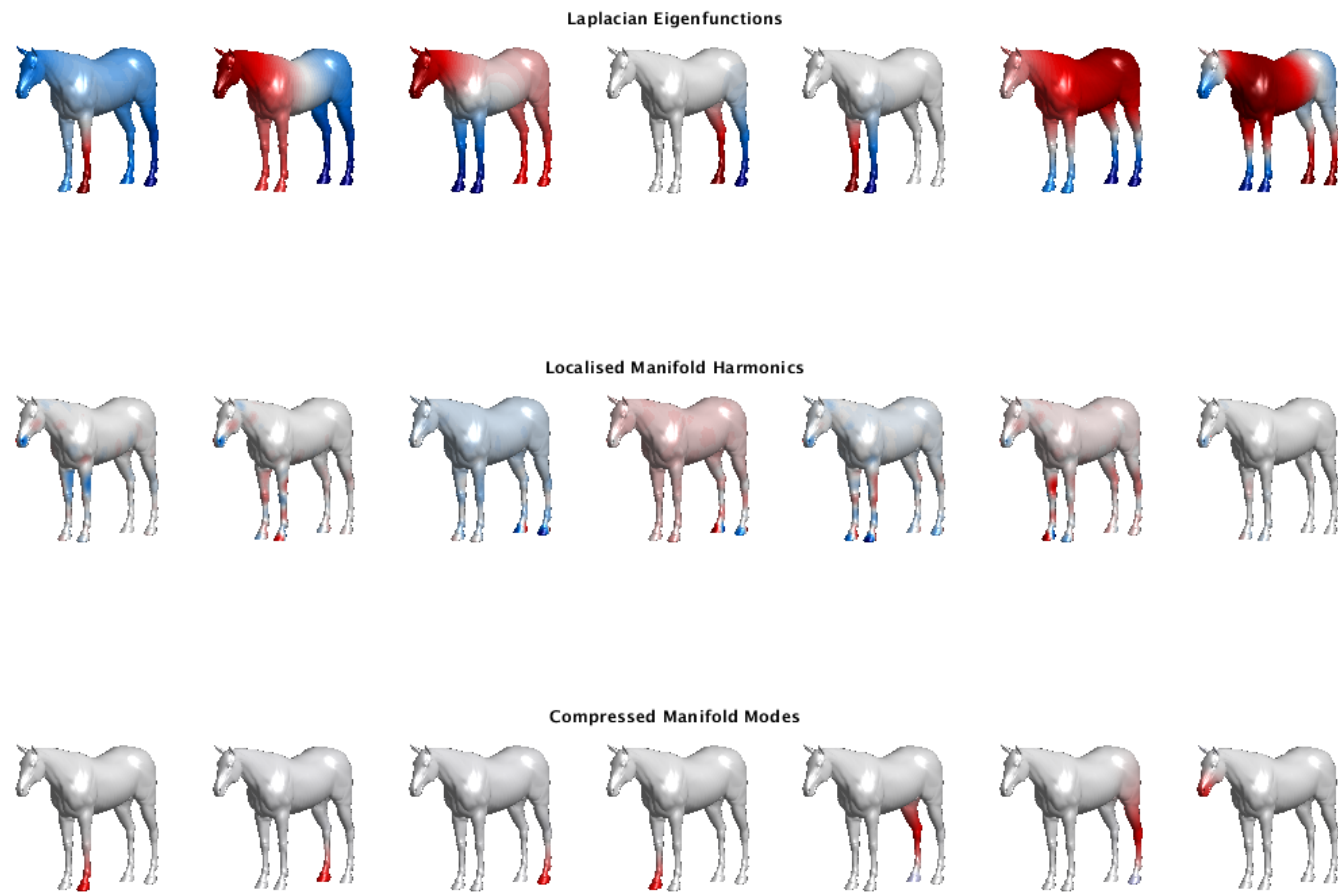


Figure 3.7: A comparison of basis functions on the horse0 mesh.

3.3 Generalised Manifold Harmonics

A general problem can be constructed, which encompasses the alternative basis methods of the previous sections.

Let Q and M be $n \times n$ matrices, and let Ψ be a $n \times k$ matrix. Consider the minimisation problem

$$\arg \min_{\Psi \in \mathbb{R}^{n \times k}} \text{tr}(\Psi^T Q \Psi) + \mu_f f(\Psi) \text{ subject to } \Psi^T M \Psi = I \quad (3.3.34)$$

where $f(\Psi)$ is some function of Ψ and $\mu_f \in \mathbb{R}$.

Definition 3.3.1: Solutions to equation (3.3.34) are called **generalised manifold harmonics**.

Examples 3.3.2: Generalised manifold harmonics have all the basis functions of sections 3.1 and 3.2 as specific cases: Let $L = A^{-1}W$ be a Laplacian and let $Q = W + \mu_V AV + \mu_\perp A \Phi \Phi^T A$, $M = A$. As before Φ_k denotes the matrix of the first k Laplacian eigenfunctions and $\mu_V, \mu_\perp \in \mathbb{R}$. Then the minimisation problem (3.3.34) results in

- (i) Laplacian eigenfunctions (equation (2.5.34)), when $\mu_V = \mu_\perp = \mu_f = 0$;
- (ii) localised manifold harmonics (equation (3.1.5)), when $\mu_f = 0$ and $V = \text{diag}(v)$, where v is the region penalty function defined in equation (3.1.2);
- (iii) Hamiltonian eigenfunctions (equation (3.1.9)), when $\mu_f = 0$, $\mu_\perp = 0$ and V is diagonal;
- (iv) compressed manifold modes (equation (3.2.13)), when $\mu_V = 0$, $\mu_\perp = 0$ and $f(\Psi) = \|A\Psi\|_1$.

Before considering the solutions to the generalised manifold harmonic minimisation problem first note the following lemma:

Lemma 3.3.3: Let A and B be $m \times n$ matrices, then $\sum_{ij} (A \odot B)_{ij} = \text{tr}(AB^T)$, where \odot denotes the Hadamard product.

Proof.

$$\begin{aligned} \sum_{ij} (A \odot B)_{ij} &= \sum_{ij} A_{ij}B_{ij}, \text{ by definition of the Hadamard product,} \\ &= \sum_{ij} A_{ij} (B^T)_{ji} \\ &= \sum_i (AB^T)_{ii} \\ &= \text{tr}(AB^T). \end{aligned}$$

□

Theorem 3.3.4: Solving the generalised manifold harmonic minimisation problem (3.3.34) is equivalent to solving

$$(Q + Q^T) \Psi + \mu_f \partial f(\Psi) = M^T \Psi \Lambda + M \Psi \Lambda^T, \text{ subject to } \Psi^T M \Psi = I, \quad (3.3.35)$$

where $\partial f(\Psi) := \frac{\partial f(\Psi)}{\partial \Psi}$, and Λ is a $k \times k$ matrix of unknowns.

Proof. Solve (3.3.34) for Ψ via Lagrange multipliers, first noting that the condition $\Psi^T M \Psi = I$ can be rephrased as k^2 conditions of the form $P_{ij} = 0$, where $P_{ij} = (\Psi^T M \Psi)_{ij} - \delta_{ij} = \psi_i^T M \psi_j - \delta_{ij}$, where ψ_i denotes the i -th column of Ψ . The Lagrange multiplier function is then given by

$$\mathcal{L}(\Psi, \Lambda) = \text{tr}(\Psi^T Q \Psi) + f(\Psi) - \sum_{ij} \Lambda_{ij} P_{ij},$$

where Λ is a $k \times k$ matrix of variables Λ_{ij} , so

$$\mathcal{L}(\Psi, \Lambda) = \text{tr}(\Psi^T Q \Psi) + \mu_f f(\Psi) - \sum_{ij} (\Lambda \odot P)_{ij},$$

where \odot denotes the Hadamard product,

$$\begin{aligned} &= \text{tr}(\Psi^T Q \Psi) + \mu_f f(\Psi) - \text{tr}(\Lambda P^T), \text{ by lemma 3.3.3} \\ &= \text{tr}(\Psi^T Q \Psi) + \mu_f f(\Psi) - \text{tr}(\Lambda (\Psi^T M^T \Psi - I)) \\ &= \text{tr}(\Psi^T Q \Psi) + \mu_f f(\Psi) - \text{tr}(\Lambda \Psi^T M^T \Psi) + \text{tr}(\Lambda). \end{aligned} \quad (3.3.36)$$

Then, denoting $\frac{\partial}{\partial X} f(X)$ by $\partial f(X)$,

$$\begin{aligned}\nabla_{\Psi} \mathcal{L}(\Psi, \Lambda) &= (Q + Q^T) \Psi + \mu_f \partial f(\Psi) - (M^T \Psi \Lambda + M \Psi \Lambda^T), \\ \nabla_{\Lambda} \mathcal{L}(\Psi, \Lambda) &= I_{k \times k} - \Psi^T M \Psi.\end{aligned}$$

This results in the following system of equations:

$$\begin{aligned}(1) \quad & (Q + Q^T) \Psi + \mu_f \partial f(\Psi) = M^T \Psi \Lambda + M \Psi \Lambda^T, \\ (2) \quad & \Psi^T M \Psi = I.\end{aligned}$$

□

Remark 3.3.5: For the specific case where where $f(\Psi) = \|A\Psi\|_1$ as in compressed manifold modes, $\partial f(\Psi) = \text{sgn}(A\Psi)$.

Proof. Let $f(\Psi) = \|A\Psi\|_1$. Then,

$$\begin{aligned}\partial f(\Psi) &= \frac{\partial f(\Psi)}{\partial \Psi} \\ &= \frac{\partial \|A\Psi\|_1}{\partial \Psi} \\ &= \frac{\partial}{\partial \Psi} \text{vec}(A\Psi)^T \text{vec}(\text{sgn}(A\Psi)), \text{ by lemma A.8,}\end{aligned}$$

and, by considering the entries of $\partial f(\Psi)$,

$$\begin{aligned}\left(\frac{\partial}{\partial \Psi} \text{vec}(A\Psi)^T \text{vec}(\text{sgn}(A\Psi)) \right)_{ij} &= \frac{\partial}{\partial \Psi_{ij}} \sum_{ij} (A\Psi)_{ij} \text{sgn}((A\Psi)_{ij}) \\ &= \text{sgn}((A\Psi)_{ij}).\end{aligned}$$

Therefore, $\partial f(\Psi) = \text{sgn}(A\Psi)$ and the problem becomes

$$\begin{aligned}(1) \quad & (Q + Q^T) \Psi + \text{sgn}(A\Psi) = M^T \Psi \Lambda + M \Psi \Lambda^T \\ (2) \quad & \Psi^T M \Psi = I\end{aligned}$$

which can be solved via ADMM (see section 2.4.2).

□

Aside from the dimensional constraints the matrices Q and M are free from restriction. First, note that $Q + Q^T$ is symmetric by construction, and so consider a symmetry constraint on M . This will simplify the system of equations obtained in theorem 3.3.4.

Lemma 3.3.6: Let $\mathcal{L}(\Psi, \Lambda)$ be the Lagrangian multiplier function where Λ is the matrix of λ_{ij} associated to the equations $\psi_i^T M \psi_j - \delta_{ij} = 0$. Then, if M is symmetric so is Λ .

Proof. If M is symmetric then $\psi_i^T M \psi_j = (\psi_i^T M \psi_j)^T$, since the product is a scalar. Therefore $\psi_i^T M \psi_j = \psi_j^T M^T \psi_i^T = \psi_j M \psi_i$. Then, since $\delta_{ij} = \delta_{ji}$ the equations $\psi_i^T M \psi_j - \delta_{ij} = 0$ and $\psi_j^T M \psi_i - \delta_{ji} = 0$ are equivalent, and are, therefore, associated to the same Lagrangian variable λ_{ij} . This results in the matrix Λ where $\lambda_{ij} = \lambda_{ji}$; that is, Λ is symmetric. \square

When the symmetric matrix M is given by an area matrix it will always be invertible (as area matrices are positive definite), so assume also that M is invertible. Then when the parameter $\mu_f = 0$ the generalised manifold harmonics problem can be solved as an eigenproblem.

Theorem 3.3.7: Let μ_f be equal to zero and M be symmetric and invertible. Then solution to the generalised manifold harmonic problem is given by the matrix with columns ψ_i , where ψ_i are the eigenfunctions satisfying $(Q + Q^T) \psi_i = \lambda_i M \psi_i$, and λ_i are the k smallest eigenvalues. The matrix Λ is given by the diagonal matrix with elements $\Lambda_{ii} = \lambda_i$, 0 elsewhere.

Proof. Define $\bar{Q} := \frac{1}{2}(Q + Q^T)$, then, via theorem 3.3.4, the solution Ψ is found by solving the system of equations given by

- (1) $\bar{Q}\Psi = M\Psi\Lambda$,
- (2) $\Psi^T M\Psi = I$,

The proof is in two parts. Let the generalised eigenvalue problem $\bar{Q}f = \lambda Mf$ be solved by eigenvalue/vector pairs (λ_i, ψ_i) .

(i) The matrix $\bar{\Psi} = (\psi_1, \dots, \psi_k)$ is a solution to $\bar{Q}\bar{\Psi} = M\bar{\Psi}\bar{\Lambda}$ where $\bar{\Lambda}$ is the matrix with diagonal entries λ_i and all other entries equal to zero.

To see this, consider $\bar{Q}\bar{\Psi} = M\bar{\Psi}\Lambda$, and solve for the values Λ_{ij} . First, multiply by $\bar{\Psi}^T$ on the left, giving

$$\bar{\Psi}^T \bar{Q} \bar{\Psi} = \bar{\Psi}^T M \bar{\Psi} \Lambda$$

then since $\bar{\Psi}^T M \bar{\Psi} = I$ (via lemmas A.2 and A.3), it must be that

$$\begin{aligned} \Lambda_{ij} &= \psi_i \bar{Q} \psi_j \\ &= \psi_i (\lambda_j M \psi_j), \text{ since } \bar{Q} \psi_j = \lambda_j M \psi_j, \text{ by the initial assumption,} \\ &= \lambda_j \psi_i M \psi_j \\ &= \lambda_j \delta_{ij}, \text{ since eigenfunctions of } \bar{Q}f = \lambda Mf \text{ are } M\text{-orthogonal.} \end{aligned}$$

That is, $\Lambda_{ij} = \lambda_i$ if $i = j$ and zero otherwise.

(ii) If Λ is diagonal with entries λ_i then the columns of Ψ are eigenvectors and λ_i are the corresponding eigenvalues.

To see this let Λ be a diagonal matrix, then consider $Q\Psi = M\Psi\Lambda$ where columns of Ψ are denoted by ψ_i . Then $(Q\psi_1, \dots, Q\psi_k) = (\lambda_1 M\psi_1, \dots, \lambda_k M\psi_k)$ so there are k equations of the form $Q\psi_i = \lambda_i M\psi_i$, which is the generalised eigenvalue problem, therefore (λ_i, ψ_i) are eigenvalue/vector pairs. \square

Corollary 3.3.8: Let M and \bar{Q} be symmetric $m \times m$ matrices, with M positive definite, and \bar{Q} positive semi-definite. Then the eigenvalues of the generalised eigenvalue problem $\bar{Q}f = \lambda Mf$ are non-negative.

Proof. By assumption, $x^T \bar{Q}x \geq 0$ for all non-zero x , and $y^T M y > 0$ for all non-zero y . Consider an eigenvector f such that $\bar{Q}f = \lambda Mf$. Multiplying by f^T on the left

gives

$$\begin{aligned} f^T \bar{Q} f &= \lambda f^T M f \\ \frac{f^T \bar{Q} f}{f^T M f} &= \lambda, \end{aligned}$$

which is non-negative for all f , as eigenvectors are non-zero. \square

Definition 3.3.9: Let $Q, M \in \mathbb{R}^{n \times n}$, with M symmetric and invertible. The solutions to the problem

$$\arg \min_{\Psi \in \mathbb{R}^{n \times k}} \text{tr}(\Psi^T Q \Psi) \quad \text{subject to } \Psi^T M \Psi = I \quad (3.3.37)$$

are called **generalised localised manifold harmonics** (GLMHs).

3.4 Discrete Isometry

Given two meshes with the same number of vertices and a bijection between the vertex sets, it is natural to ask if the meshes are in some way similar. In \mathbb{R}^2 , with the Euclidean metric, a set of points can be mapped via combinations of rotations, reflections and translations without altering the distances or angles between any of the points. These kinds of transformations are called *isometries* and the full set of isometries in \mathbb{R}^2 is given by all rotations, reflections, translations, combinations of reflections and translations, and the identity map. These types of transformations are often referred to as *rigid motions*. The notion of isometry becomes more complicated when applied to surfaces, as the full set of isometries is not equal to the set of rigid motions – there are more. To see this in an informal way, consider deforming a piece of paper by adding a twist. The twisted paper is isometric to the original sheet, but twisting is not a rigid motion.

The ideas of isometries between shapes are key to problems where the aim is to find a map between shapes. However, in much of the shape matching literature the language of an isometry between meshes is used, without giving a formal definition of what is really meant. Informally, it is assumed that an isometry between meshes

takes faces to faces with the same shape and size. This assumption is used implicitly in much of the existing geometry processing literature. The focus in applications is often finding a map between meshes which are “near-isometric”, for example two meshes representing the same animal in different poses (see figure 5.1). The following list gives examples of occurrences of this informal assumption:

- Results are proved for isometries in the smooth case and presumed to carry across to the “near-isometric” discrete case without issue [73],[5],[74];
- Ambiguous terminology – does “shape” mean manifold or mesh? [73],[75]; use of “isometric” to mean “near-isometric” or “ ε -isometric” [76],[77, section 3.4]?
- No reference to any definitions [78],[79],[12],[80],[10].

It is usually implied that an isometry between meshes is a map which preserves geodesic distance between points, and this is the primary method of evaluating quality of a point-to-point match [5],[78],[75],[81]. However, calculating geodesics on meshes is demanding, and approximations can be calculated in a variety of ways, see [82],[83],[84],[85].

In this section the definition of a discrete isometry between meshes is constructed by analogy to the definition of an isometry between Riemannian manifolds, and is related to the notion that an isomorphism between simple graphs can be represented as a permutation [86, p.158]. This formulation of a discrete isometry is touched upon in the proof of theorem 1 in the appendix of [87]. Here the assertion that an isometry between meshes requires area and weight matrices to be equal is corrected by inclusion of the permutation. Similarly, [88] uses an underlying graph to match sets of voxels, recognising that isometry can be represented as a permutation of the adjacency matrix, and constructing a matching based on the eigenbasis of the adjacency matrix, with reference to [89].

Definition 3.4.1: Let (\mathcal{N}, g) and (\mathcal{P}, h) be Riemannian manifolds and let $T : \mathcal{N} \rightarrow \mathcal{P}$ be a diffeomorphism. Then T is called an **isometry** if

$$g = T^*h,$$

where T^*h denotes the pullback of h by T . The manifolds \mathcal{N} and \mathcal{P} are said to be **isometric**.

To discretise this definition a sufficient and necessary condition is exploited. This first requires the following definitions:

Definition 3.4.2: Let (\mathcal{N}, g) be a Riemannian manifold and let f, \bar{f} be functions $\mathcal{N} \rightarrow \mathbb{R}$. Then the **L² inner product** is defined to be

$$\langle f, \bar{f} \rangle_{L^2} := \int_{\mathcal{M}} f \bar{f} d\mu_{\mathbb{R}} \quad (3.4.38)$$

where $d\mu$ is the volume form.

Definition 3.4.3: Let (\mathcal{N}, g) be a Riemannian manifold and let f, \bar{f} be functions $\mathcal{N} \rightarrow \mathbb{R}$. Then the **conformal inner product** is defined to be

$$\langle f, \bar{f} \rangle_{\text{conf}}^{\mathcal{N}} = \int_{\mathcal{N}} \langle \nabla f, \nabla \bar{f} \rangle_{\ell_2}^{\mathcal{N}} d\mu,$$

where $d\mu$ is the volume form. (Recall the ℓ_2 inner product from equation ??.)

Definition 3.4.4: Let (\mathcal{N}, g) and (\mathcal{P}, h) be Riemannian manifolds and let $T : \mathcal{N} \rightarrow \mathcal{P}$ be a diffeomorphism. Then T is **area-preserving** if

$$\langle f, \bar{f} \rangle_{L^2}^{\mathcal{N}} = \langle f \circ T^{-1}, \bar{f} \circ T^{-1} \rangle_{L^2}^{\mathcal{P}},$$

where f, \bar{f} are functions $\mathcal{N} \rightarrow \mathbb{R}$ and T is **conformal** if

$$\langle f, \bar{f} \rangle_{\text{conf}}^{\mathcal{N}} = \langle f \circ T^{-1}, \bar{f} \circ T^{-1} \rangle_{\text{conf}}^{\mathcal{P}},$$

where f, \bar{f} are functions $\mathcal{N} \rightarrow \mathbb{R}$.

Theorem 3.4.5: Let (\mathcal{N}, g) and (\mathcal{P}, h) be Riemannian manifolds and let $T : \mathcal{N} \rightarrow \mathcal{P}$ be a diffeomorphism. Then T is an isometry if and only if T is area-preserving and conformal. [90, chapter 8, theorem 5]

Note the similarity between the Dirichlet energy (equation (2.5.8)) and the conformal inner product. The conformal inner product can be discretised by following an

analogous chain of reasoning (see equation (2.5.10)). That is, for mesh N constructed from \mathcal{N} ,

$$\langle f, \bar{f} \rangle_{\text{conf}}^{\mathcal{N}} \approx \mathbf{f}^T W_N \bar{\mathbf{f}}, \quad (3.4.39)$$

where W_N is a weight matrix for mesh N constructed from \mathcal{N} and the vectors $\mathbf{f}, \bar{\mathbf{f}}$ are constructed by evaluating the functions f, \bar{f} on the vertices of N . Similarly, via equation (2.5.3), the L_2 inner product discretises to

$$\langle f, \bar{f} \rangle_{L_2}^{\mathcal{N}} \approx \mathbf{f}^T A_N \bar{\mathbf{f}}, \quad (3.4.40)$$

where A_N is an area matrix for N .

Let \mathcal{N} and \mathcal{P} be manifolds, with bijection $T : \mathcal{N} \rightarrow \mathcal{P}$. Let N be a mesh constructed from \mathcal{N} , then a vertex set for a mesh P constructed from \mathcal{P} can be obtained via the restriction of T to \mathcal{V}_N . In this chapter we consider the map between vertex sets ($T : \mathcal{V}_N \rightarrow \mathcal{V}_P$). In later chapters, since the distinction between meshes and manifolds is clear, this restriction is denoted by $T : N \rightarrow P$.

Definition 3.4.6: Let N be a mesh with vertex set $\mathcal{V}_N = \{x_1, \dots, x_n\}$, let P be a mesh with vertex set $\mathcal{V}_P = \{y_1, \dots, y_n\}$ and let $T : \mathcal{V}_N \rightarrow \mathcal{V}_P$ be a bijection. Then the **permutation matrix representing \mathbf{T}** , denoted by Π , is an $n \times n$ orthogonal matrix with entries defined by

$$\Pi_{ij} = \begin{cases} 1, & \text{if } T(x_i) = y_j, \\ 0, & \text{otherwise.} \end{cases}$$

Then composed function $f \circ T^{-1}$ discretises to $\Pi^T \mathbf{f}$. Combining this with equations (3.4.39) and (3.4.40) the notions of area-preserving and conformal can be carried over to the discrete setting. That is, T is area-preserving if

$$\mathbf{f}^T A_N \bar{\mathbf{f}} = \mathbf{f}^T \Pi A_P \Pi^T \bar{\mathbf{f}}, \quad (3.4.41)$$

and conformal if

$$\mathbf{f}^T W_N \bar{\mathbf{f}} = \mathbf{f}^T \Pi W_P \Pi^T \bar{\mathbf{f}}, \quad (3.4.42)$$

for all vectors \mathbf{f} and $\bar{\mathbf{f}}$ representing functions on N . Equation (3.4.43) is equivalent to saying that $A_N = \Pi A_P \Pi^T$ and equation (3.4.44) is equivalent to saying that $W_N = \Pi W_P \Pi^T$ and so the area-preserving and conformal properties are defined for a discrete isometry as follows.

Definition 3.4.7: Let N and P be meshes with area and weight matrices A_N, W_N and A_P, W_P respectively. Let $T : \mathcal{V}_N \rightarrow \mathcal{V}_P$ be a bijection represented by matrix Π . Then T is **area-preserving** if

$$A_N = \Pi A_P \Pi^T, \quad (3.4.43)$$

and **conformal** if

$$W_N = \Pi W_P \Pi^T. \quad (3.4.44)$$

Definition 3.4.8: Let N and P be meshes with area and weight matrices A_N, W_N and A_P, W_P respectively. Let $T : \mathcal{V}_N \rightarrow \mathcal{V}_P$ be a bijection represented by matrix Π . Then T is a **discrete isometry** if

$$A_N = \Pi A_P \Pi^T \quad \text{and} \quad W_N = \Pi W_P \Pi^T.$$

Meshes N and P are said to be **isometric**.

The term *isometry* is used rather than *discrete isometry* since it is clear when maps are between meshes rather than manifolds.

Remark 3.4.9: That $T : \mathcal{V}_N \rightarrow \mathcal{V}_P$ is an isometry depends on the choice of area and weight matrices.

To see this consider the following simple example.

Example 3.4.10: The meshes M and N are the prisms shown in the first column of figure 3.8. The mesh N is obtained from M by affine transform. The bijection T between the vertex sets is given by the identity map, so $\Pi = I$. Assume that the uniform area matrix is being used (see example 2.5.17.(i)), so $A_N = A_P$. Then for T to be an isometry it must be that $W_N = W_P$. To display a weight matrix W as

function on a mesh each vertex v_i is given a colour based on the value of W_{ii} . If the weight matrices are different the colours of the corresponding vertices will not match. The second column of figure 3.8 shows the meshes coloured according to the weight matrix of the cot Laplacian (example 2.5.22.(i)) and the third column shows the meshes coloured according to the weight matrix of the graph Laplacian (example 2.5.22.(ii)). From these colourings it is obvious that under the first choice of weight matrix T is not an isometry, but under the second choice of weight matrix T is an isometry. The weight matrix of the graph Laplacian is blind to changes in angle, and it is clear that angles have been altered by the transformation. An alternative choice of area matrix helps protect against this, The fourth column shows vertices coloured by their mixed Voronoi cell weight (i.e. the area matrix is constructed as in example 2.5.17.(iii)). Since the vertex colours do not correspond, $A_N \neq \Pi A_P \Pi^T$ and the map T is not an isometry.

It is known that in the manifold case that the Laplacian commutes with isometries [91]. Using the above definitions this is easy to show for the discrete case.

Proposition 3.4.11: Let N and P be meshes with area and weight matrices A_N, W_N and A_P, W_P respectively. Let $T : \mathcal{V}_N \rightarrow \mathcal{V}_P$ be a bijection represented by matrix Π . Then

$$L_N = \Pi L_P \Pi^T.$$

Proof. First note that since Π is orthogonal, $A_P = \Pi^T A_N \Pi$ so

$$\begin{aligned} A_P^{-1} &= (\Pi^T A_N \Pi)^{-1} \\ &= \Pi^T A_N^{-1} \Pi. \end{aligned} \tag{3.4.45}$$

Then, since $W_P = \Pi^T W_N \Pi$,

$$\begin{aligned} L_P &= A_P^{-1} W_P \\ &= \Pi^T A_N^{-1} \Pi \Pi^T W_N \Pi, \text{ via equation (3.4.45),} \\ &= \Pi^T A_N^{-1} W_N \Pi \\ &= \Pi^T L_N \Pi. \end{aligned}$$

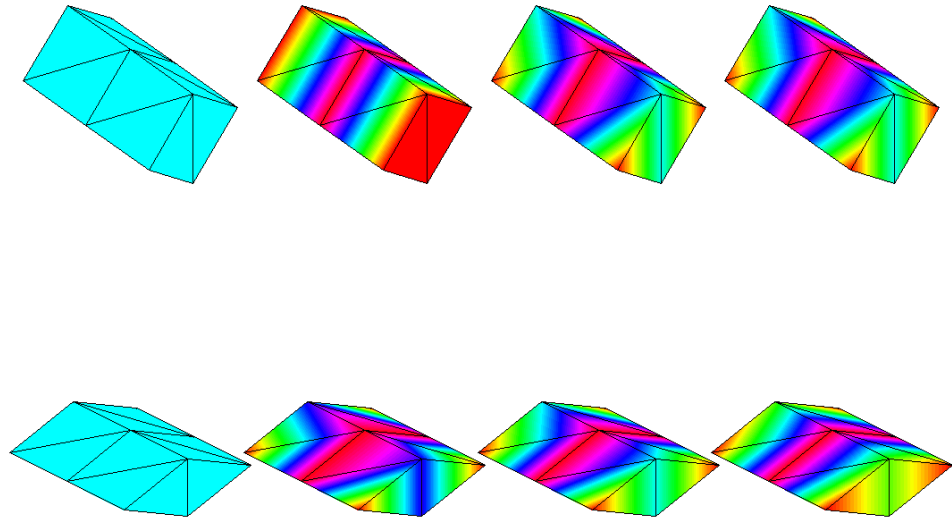


Figure 3.8: Choice of weight matrix affects isometry.

□

Proposition 3.4.12: Let $T : \mathcal{V}_N \rightarrow \mathcal{V}_P$ be an isometry represented by permutation matrix Π . Let $\{\phi_i\}$ be the set of eigenfunctions of the Laplacian L_N with eigenvalues λ_i . Then the eigenpairs of L_P are given by $(\Pi^T \phi_i, \lambda_i)$.

Proof. Since, via proposition 3.4.11, $L_N = \Pi L_P \Pi^T$,

$$\begin{aligned}
 L_P \Pi^T \phi_i &= \Pi^T L_N \Pi \Pi^T \phi_i \\
 &= \Pi^T L_N \phi_i \\
 &= \Pi^T \lambda_i \phi_i \\
 &= \lambda_i \Pi^T \phi_i.
 \end{aligned}$$

□

3.5 GMHs and Isometry

One of the important properties of the Laplacian eigenfunctions is that they commute with isometry. Although stated in [53, section 5] that localised manifold harmonics commute with isometry in the smooth case, it is not proved and we are unaware of any results about isometry for Hamiltonian eigenfunctions or compressed manifold modes. In this section it is proved that the alternative basis types presented in sections 3.1 and 3.2 commute with discrete isometry. Finally, the requirements for GMHs to commute with isometry are described.

Proposition 3.5.1: Let $T : \mathcal{V}_N \rightarrow \mathcal{V}_P$ be an isometry between meshes, represented by permutation matrix Π . Let $Q_N = W_N + \mu_V A_N V_N + \mu_\perp A_N \Phi_N \Phi_N^T A_N$ where V_N is a symmetric potential matrix for N and A_N, W_N are the area and weight matrices. Let Ψ_N be the matrix of GLMHs satisfying

$$\arg \min_{\Psi} \text{tr}(\Psi^T W_N \Psi) \quad \text{subject to} \quad \Psi^T A_N \Psi.$$

Then, if $V_P = \Pi^T V_N \Pi$ the GLMHs commute with isometry. That is, the GLMHs Ψ_P on P are given by $\Psi_P = \Pi^T \Psi_N$.

Proof. From theorem 3.3.7, and since Q_N is symmetric, the GLMHs on N are eigenfunctions Ψ_N satisfying

$$Q_N \Psi_N = A_N \Psi_N \Lambda_N, \tag{3.5.46}$$

where Λ_N is a diagonal matrix of eigenfunctions.

Since T is an isometry

$$A_N = \Pi A_P \Pi^T, \quad W_N = \Pi W_P \Pi^T, \quad \text{and} \quad \Phi_N = \Pi \Phi_P.$$

Expanding Q_N via these expressions, along with the assumption that $V_N = \Pi V_P \Pi^T$, gives

$$\begin{aligned} Q_N &= \Pi W_P \Pi^T + \mu_V \Pi A_P \Pi^T \Pi V_P \Pi^T + \mu_\perp \Pi A_P \Pi^T \Pi \Phi_P \Phi_P^T \Pi^T \Pi A_P \Pi^T \\ &= \Pi (W_P + \mu_V A_P V_P + \mu_\perp A_P \Phi_P \Phi_P^T A_P) \Pi^T \\ &= \Pi Q_P \Pi^T. \end{aligned}$$

Therefore, since $A_N^{-1} = \Pi A_P^{-1} \Pi^T$, equation (3.5.46) rearranges to become

$$\Pi A_P^{-1} \Pi^T \Pi Q_P \Pi^T \Psi_N = \Psi_N \Lambda_N.$$

So

$$A_P^{-1} Q_P \Pi^T \Psi_N = \Pi^T \Psi_N \Lambda_N$$

and hence the GLMHs on P are given by $\Psi_P = \Pi^T \Psi_N$. \square

The above proposition encompasses both localised manifold harmonics and Hamiltonian eigenfunctions. To consider compressed manifold modes first consider the additional function which prevents solution as an eigenproblem.

Definition 3.5.2: Let $f : \mathbb{R}^{n \times k} \rightarrow \mathbb{R}$ be a real-valued function, and let Π be an $n \times n$ permutation matrix. Then f is **permutation invariant** if, for any $M \in \mathbb{R}^{n \times k}$,

$$f(\Pi M) = f(M).$$

Examples 3.5.3:

- (i) The ℓ_1 norm is permutation invariant since multiplication on the left by a permutation matrix permutes the rows of the matrix M and the entries of M are in one-to-one correspondence to the entries of ΠM . Therefore

$$\|\Pi M\|_1 = \|M\|_1. \quad (3.5.47)$$

- (ii) The Frobenius norm is permutation invariant. This follows from lemma A.10, since permutation matrices are orthogonal.

Another important fact to note is that when Π is a fixed permutation matrix, independent of the function being minimised,

$$\arg \min_{\Pi Y} f(Y) = \Pi \left(\arg \min_Y f(Y) \right) \quad (3.5.48)$$

Proposition 3.5.4: Compressed manifold modes commute with isometry. That is, given isometry $T : \mathcal{V}_N \rightarrow \mathcal{V}_P$ represented by Π and CMMs Ψ_N on N , the CMMs on P are given by $\Pi^T \Psi_N$.

Proof. Let N have area matrix A_N and weight matrix W_N , and let P have area matrix A_P and weight matrix W_P . The compressed manifold modes on N are given by

$$\Psi_N = \arg \min_{X \in \mathbb{R}^{n \times k}} \text{tr}(X^T W_N X) + \mu \|A_N X\|_1 \text{ subject to } X^T A_N X = I,$$

and the compressed manifold modes on P are given by

$$\Psi_P = \arg \min_{Y \in \mathbb{R}^{n \times k}} \text{tr}(Y^T W_P Y) + \mu \|A_P Y\|_1 \text{ subject to } Y^T A_P Y = I.$$

Since T is an isometry $A_N = \Pi A_P \Pi^T$ and $W_N = \Pi W_P \Pi^T$ the CMMs on N are given by

$$\Psi_N = \arg \min_{X \in \mathbb{R}^{n \times k}} \text{tr}(X^T \Pi W_P \Pi^T X) + \mu \|\Pi A_P \Pi^T X\|_1 \text{ subject to } X^T \Pi A_P \Pi^T X = I.$$

As the ℓ_1 norm is permutation invariant this is

$$\Psi_N = \arg \min_{X \in \mathbb{R}^{n \times k}} \text{tr}(X^T \Pi W_P \Pi^T X) + \mu \|A_P \Pi^T X\|_1 \text{ subject to } X^T \Pi A_P \Pi^T X = I.$$

Substitute $Y = \Pi^T X$, giving

$$\Psi_N = \arg \min_{\Pi Y \in \mathbb{R}^{n \times k}} \text{tr}(Y^T W_P Y) + \mu \|A_P Y\|_1 \text{ subject to } Y^T A_P Y = I.$$

Then since Π is a fixed permutation, making use of equation (3.5.48),

$$\Pi^T \Psi_N = \arg \min_{Y \in \mathbb{R}^{n \times k}} \text{tr}(Y^T W_P Y) + \mu \|A_P Y\|_1 \text{ subject to } Y^T A_P Y = I.$$

That is, $\Psi_P = \Pi^T \Psi_N$. □

To prove an isometry result for generalised manifold harmonics note that the previous results have relied upon the relationships between A_N and A_P , W_N and W_P , resulting from the isometry.

Theorem 3.5.5: Let $T : \mathcal{V}_N \rightarrow \mathcal{V}_P$ be an isometry, represented by permutation matrix Π . Let Ψ_P be generalised manifold harmonics on N such that

$$\Psi_N = \arg \min_{X \in \mathbb{R}^{n \times k}} \text{tr}(X^T Q_N X) + \mu_f f(X) \text{ subject to } X^T M_N X = I,$$

where $Q_N, M_N \in \mathbb{R}^{n \times k}$ and f is a permutation invariant real-valued function. Then if $M_P = \Pi^T M_N \Pi$ and $Q_P = \Pi^T Q_N \Pi$, the GMHs commute with isometry.

Proof. Let the GMHs on P be given by

$$\Psi_P = \arg \min_{Y \in \mathbb{R}^{n \times k}} \text{tr}(Y^T Q_P Y) + \mu_f f(Y) \text{ subject to } Y^T M_P Y = I.$$

Since it is assumed that $M_N = \Pi M_P \Pi^T$ and $Q_N = \Pi Q_P \Pi^T$, the GMHs on N are given by

$$\Psi_N = \arg \min_{X \in \mathbb{R}^{n \times k}} \text{tr}(X^T \Pi Q_P \Pi^T X) + \mu_f f(X) \text{ subject to } X^T \Pi M_P \Pi^T X = I.$$

Substitute $Y = \Pi^T X$, giving

$$\Psi_N = \arg \min_{\Pi Y \in \mathbb{R}^{n \times k}} \text{tr}(Y^T Q_P Y) + \mu_f f(\Pi Y) \text{ subject to } Y^T M_P Y = I.$$

Since f is permutation invariant, this is

$$\Psi_N = \arg \min_{\Pi Y \in \mathbb{R}^{n \times k}} \text{tr}(Y^T Q_P Y) + \mu_f f(Y) \text{ subject to } Y^T M_P Y = I.$$

Then, via equation (3.5.48),

$$\Pi^T \Psi_N = \arg \min_{Y \in \mathbb{R}^{n \times k}} \text{tr}(Y^T Q_P Y) + \mu_f f(Y) \text{ subject to } Y^T M_P Y = I$$

and hence, $\Psi_P = \Pi^T \Psi_N$. □

Of course, propositions 3.4.12, 3.5.1 and 3.5.4 are all special cases of theorem 3.5.5.

3.6 Summary and Future Work

A review of existing constructions of basis functions for $\mathcal{F}(N, \mathbb{R})$ with specific properties was given. In the section about compressed manifold modes (section 3.2) alterations were made to the solution of [9], to allow for the scaling by area matrix in the ℓ_1 norm, and to allow for an orthogonality constraint of the form $\Psi^T A \Psi = I$. To achieve this, it is assumed that the area matrix $A = \frac{1}{n}I$. An alteration was also made to the method of [66] for calculating compressed eigenvalues, to allow for the scaling by area matrix in the ℓ_1 norm.

The alternative basis functions methods were combined into a general problem, with solutions called generalised manifold harmonics (GMHs). The conditions required to find GMHs as solutions to a generalised eigenvalue problem have been described.

A new definition for a discrete isometry between meshes has been given, as an analogy to the necessary and sufficient conditions for an isometry between Riemannian manifolds. Using this definition it has been proved that the specific cases of Laplacian eigenfunctions, localised manifold harmonics, Hamiltonian eigenfunctions and compressed manifold modes commute with discrete isometry. Finally, the conditions required for GMHs to commute with isometry were discussed.

The sparsity parameter μ , used in the calculation of compressed manifold modes is difficult to choose, and work is required to find a good method of selection, based on the target sparsity. A potential start point would be a discretisation of [67], which requires a formal definition for *compact support* for a mesh.

A map T between metric spaces (X, d_X) and (Y, d_Y) is an ε -isometry if for $x, \bar{x} \in X$, $|d_Y(T(x), T(\bar{x})) - d_X(x, \bar{x})| < \varepsilon$ and for every $y \in Y$ there exists $x \in X$ with $d(T(x), y) \leq \varepsilon$. Our definition of discrete isometry avoids reference to geodesic distance on the mesh. Is there a way to adapt this to define ε -isometry, in a way which still tells us something about the distance between points?

Chapter 4

A Comparison Of Basis Methods For Reconstructing Functions

The shape matching method of functional maps [5] relies on a set of functions believed to correspond on two meshes. These functions are then represented in a truncated basis of Laplacian eigenfunctions, via a matrix of coefficients. As discussed in chapter 3, basis truncation leads to a loss of information, and it is important that functions can be reconstructed accurately. (For background on the functional maps framework see section 5.1.)

This chapter reconstructs a variety of functions using the basis functions of sections 3.1 and 3.2. The function types include those regularly used when calculating functional maps. Two reconstruction methods are evaluated, for speed and accuracy. These are the least squares solution and a matrix product based on a reconstruction method posed in [9].

Functions and bases have been calculated for a set of 26 meshes. The meshes used appear in table B.1, marked by †. Information about the construction of the functions and bases is provided, including calculation times and details about the orthogonality of the localised manifold harmonics.

A weighted error for measuring function reconstruction is introduced. Using this, we reach conclusions about which function and basis types should be avoided.

The difference between spaces spanned by the different basis functions are considered and it is noted that Laplacian eigenfunctions and compressed manifold modes span similar subspaces for some meshes. To investigate this further, the ability to reconstruct one basis in another is examined. This raises questions about the choice of the sparsity parameter μ .

4.1 Function choice

The theory of functional maps uses indicator functions on each vertex to allow reconstruction of a point-to-point map (see section 5.1.4) but to use such a large set of functions is computationally expensive. In many existing works (e.g. [5],[92],[13],[11]) the heat and wave kernel signatures are used as, when taken over various time samples, they provide information about both local areas and the mesh as a whole. This is referred to as the *multi-scale property*. Other works use indicator functions on segments (e.g. [10],[75]). Here the types of functions used in the later experiments are described.

Definition 4.1.1: A function $f_p : N \rightarrow \mathbb{R}$ is called **point-based** if it depends explicitly on a specified point $p \in N$. The function f_p is **based at p** and the point p is called a **landmark**.

Examples 4.1.2:

- (i) A function which measures distance between any vertex v and the specified vertex p , with

$$f_p(v) = d(v, p).$$

The distance function could be the geodesic distance between the points, or the Euclidean distance, etc.

- (ii) The indicator function of the point p , defined by

$$f_p(v) = \begin{cases} 0, & x \neq p \\ 1, & x = p. \end{cases}$$

Landmarks can be chosen manually (see [93]) or as points which satisfy some condition (see [94] where landmarks are chosen to be local maxima of the heat kernel signature). As a quick method of selecting landmark points the vertex coordinate functions can be exploited.

Definition 4.1.3: Let p be a vertex written as a point in \mathbb{R}^3 , $p = (p_1, p_2, p_3)$. The vertex p is called **extremal** if

$$p_i \geq v_i \quad \text{or} \quad p_i \leq v_i \quad i \in \{1, 2, 3\}$$

for all vertices $v \in N$.

Definition 4.1.4: Define for $i \in \{1, 2, 3\}$

$$a_i := \max_{p \in N} p_i, \quad b_i := \min_{p \in N} p_i, \quad r_i := \frac{a_i + b_i}{2}.$$

Then a vertex p is called **central** if

$$|p_i - r_i| \leq |v_i - r_i| \quad \forall i \in \{1, 2, 3\}$$

for all vertices $v \in N$.

Figure 4.1 shows the central and extremal points on three meshes.

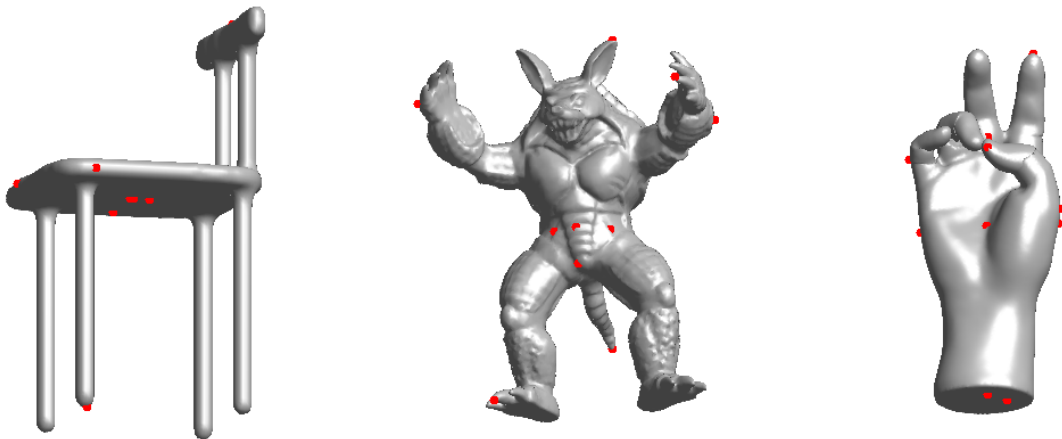


Figure 4.1: Extremal and central points on the chair 1, armadillo and hand 2 meshes.

Point-based functions are constructed for the set of points \mathcal{P} containing all extremal and central points. The following point-based functions are used:

GEODISTS: Define a function $f : N \rightarrow \mathbb{R}$ based at p by

$$f(v) = d(p, v),$$

where the distance function d is the shortest path along edges between the vertices p and v . Such a path can be found via the Dijkstra algorithm for weighted graphs [95]. As the shortest path across the mesh may not be along the edges of the faces, but instead across the interior of faces, this distance is an approximation of the geodesic distance. In practice the function is scaled to have values in the range $[0, 1]$.

DELTAS: The Dirac delta function $\delta : \mathbb{R} \rightarrow \mathbb{R}$ is defined by

$$\delta(x) = \begin{cases} 0, & x \neq 0 \\ \infty, & x = 0 \end{cases}$$

and can be constructed via the limit

$$\delta(x) = \lim_{a \rightarrow \infty} \frac{1}{|a|\sqrt{\pi}} \exp\left(-\left(\frac{x}{a}\right)^2\right).$$

To construct a highly peaked point-based function related to the Dirac delta function let $f : N \rightarrow \mathbb{R}$ be defined by

$$f(v) = \begin{cases} \frac{1}{|r|\sqrt{\pi}} \exp\left(-\left(\frac{d(p, v)}{r}\right)^2\right), & d(p, v) \leq r \\ 0, & d(p, v) > r \end{cases}$$

for a fixed r , where $d(p, v)$ is the approximate geodesic distance function as described above. The function is equal to zero on any point v which lies outside of a geodesic radius of r from the point p . Here r is set to 0.01.

NORMDIST: Similar to the above function based on the Dirac delta function, a function based on the normal distribution can be constructed. Define $f : N \rightarrow \mathbb{R} \cup \{\infty\}$ by

$$f(v) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma^2} \exp\left(-\left(\frac{d(p, v)}{2\sigma}\right)^2\right), & d(p, v) \leq r \\ 0, & d(p, v) > r \end{cases}$$

for a fixed r , where $d(p, v)$ is the approximate geodesic distance function as described above and the standard deviation σ is defined to be $\frac{1}{3} \times$ the maximum approximate geodesic distance between p and any v . Here r is set to be 0.1 and the functions are scaled so that $f(p) = 1$.

It is clear from the construction of the functions that the GEODISTS functions are globally supported whereas the DELTA and NORMDIST functions are locally supported around p . Figure 4.2 shows how the values of the above point-based functions differ. The graph plots the approximate geodesic distance $d(x, v)$ against the value of $f(v)$. Note that the natural log of the distance is taken to better display how the functions differ for vertices close to x .

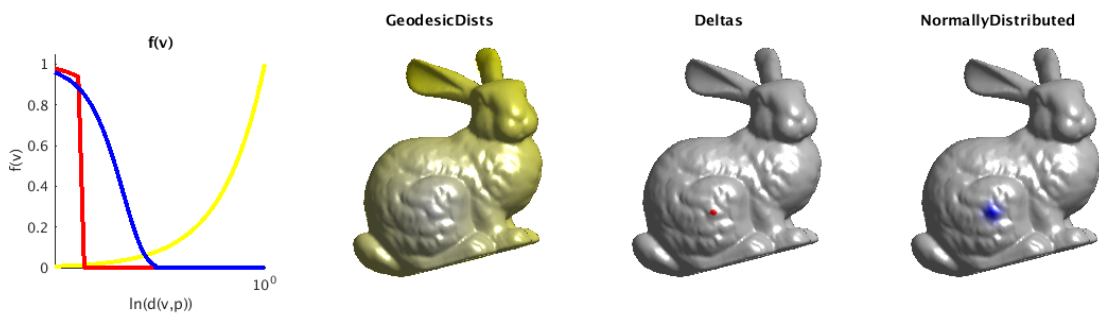


Figure 4.2: Functions based around a point on the bunny mesh.

An alternative to point-based functions are so called *multi-scale* functions which appear in geometry processing as a method of feature detection [96],[97],[74]. Here these are functions which depend on a time t . The following multi-scale functions are used:

HKS: The heat kernel signature (HKS) function $\text{HKS} : N \times \mathbb{R}^+ \rightarrow \mathbb{R}$ is defined by

$$\text{HKS}(v, t) = \sum_{i=1}^n e^{-\lambda_i t} \phi_i(v)^2, \quad [74]$$

where (ϕ_i, λ_i) are eigenvector/eigenvalue pairs of a discrete Laplacian. In practice, a set of 30 HKS functions are calculated via the first 300 eigenpairs of the FEM Laplacian for the mesh. Times t are taken as a logarithmically spaced sample of the interval $\left[\log_{10} \left(\frac{4 \ln 10}{|\lambda_n|} \right), \log_{10} \left(\frac{4 \ln 10}{|\lambda_{10}|} \right) \right]$ [74, section 5].

WKS: The wave kernel signature (WKS) function $WKS : N \times \mathbb{R}^+ \rightarrow \mathbb{R}$ is defined by

$$WKS(x, t) = \left(\sum_k C_{kt} \right)^{-1} \sum_{k=1}^n \phi_k^2(x) C_{kt}, \quad [98]$$

where (λ_i, ϕ_i) eigenpairs of a discrete Laplacian for the mesh, $C_{kt} = \exp\left(\frac{-(t - \ln \lambda_k)^2}{2\sigma^2}\right)$, $\sigma = \frac{\alpha(\ln \lambda_2 - \ln \lambda_n)}{m + 4\alpha}$ and $\alpha = 7$ [98, section 2.3]. In practice, a set of 100 WKS functions are calculated via the first 300 eigenpairs of the FEM Laplacian for the mesh. Define $s := \frac{7(\ln \lambda_2 - \ln \lambda_n)}{\text{number of time samples}}$, then times t are taken as a linearly spaced sample of the interval $[\ln \lambda_2 + 2s, \ln \lambda_n - 2s]$.

Figure 4.3 shows the heat and wave kernel signatures for increasing t , from left to right.

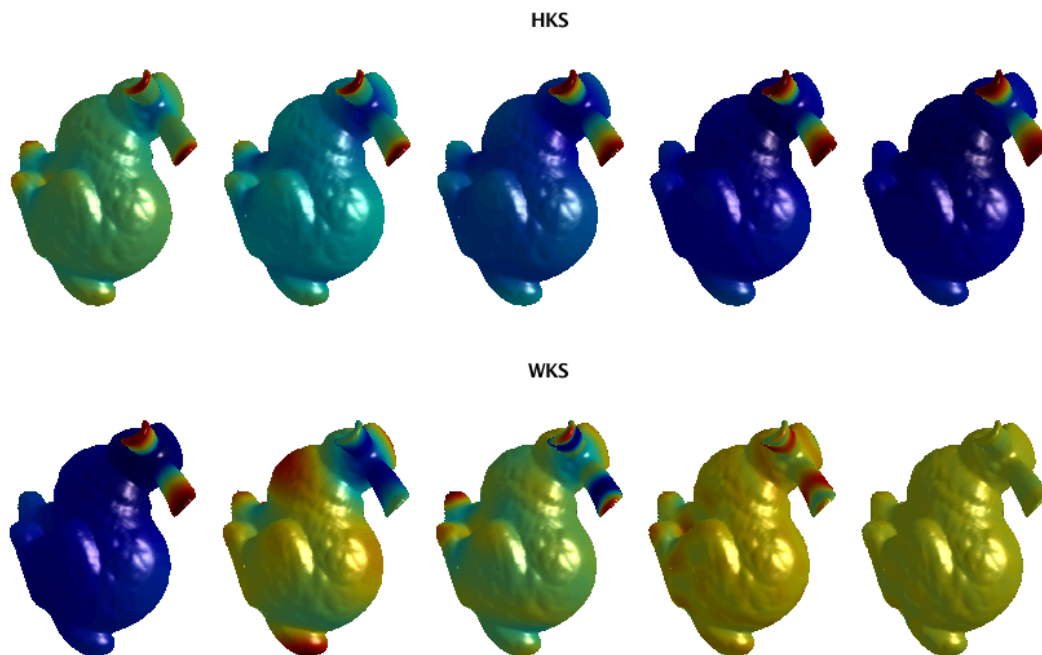


Figure 4.3: Multi-scale functions on the bunny mesh.

Additionally, the following functions are also used:

CONSTANT: The constant function on the mesh, $f(x) = 1, \forall x \in N$.

SEG: A set of segment indicator functions. Meshes are segmented via persistence-based segmentation [99]. Figure 4.4 shows the segment indicator functions for the bunny mesh. Note that there are no segmentations for the fish or victoria meshes.

VERTS: The vertex position functions. That is, the three vectors of coordinates defining the positions of vertices in \mathbb{R}^3 . Recall that these are the functions used in [6] to construct localised manifold harmonics.

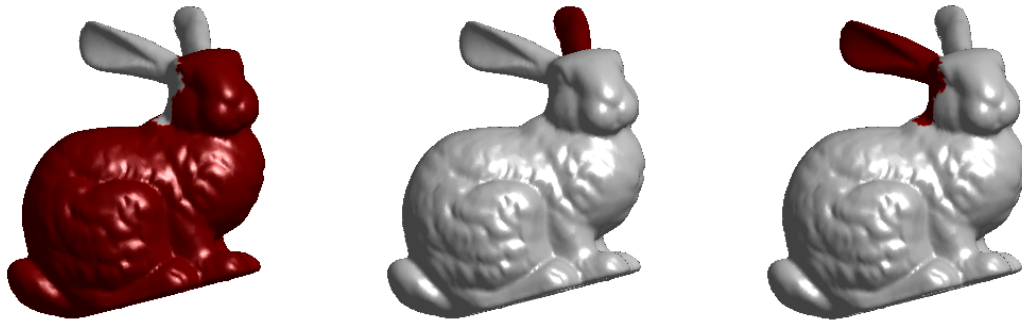


Figure 4.4: Segment indicator functions on the bunny mesh.

4.2 Function Reconstruction

Let Φ be a matrix representing a complete set of A -orthogonal basis functions for $\mathcal{F}(N, \mathbb{R})$ and let Φ_k be the $n \times k$ matrix with the first k basis functions as columns. Let a set of f functions represented by vectors in \mathbb{R}^n be arranged as columns in an $n \times f$ matrix \mathbf{F} . The matrix \mathbf{F} can be reconstructed in basis Φ_k such that $\mathbf{F} \approx \Phi_k M$ where M is a $k \times f$ matrix of coefficients. Let \mathbf{R} denote the reconstructed vector, with $\mathbf{R} = \Phi_k M$. Two methods of solving for the matrix M are considered.

4.2.1 The least squares solution

Using least squares minimisation, M can be found as the solution to

$$\arg \min_{M \in \mathbb{R}^{k \times f}} \|\mathbf{F} - \Phi_k M\|_F. \quad (4.2.1)$$

That is, $M = (\Phi_k^T \Phi_k)^{-1} \Phi_k^T \mathbf{F}$, via theorem 2.4.1. (The matrix Φ_k is full rank as its columns are linearly independent basis functions.) So, \mathbf{R} is given by

$$\mathbf{R} = \Phi_k (\Phi_k^T \Phi_k)^{-1} \Phi_k^T \mathbf{F}. \quad (4.2.2)$$

This is the method that MATLAB's backslash operator employs [100].

4.2.2 Reconstruction via matrix manipulation

Assume that $\mathbf{F} = \Phi_k M$, then since $\Phi_k^T A \Phi_k = I$ multiplying both sides on the left by $\Phi_k^T A$ gives

$$\Phi_k^T A \mathbf{F} = M. \quad (4.2.3)$$

Therefore, \mathbf{R} is given by

$$\mathbf{R} = \Phi_k \Phi_k^T A \mathbf{F}. \quad (4.2.4)$$

Note that this is the solution to

$$\arg \min_{M \in \mathbb{R}^{k \times f}} \|\mathbf{F} - \Phi_k M\|_A.$$

This method was inspired by function reconstruction in [9, section 4]. The claim is that a function $\mathbf{f} \in \mathbb{R}^n$ has reconstruction \mathbf{r} such that

$$\mathbf{r} = \Phi_k A \Phi_k^T \mathbf{f}, \quad (4.2.5)$$

where Φ_k is an $n \times k$ A -orthogonal matrix formed by columns of basis functions and A is an $n \times n$ area matrix. Clearly this is incorrect as the matrix dimensions are not compatible. When Φ_k is full ($n = k$) then the above reasoning can be used to show that $\mathbf{f} = \Phi_k \Phi_k^T A \mathbf{f}$. For both equation 4.2.4 and equation 4.2.5 to hold it must be that $\Phi_k^T A \mathbf{F} = A \Phi_k \mathbf{f}$. In general this is only true when $n = k$ and $A = \alpha I$ for some $\alpha \in \mathbb{R}$. Note that it is also claimed that reconstruction in this way can be achieved 'quickly'. The experiments below verify that this is false, when comparing this method against least squares via backslash in MATLAB 2018a. Of course, MATLAB is optimised for use of backslash, and this may not hold true when using alternative versions.

4.2.3 The $A = I$ Case

When the area matrix A is equal to the identity matrix (for example in the graph Laplacian case) the two minimisation problems above are equivalent, and so the reconstructions are equal. To see this consider Φ such that $\Phi^T A \Phi = I$, with $A = I$. Then

$$\begin{aligned} (\Phi^T \Phi)^{-1} \Phi^T \mathbf{F} &= \Phi^T \mathbf{F}, \text{ since } \Phi^T \Phi = I, \\ &= \Phi^T A \mathbf{F}, \text{ since } A = I. \end{aligned}$$

4.2.4 Measuring Reconstruction Error

Reconstruction error for a single function is computed as the Euclidean distance between the original function and its reconstruction. To measure the reconstruction error between a set of f functions \mathbf{F} and reconstruction \mathbf{R} , consider the distance of the reconstruction from the original function set *per vertex*, then find an average value by taking the mean over the vertex set. This construction of an error value is formalised in the following definitions:

Definition 4.2.1: Let \mathbf{F} be a matrix representing f functions on n vertices, with functions as columns. Let \mathbf{R} be the corresponding reconstruction of \mathbf{F} in a given basis. Note that the value of the function j -th function in \mathbf{F} evaluated at vertex v_i is represented by the matrix entry \mathbf{F}_{ij} . Then, the **reconstruction error function** Err_f is defined by

$$Err_f(\mathbf{F}, \mathbf{R}, v_i) = \sqrt{\sum_{j=1}^f |\mathbf{F}_{ij} - R_{ij}|^2}. \quad (4.2.6)$$

As with other real-valued functions define for the vertices of a mesh this can be written as an n -dimensional vector.

Definition 4.2.2: The **reconstruction error** is obtained as the mean of the reconstruction error function, i.e.

$$Err(\mathbf{F}, \mathbf{R}) = \frac{1}{n} \sum_{i=1}^n Err_f(\mathbf{F}, \mathbf{R}, v_i). \quad (4.2.7)$$

4.3 Basis Choice and Calculation

The space of real-valued functions on a mesh $\mathcal{F}(M, \mathbb{R})$ can be approximated via a set of basis vectors. The classical choice is a truncated set of Laplacian eigenfunctions. Here several additional basis constructions are also considered. The sets of basis functions are as follows:

LBO: The first 150 eigenfunctions of the FEM Laplacian, calculated via MATLAB's `eigs` function.

GL: The first 150 eigenfunctions of the graph Laplacian, calculated via MATLAB's `eigs` function.

GLMH + (*function type*): Bases constructed from a set of the first 100 Laplacian eigenfunctions, then 50 localised manifold harmonics. Localised manifold harmonics were calculated by considering the reconstruction of the function types listed in section 4.1, with $\mu_R = \mu_{\perp} = 10^6$. The Laplacian eigenfunctions used are the same as the Laplacian eigenfunctions LBO above. These are all variants of LMHs, using the different function types in the construction of the region indicator, however the label GLMH is used, as in definition 3.3.9. Note that Q and M are as in equation (3.1.5). The GLMH functions were calculated using a combination of MATLAB's `eigs` function and the Woodbury matrix identity to reduce computation complexity (see theorem 6.1.17/appendix G).

CMM: Bases of compressed manifold modes calculated via the sequential method described in 3.2, with $\mu = 50$.

CMMs were calculated twice (allowing 10K/100K iterations), with a cap on the calculation time after 150 minutes. The calculation times for basis/mesh pairings are shown in figure 4.5. For the GLMH functions timings do not include the time taken to calculate the initial set of Laplacian eigenfunctions, but do include the time taken to locate the regions of localisation. Note that the y-axis is log-scaled, that markers are slightly offset to allow for ease of reading and that CMM bases may consist of fewer than 150 basis functions as the full set may have failed to

be calculated in the two-and-a-half-hour calculation period. (See appendix D for further details on CMM calculation.)

As expected, the bases calculated using `eigs` are calculated faster than the CMMs which are calculated by an iterative algorithm. The GLMH functions take longer to calculate than the LBO of GL eigenfunctions, even though there are fewer of them – 50 GLMH eigenfunctions in comparison to 150 LBO/GL eigenfunctions. This is due to the time spent locating the region of localisation and the additional matrix multiplication involved in the eigenproblem. The similarity between the subspaces spanned by the different sets of basis functions is discussed in section 4.6.

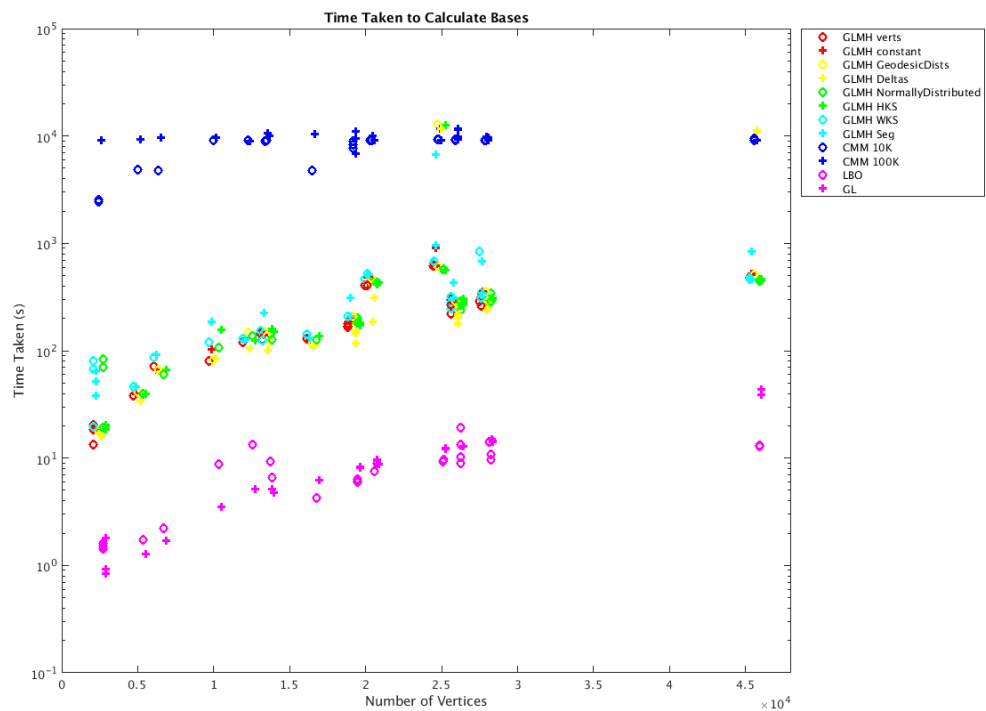


Figure 4.5: Time taken to calculate various basis types

4.3.1 GLMH Orthogonality Failure

Recall from the construction of localised manifold harmonics that the aim is to construct new functions which are both A -orthogonal to each other and to a set of existing functions, where A is an area matrix. Let Φ denote the matrix of existing functions (Laplacian eigenfunctions) and let Ψ denote the matrix of GLMH functions. That the matrix Ψ is such that $\Psi^T A \Psi = I$ is a result of the matrix Ψ being a matrix of A -orthogonal eigenfunctions. The condition that $\Phi^T A \Psi = 0$ is enforced in the formulation of the eigenproblem, via the priority parameter μ_{\perp} . To evaluate the failure to meet the orthogonality condition consider the norm $\|\Phi^T A \Psi\|_F$. Figure 4.6 plots this error for each mesh/GLMH-type basis pair. To aid reading the error is capped at 0.2. This means that the GLMH Delta errors for the head1, head2 and victoria21 meshes have been omitted. The full figure can be found in appendix E. As before markers are slightly offset for each basis type.

Figure 4.6 shows that the error is consistently worst for the GLMHs calculated via the delta functions. GLMHs calculated via the WKS functions perform inconsistently; there is not even a trend for meshes from the same collection – large error for horse0 and horse6 but small error for horse10.

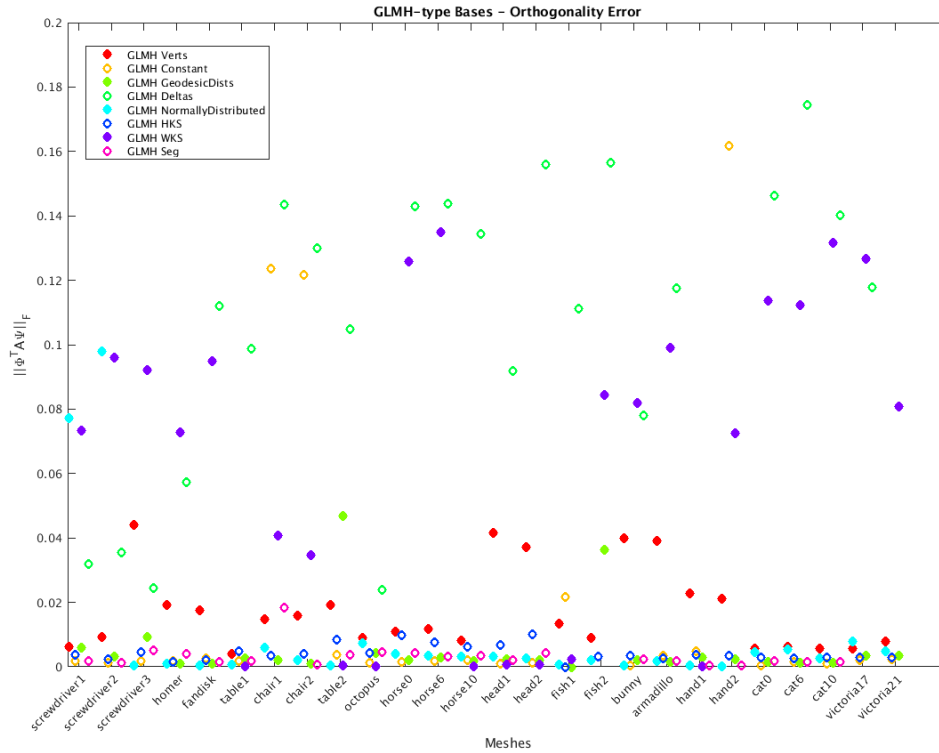


Figure 4.6: The failure to meet the $\Phi^T A \Psi = 0$ orthogonality condition.

4.4 Mesh Choice and Quality

The meshes used have less than 50 000 vertices. (Smallest, screwdriver meshes with 2 502 vertices; largest, victoria meshes with 45 659 vertices.) This is not a particularly large number of vertices, but the variation in size of vertex set is chosen to allow comparison between meshes of various sizes.

The area of faces in a mesh can cause problems in numerical calculations, e.g. when an area is very small there can be errors when using the inverse area matrix. Figure 4.7 shows a set of boxplots detailing the sizes of faces in the meshes used in experiments. The areas used are taken from the (lumped) FEM Laplacian calculated for each mesh, since this is the area matrix used in subsequent numerical experiments. In all calculations meshes are scaled to have a surface area equal to 1.

Therefore, to view the face area in a consistent way, the area is multiplied by the number of vertices in the mesh. That is, if the mesh has faces of a uniform size, the faces will have a scaled area equal to 1 in the boxplot. The meshes with a small range are more uniform. The cyan line provides a plot of area = 1 for comparison.

Note that the fandisk, table, chair and hand meshes are closest to being uniform. The cat and victoria meshes have greatest variation and so are least uniform. Figure 4.8 shows the variation in area by assigning a colour to each vertex. Orange areas indicate vertices with scaled area close to 1, red areas indicate vertices with small associated area and blue areas indicate vertices with large associated area. The zoomed-in snapshots show the variation in mesh faces. Appendix E contains the box plot figure for all meshes featured in the thesis.

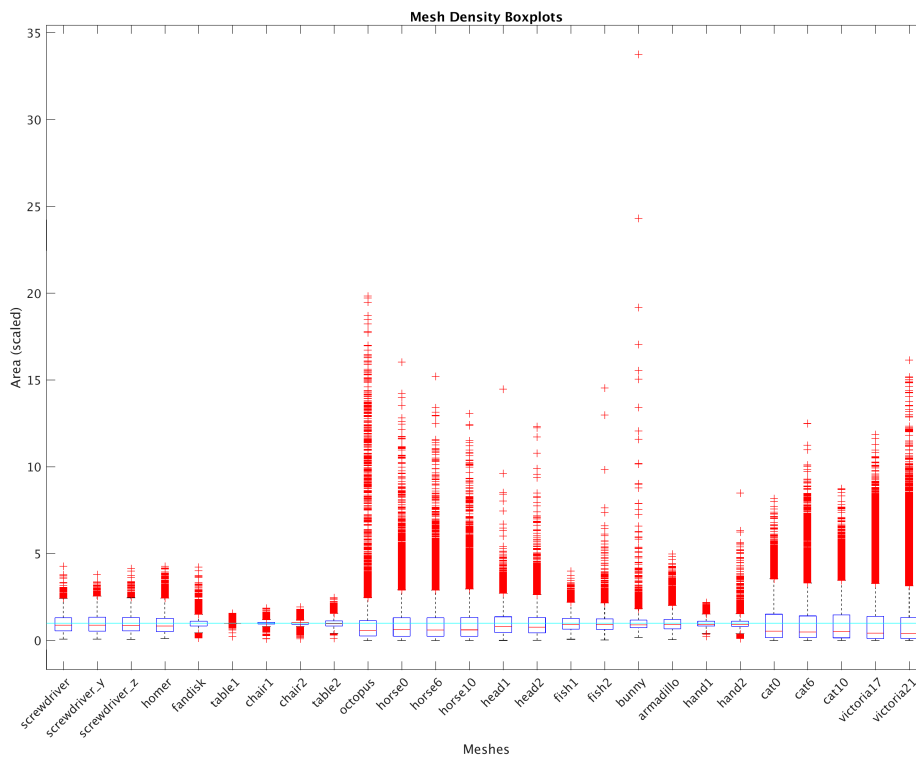


Figure 4.7: Mesh area comparison boxplots

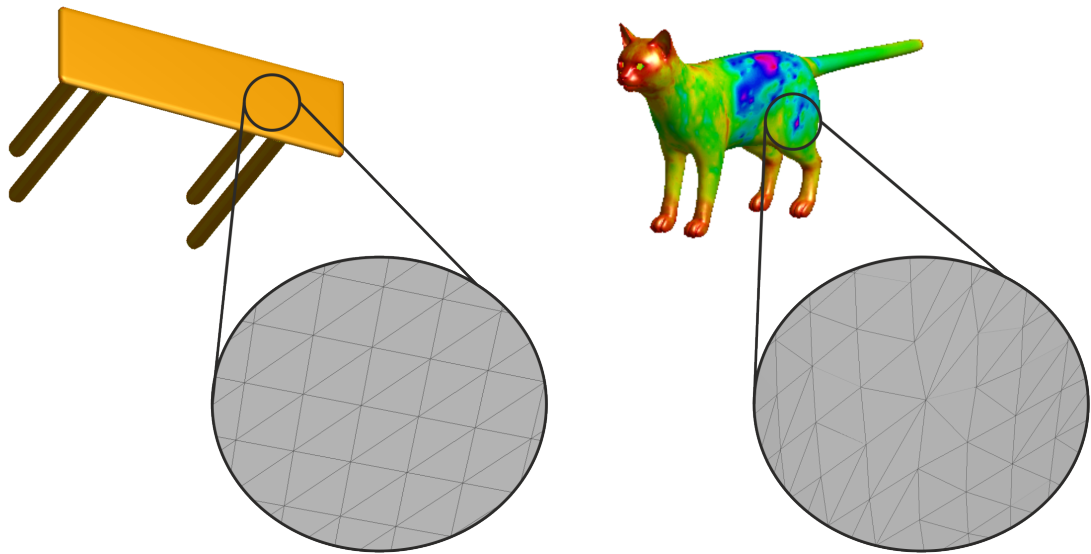


Figure 4.8: Mesh area uniformity – close ups on table1 and cat0 meshes

4.5 Function Reconstruction Experiments

Function sets were reconstructed for the mesh set via a variety of basis types. There were a total of 96 function/basis combinations (8 function types, 12 different bases). Functions were reconstructed both via backslash/least squares and via equation (4.2.4). Figure 4.9 shows a typical comparison of methods by basis type (in this case, the basis is the classical Laplacian eigenbasis). Reconstruction errors are plotted per mesh, for each function set. Solid markers denote least squares reconstruction and unfilled markers denote reconstruction via matrix manipulation. The unfilled markers are slightly offset to allow for ease of reading. The least squares reconstructions tend to have smaller error (i.e. solid markers appear lower than unfilled markers of the corresponding colour).

As expected the graph Laplacian basis reconstructs with the same error for both reconstruction methods (as they are equivalent, see section 4.2.3). This can be seen in figure 4.10 as the solid and unfilled markers appear side-by-side.

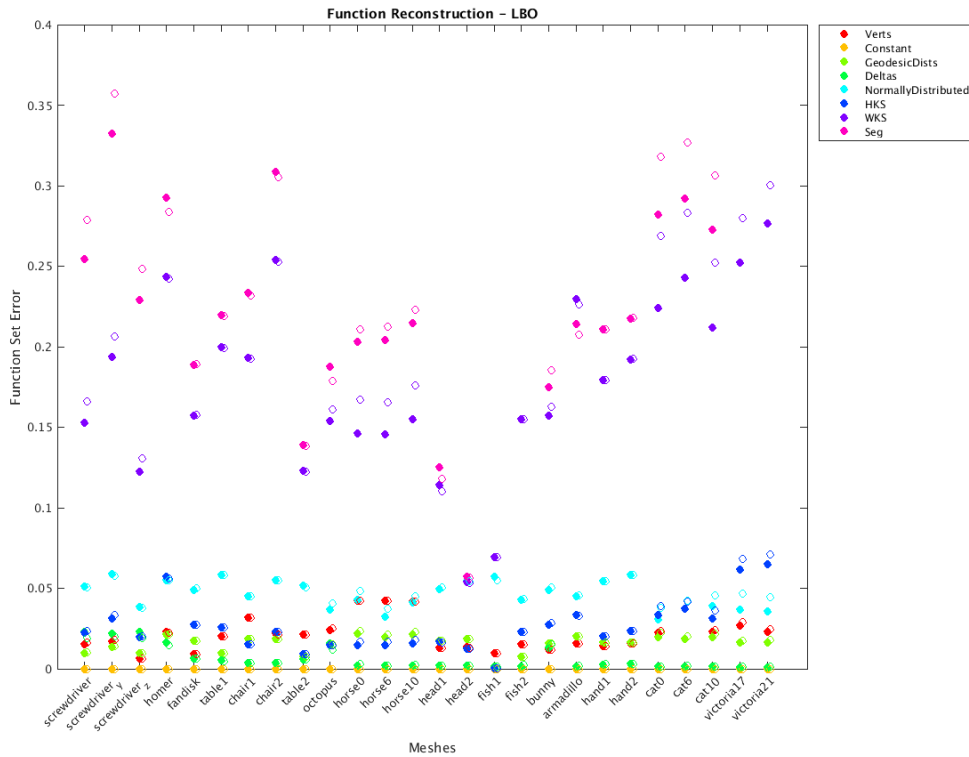


Figure 4.9: A typical reconstruction error figure.

To compare the reconstruction methods sets of functions for each mesh were reconstructed in each basis for an increasing number of basis vectors. The time taken to calculate the reconstruction and the function set reconstruction error were stored. The results were normalised by the number of vertices in the mesh, and then collated. Only the meshes with fewer than 24 850 vertices were used in these reconstructions due to the time taken to compute multiple reconstructions for larger meshes. Figure 4.11 shows two graphs based on this data. The left-hand graph plots basis size against time taken, comparing the smallest and largest values for each reconstruction method. Using backslash is faster: recall that this may be a result of the optimisation of MATLAB for solving least squares problems via backslash. The right-hand figure confirms that reconstruction via least squares minimisation (backslash) gives a smaller error than when using matrix manipulation reconstruction. The large jump in reconstruction accuracy in the

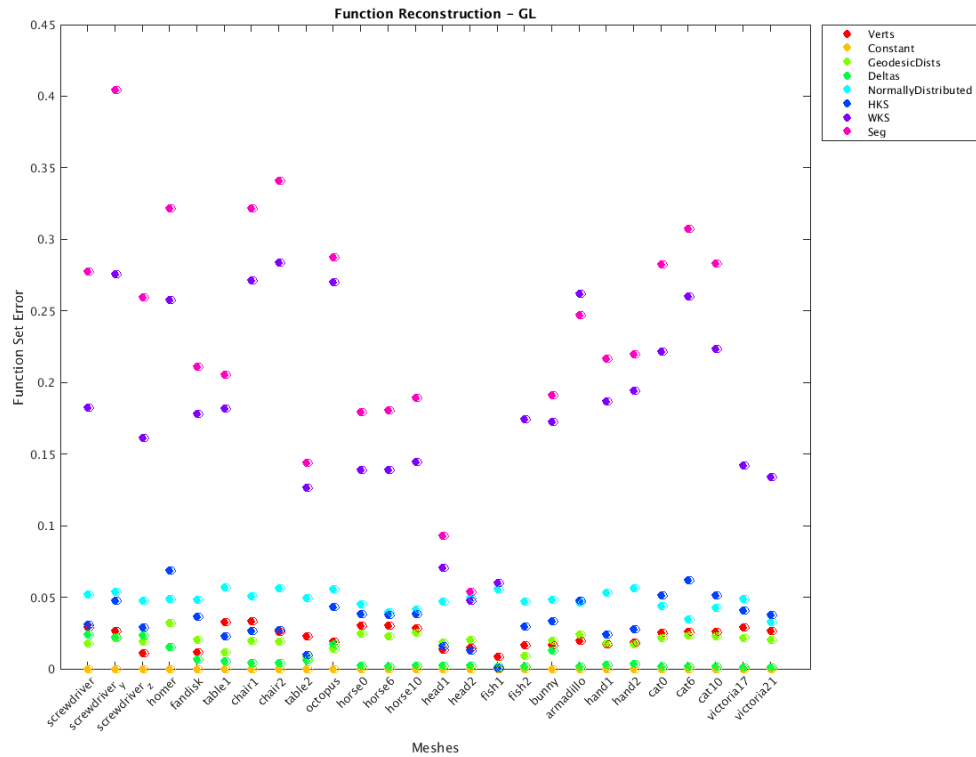


Figure 4.10: Reconstruction error when using the graph Laplacian.

matrix manipulation reconstruction at the basis size equal to 100 mark is due to the large number of functions reconstructed in a GLMH basis. Recall that the matrix manipulation reconstruction uses the assumption that the matrix $\Phi^T A \Phi = I$ where Φ is the matrix of basis functions and A is the Laplacian area matrix. Recall also that the GLMH functions are such that the first 100 basis functions are Laplacian eigenfunctions (A -orthogonal), and the following 50 basis functions are A -orthogonal to each other, however the two subsets of basis functions are not exactly orthogonal (recall figures 3.3 and 4.6). Therefore the matrix manipulation reconstruction is adversely affected by these basis types. From this point reconstructions of functions are calculated using backslash.

The trends shown in the reconstruction error for each function type are consistent across each basis type. This can be seen by considering the full set of reconstruction error figures produced in the same way as figures 4.9 and 4.10 and is shown in

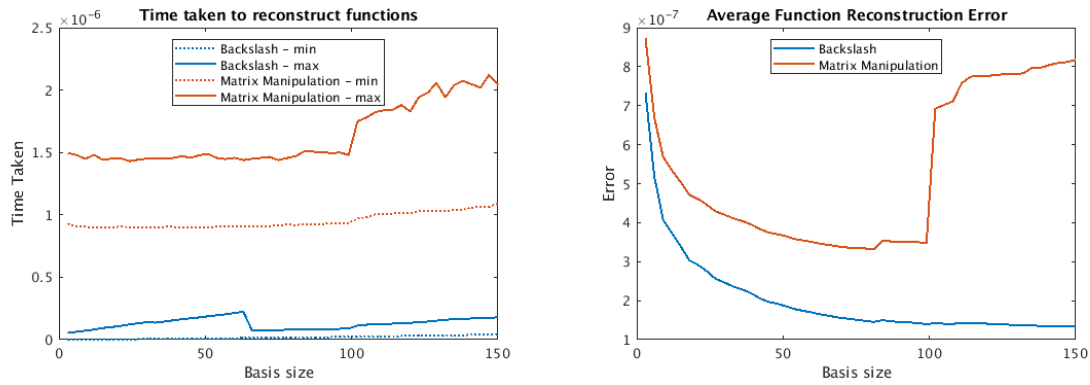


Figure 4.11: Function reconstruction details, taken as a mean of values from all tested bases.

figure 4.12, where the function reconstruction error has been scaled to be between 0 and 1 for each basis type. To see this in an alternative way, figure 4.13 displays a matrix of function set reconstruction errors for each mesh, with reconstruction calculated via backslash. The elements of the matrix are represented by a colour, using MATLAB's colormap `hot`, which gives small values a pale colour, and large values a dark colour. The rows of the matrices are indexed by basis choice, and the columns are indexed by function type (for precise details see figure 4.14). Note that since the fish and victoria meshes do not have a segmentation the GLMH_SEG bases do not exist, hence the corresponding cells in the matrix are marked by strikethrough. Colours form noticeable vertical stripes, indicating that function type has greater impact on quality of a reconstruction than basis type. Constant functions are approximated very well – this is expected for the Laplacian-based bases as the first eigenfunction is the constant function (in practice very close to the constant function). The highly localised delta functions are also approximated well, which is more surprising.

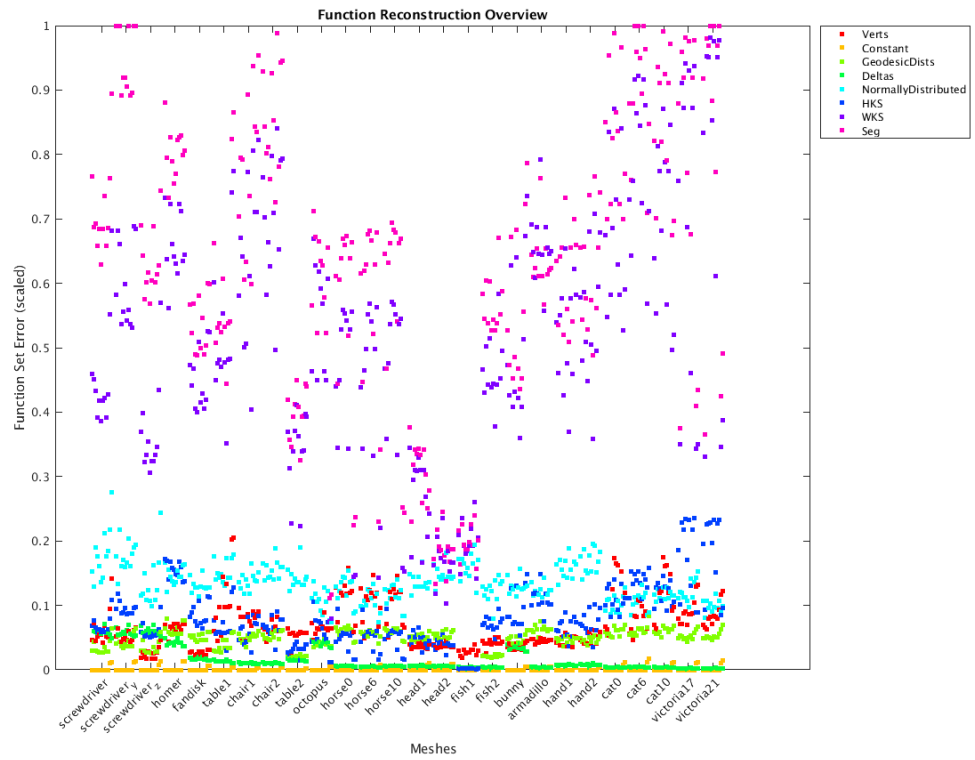


Figure 4.12: Function reconstruction error (via backlash) across all basis types

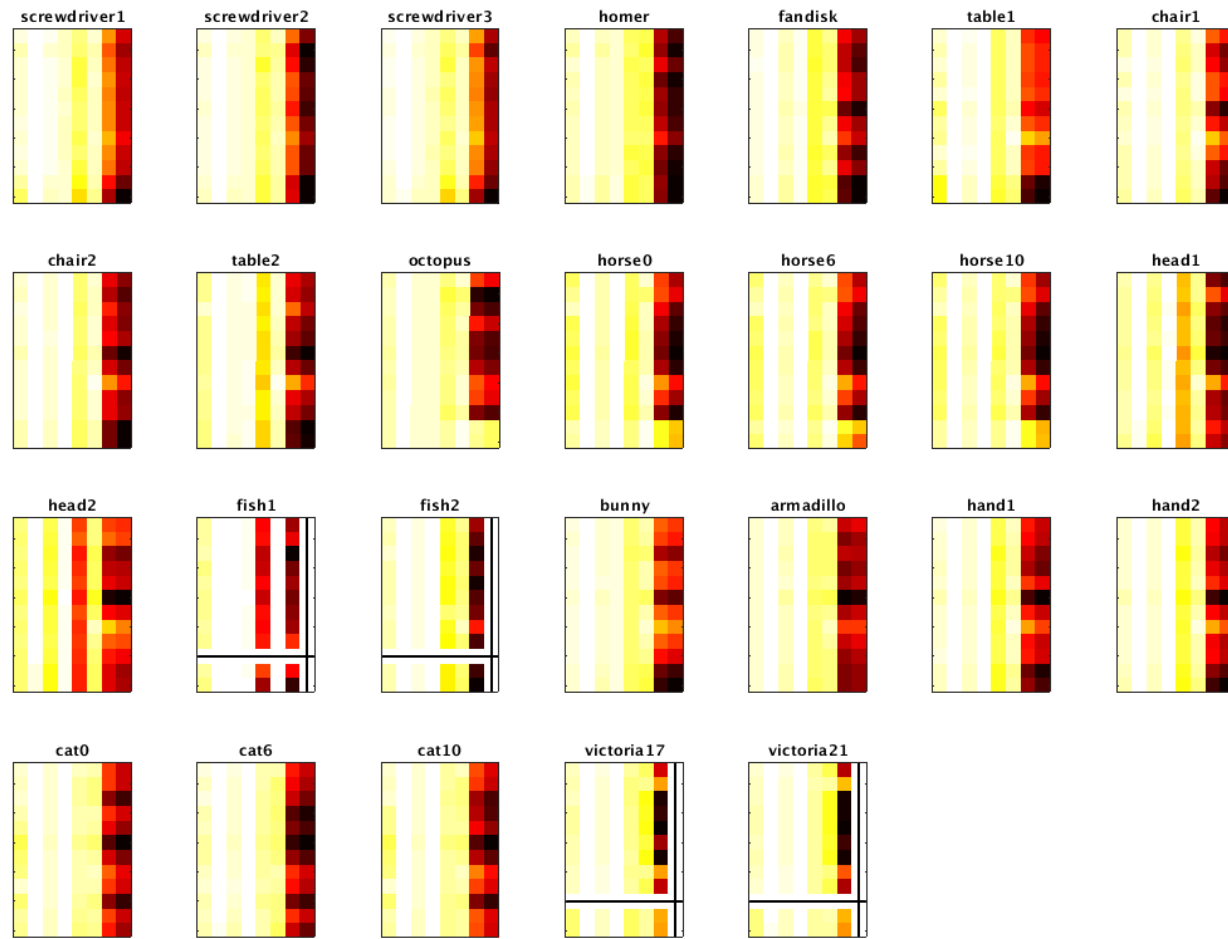


Figure 4.13: Matrices of function set reconstruction errors.

Figure 4.14 shows two matrices of errors where values have been averaged across the mesh set. The left-hand matrix shows the unscaled data, which shows that the WKS and SEG functions are reconstructed with greatest error, as was evident from figure 4.12. The right-hand matrix has been scaled so that the columns sum to 1. This emphasises rows which have consistently low values and should highlight variance between performance in basis type. The CMM bases do not reconstruct the constant function well. Of course, this was expected, but it is interesting to note that this is the only basis type which performs noticeably poorly for any of the function types.

It was expected that the GLMH basis types would reconstruct the functions used in their construction far better than any of the other basis types. To examine this, figure 4.15 shows the right-hand matrix from figure 4.14 with the constant function removed. A significantly lower error would be indicated by a pale cell in the entry indexed by, for example, GLMH VERTS and VERTS. The only basis types where this occurs are the GLMH bases calculated via the vertex position and HKS functions. A row of evenly coloured cells indicates a basis type which performs equally well for all function types – the best example being the GLMHs calculated using the constant function.

A final visualisation of the quality of function reconstructions by basis type is given in figure 4.16. It shows a set of boxplots, plotting function reconstruction error values for each basis type. The lower the mean, the more accurate the reconstructions. The smaller the interquartile range, the more capable the basis type of reliably reconstructing functions as the reconstruction error is more consistent across function types. The boxplots corroborate the assertion that choice of function type is more important than choice of basis type in function reconstruction since there are no bases which perform significantly better or worse than any other overall.

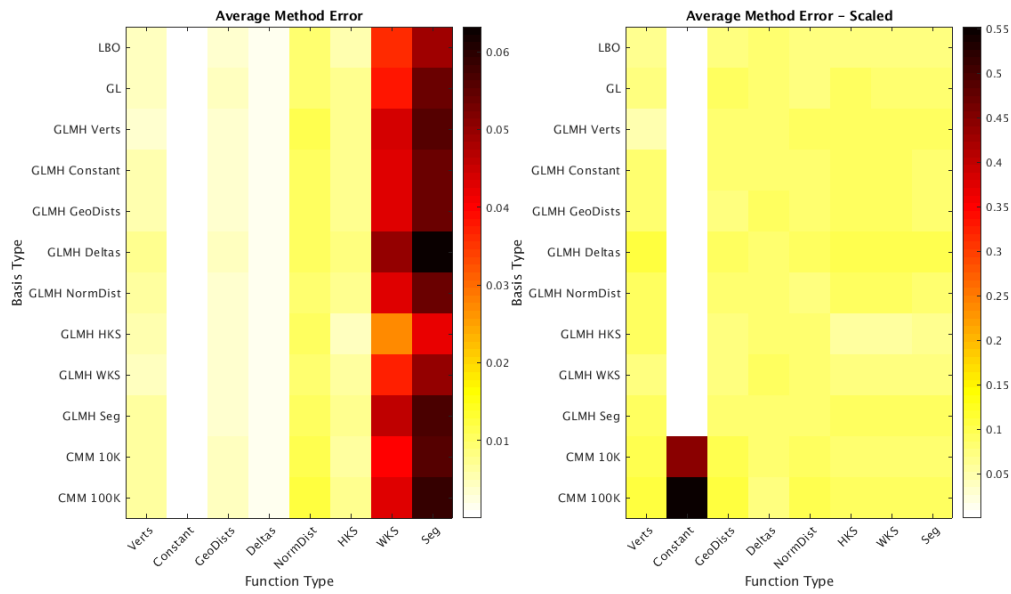


Figure 4.14: Average method error; unscaled (L) and scaled (R)

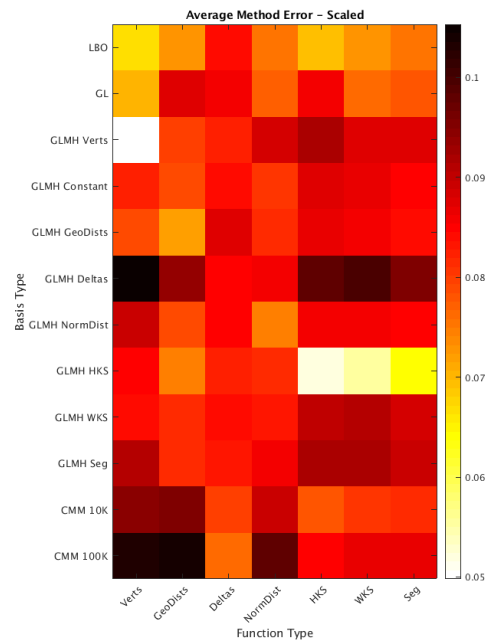


Figure 4.15: Average method error; scaled with constant function removed

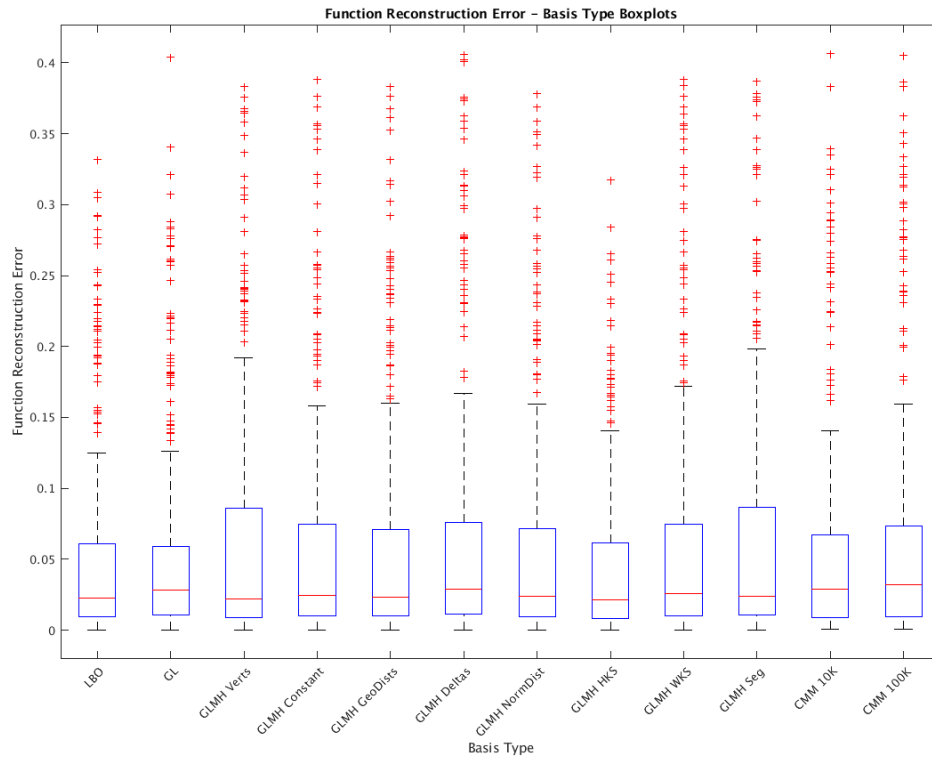


Figure 4.16: Function reconstruction error boxplots, by basis type

4.6 Basis Reconstruction

Each set of k basis functions spans a k -dimensional linear subspace of $\mathcal{F}(N, \mathbb{R}) = \mathbb{R}^n$. Of course, these subspaces are not necessarily equivalent. To give a numerical value to the difference between the subspaces a distance based on the Grassmanian distance, altered for subspaces of different sizes, described in [101] is used.

Definition 4.6.1: Let $\mathcal{U} \subseteq \mathbb{R}^n$ be a linear subspace of dimension k , with orthonormal basis $\{\phi_1, \dots, \phi_k\}$ represented as columns of matrix Φ . Let $\mathcal{V} \subseteq \mathbb{R}^n$ be a linear subspace of dimension l , with orthonormal basis $\{\psi_1, \dots, \psi_l\}$ represented as columns of matrix Ψ . Let $m = \min\{k, l\}$. The **principal angles between \mathcal{U} and \mathcal{V}** , denoted by $\theta_1, \dots, \theta_m$, are defined by

$$\theta_i = \cos^{-1} \sigma_i,$$

where σ_i is the i -th singular value of the singular value decomposition

$$U\Sigma V^T = \Phi^T\Psi.$$

Proposition 4.6.2: The principal angles between linear subspaces \mathcal{U} and \mathcal{V} are independent of the choice of bases for \mathcal{U} and \mathcal{V} .

Proof. Let $\{\phi_1, \dots, \phi_k\}$ and $\{x_1, \dots, x_k\}$ be orthonormal bases for \mathcal{U} , represented as columns of the matrices Φ and X respectively. Then $X = \Phi M$ where M is a $k \times k$ matrix of basis coefficients. Since the columns of Φ are orthonormal, $\Phi^T\Phi = I$ and

$$X^T X = M^T \Phi^T \Phi M = M^T M.$$

But, the columns of X are also orthonormal, with $X^T X = I$, and hence $M^T M = I$. That is, M is an orthonormal matrix.

Similarly, let $\{\psi_1, \dots, \psi_k\}$ and $\{y_1, \dots, y_k\}$ be orthonormal bases for \mathcal{V} , represented as columns of the matrices Ψ and Y respectively. Then $Y = \Psi N$ where N is a $k \times k$ matrix of basis coefficients. As above, N is orthonormal.

From definition 4.6.1 recall that the principal angles between \mathcal{U} and \mathcal{V} are defined via the singular value decomposition of the product $\Phi^T\Psi$. Let $U\Sigma V^T = \Phi^T\Psi$ be the singular value decomposition of $\Phi^T\Psi$. Now, consider the alternative bases:

$$\begin{aligned} X^T Y &= M^T \Phi^T \Psi^T N \\ &= M^T U \Sigma V^T N. \end{aligned}$$

Then since M and N are orthonormal,

$$X^T Y = \bar{U} \Sigma \bar{V}^T$$

where $\bar{U} = M^T U$ and $\bar{V} = N^T V$. Therefore, the singular values of $\Phi^T\Psi$ are equal to the singular values of $X^T Y$, and so the principal angles between \mathcal{U} and \mathcal{V} are independent of the choice of bases. \square

Lemma 4.6.3: Let A be an $n \times n$ symmetric positive definite matrix and let \mathcal{U} and \mathcal{V} be linear subspaces of \mathbb{R}^n with A -orthonormal bases represented by Φ and Ψ

respectively. Then the principal angles between \mathcal{U} and \mathcal{V} are given by the singular values of the singular value decomposition

$$U\Sigma V^T = \Phi^T A\Psi.$$

Proof. Let $L = \text{Chol}(A)$, then $L^T\Phi$ and $L^T\Psi$ are orthonormal matrices, representing orthonormal bases of \mathcal{U} and \mathcal{V} respectively. Applying the definition of principal angles gives the result,

$$\begin{aligned} U\Sigma V^T &= (L^T\Phi)^T(L^T\Psi) \\ &= \Phi^T L L^T \Psi \\ &= \Phi^T A \Psi, \end{aligned}$$

as $LL^T = A$. □

Definition 4.6.4: Let $\mathcal{U} \subseteq \mathbb{R}^n$ be a linear subspace of dimension k and let $\mathcal{V} \subseteq \mathbb{R}^n$ be a linear subspace of dimension l , The **Grassmanian distance between \mathcal{U} and \mathcal{V}** is defined to be

$$d_{Gr}(\mathcal{U}, \mathcal{V}) := \left(\sum_{i=1}^{\min\{k,l\}} \theta_i^2 \right)^{\frac{1}{2}} \quad [101],$$

where θ_i are the principal angles between \mathcal{U} and \mathcal{V} .

The Grassmanian distance between \mathcal{U} and \mathcal{V} is such that $d_{Gr}(\mathcal{U}, \mathcal{V}) = 0$ if and only if $\mathcal{U} \subseteq \mathcal{V}$ or $\mathcal{V} \subseteq \mathcal{U}$ [101, lemma 13]. It follows that it does not satisfy the triangle inequality, however, when the dimensions of \mathcal{U} and \mathcal{V} are equal, i.e. $l = k$, the Grassmanian distance between \mathcal{U} and \mathcal{V} reduces to the geodesic distance on the Grassmanian manifold $\text{Gr}(n, k)$.

Figure 4.17 shows a matrix of Grassmanian distances for each mesh, where rows and columns are indexed by basis type, listed as in figure 4.14. Again note that since the fish and victoria meshes do not have a segmentation the GLMH_SEG bases do not exist, the corresponding cells in the matrix are marked by strikethrough. Paler cells denote pairs of bases which span nearby subspaces of $\mathcal{F}(N, \mathbb{R})$.

It's interesting to note that the subspace spanned by the graph Laplacian eigenfunctions is furthest from all other subspaces. The variation between the subspaces spanned by the GLMH functions shows that for some meshes the different functions types used in their construction affect similar areas (see the fish1 figure, all the GLMH bases span nearby subspaces); for other meshes the function types are badly reconstructed in different areas, resulting in greater variation between the subspaces spanned by the resulting bases (see the chair or screwdriver figures). For the meshes head 1 to hand 2 the CMM bases span subspaces close to the subspace spanned by the LBO eigenfunctions, as indicated by the paler matrix elements. To investigate this further recall that the sets of basis functions can be reconstructed in the same way as any other set of functions defined on the mesh. Functions which lie in nearby subspaces should be reconstructed well by one another.

As the Grassmanian distance between subspaces spanned by Laplacian eigenfunctions and compressed manifold modes is surprisingly small for some meshes, the reconstructions of Laplacian eigenfunctions and CMMs are focused on. To do this 150 compressed manifold modes were calculated for each mesh via ADMM, with $\frac{\epsilon}{n} = 0.008$. Note that this is not the same as the CMM bases used previously in the chapter as those were calculated via a sequential method. Reconstructions were calculated for a matrix Φ representing one set of basis functions, so that $\Phi \approx \Psi M$. The coefficient matrix M was calculated via backslash and the number of basis functions in the matrix Ψ increased incrementally.

Figure 4.18 shows the results of these reconstructions. The x -axis plots the number of basis functions used in the reconstruction; the y -axis plots reconstruction error; red lines denote CMMs reconstructed in LBO; blue lines denote Laplacian eigenfunctions reconstructed in CMM. In each graph the line which appears lower is the reconstruction with smaller error. When blue lines are lower compressed manifold modes reconstruct the Laplacian eigenfunctions better than the Laplacian eigenfunctions reconstruct the compressed manifold modes; when red lines are lower Laplacian eigenfunctions reconstruct compressed manifold modes better than the compressed manifold modes reconstruct the Laplacian eigenfunctions. When both lines appear close together both bases reconstruct the other to a similar degree of

accuracy. Note that, as expected, the graphs where the reconstruction errors are similar are for the same meshes as those where the basis functions spanned nearby subspaces.

There are meshes where there is little difference between the performance of both bases (see the third row of figure 4.18). Note that these are the meshes with 20-26K vertices. As the choice of the sparsity parameter μ is difficult to make, due to the lack of any real concept of what makes a “good” choice, it is suggested that μ was “well-chosen” for these meshes. There are also notable plateaus in reconstruction error as the CMM basis size increases for some meshes (see the first row). This is due to a change in the sparsity of the modes as the basis size increases. To see this figure 4.19 plots the area-scaled ℓ_1 norm $\|A\Psi_k\|_1$ for matrix of basis functions Ψ_k as k increases. A significant change in gradient of the line indicates a significant change in the sparsity of the modes.

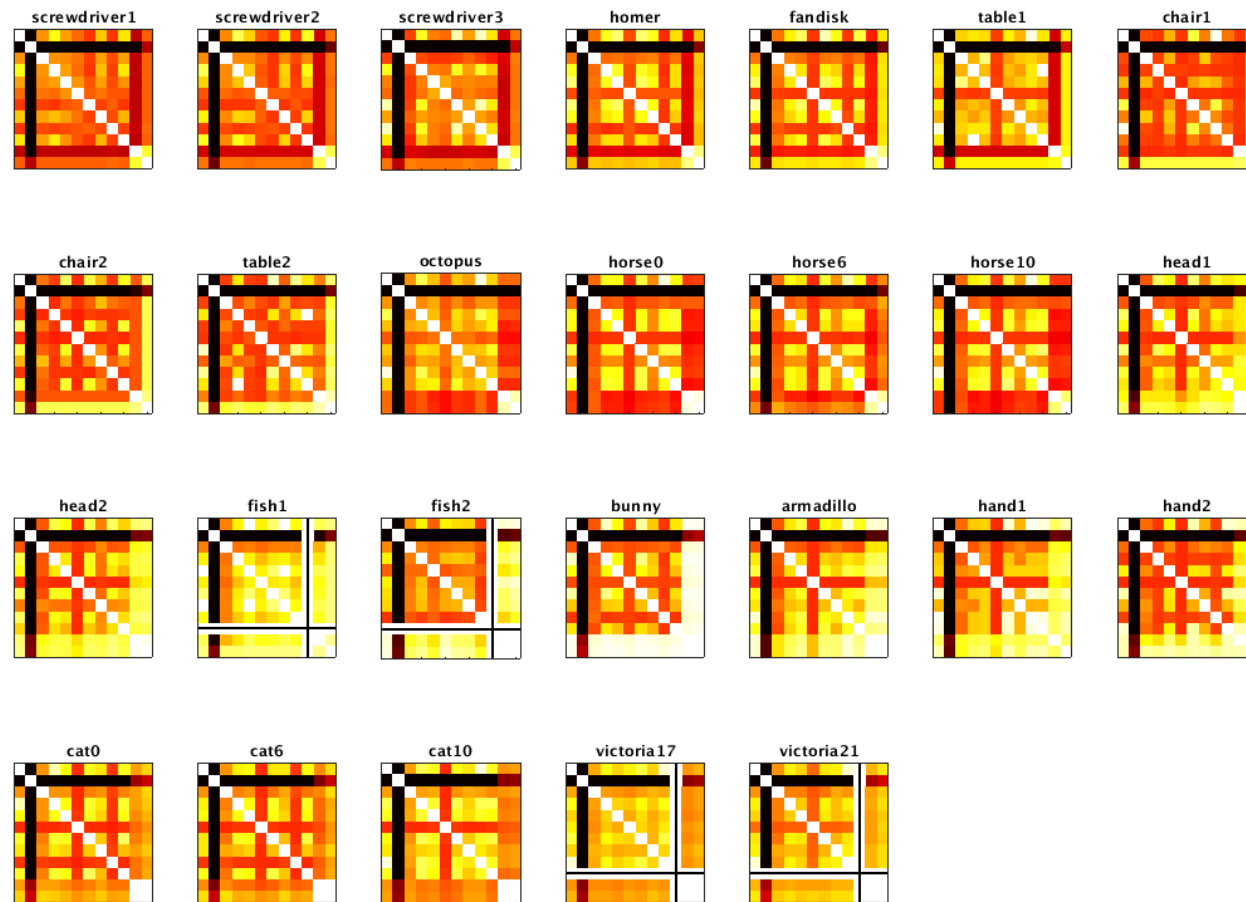


Figure 4.17: Matrices of Grassmanian distances between function space subspaces formed by different basis constructions.

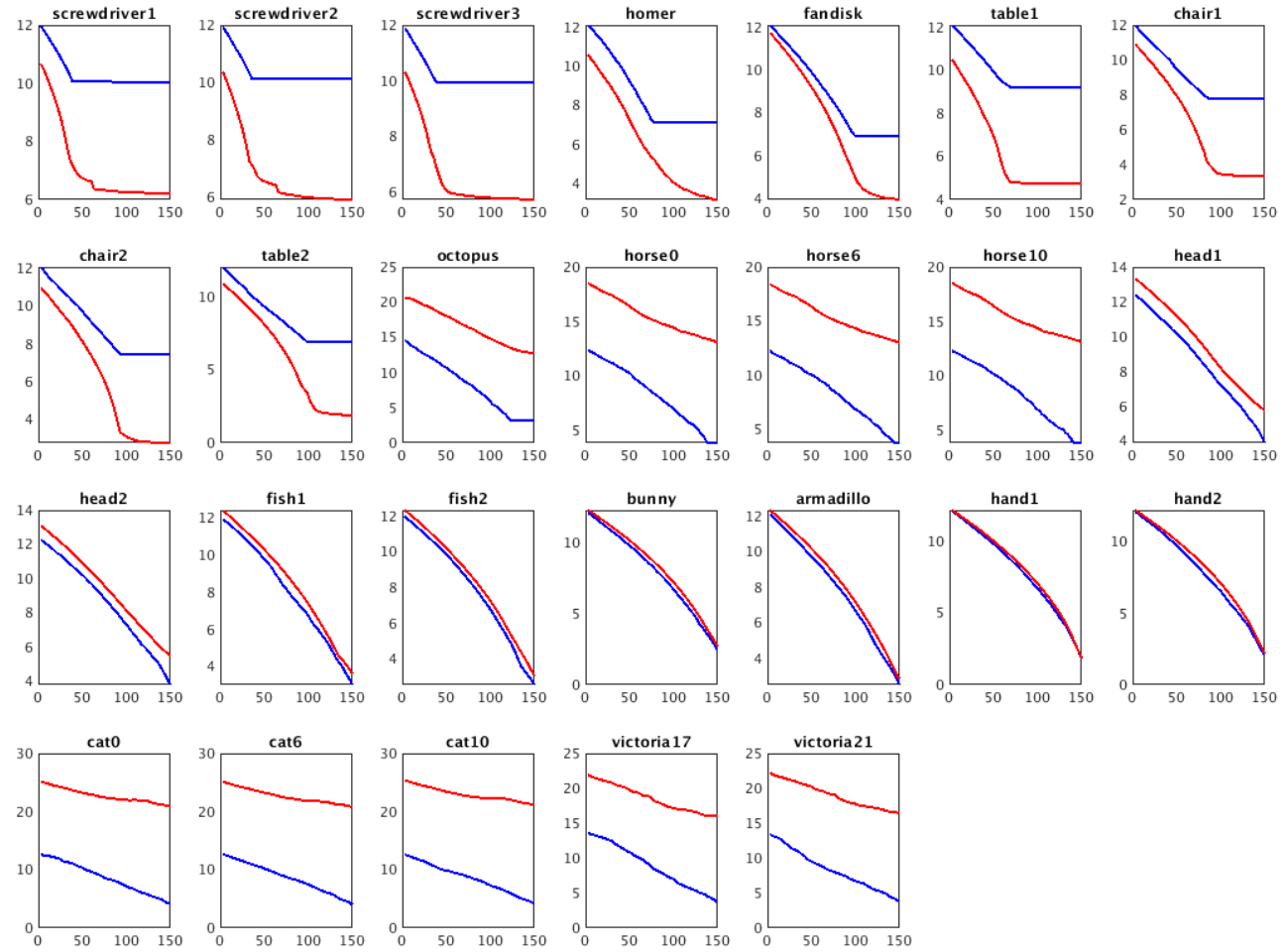


Figure 4.18: Basis function reconstruction. Red lines denote CMMs reconstructed in LBO; blue lines denote Laplacian eigenfunctions reconstructed in CMM.

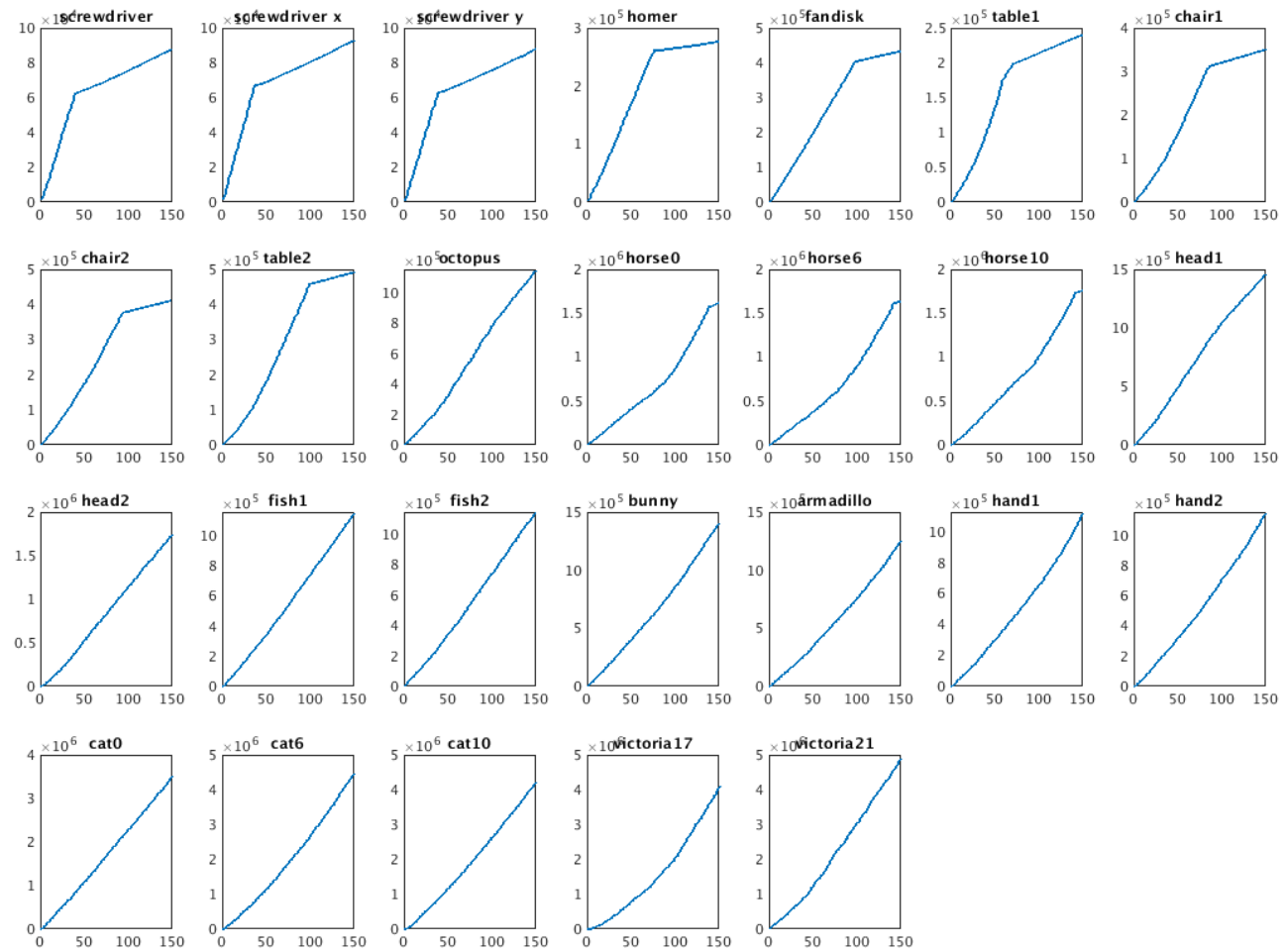


Figure 4.19: The cumulative sparsity of compressed manifold modes calculated for the mesh set.

4.7 Summary and Future Work

A selection of GLMH functions were calculated along with sets of CMMs for a variety of meshes. These truncated function space bases were used to reconstruct functions. The reconstruction experiments compared performance of two reconstruction methods, and the ability of the bases to accurately reconstruct different types of function. The key observations are as follows:

- Least squares reconstruction is fast and accurate.
- The WKS and segment indicator functions are reconstructed poorly in all bases.
- The constant function is poorly reconstructed by CMMs.
- Function type has greater impact on the quality of reconstruction than basis type.
- The specifically localised GLMH functions do not perform as well as expected.

It was noted that for some meshes the compressed manifold modes span a similar subspace to the Laplacian eigenfunctions. This was further investigated and a link found with the sparsity of the modes. Due to the poor notion of what constitutes a good choice for the sparsity parameter, we suggest that μ was well chosen in those cases. As mentioned in section 3.6, further work is required to understand μ .

A potential issue with compressed manifold modes is that they fail to reconstruct the constant function as well as the bases based on Laplacian eigenfunctions. This could be remedied by the addition of a function b such that $b^T A \psi_i = 1$ for all CMMs ψ_i ; $b^T A b = 1$; with a further condition about the minimisation of the reconstruction error of the constant function.

The poor performance of the localised manifold harmonics also requires future work. It would be sensible to look at the specific vertices where reconstruction errors are greatest. Is this failure a result of a loss of higher frequency functions? Similarly, where are the vertices with greatest reconstruction error for the segment indicator

functions? We suspect that they will be at the boundary of the indicated region due to the need for high frequency functions to approximate indicator functions accurately.

Chapter 5

A Comparison of Basis Methods For Calculating Functional Maps

Given two meshes N and P a *matching problem* (or *shape correspondence problem*) is a problem where the aim is to find a map $N \rightarrow P$. Matching problems might aim to find a map between identical shapes [102]; deformed shapes [103]; similar areas (partial matches), for example matches between a horse and a centaur [104]. For a review on shape correspondence problems before the emergence of the method of functional maps see the 2011 survey [105].

For shapes which are near-isometric, e.g. a set of meshes representing the same person in different positions, it is intuitive that there exists some matching between the shapes. Due to the non-rigid nature of the transformations - for example the bending of limbs - finding the map between vertex sets is non-trivial and maps can be very complex. Functional maps are linear maps between function spaces, which can be calculated given two sets of functions known to correspond. The original or underlying map between meshes can then be reconstructed. Here a background to the original functional maps formulation [5] is given. Proofs of important properties are given for the smooth case in the original work. Here the definition of discrete isometry from chapter 3 is used to justify claims for the discrete case. This leads to some simple proofs.

The prevalent method used to evaluate functional map quality is to extract a point-to-point map and compare the position of the matched points to a known ground truth match (see [5],[10],[11],[12],[13]). However, functional maps can be used to transfer functions between meshes without the need for a point-to-point match or refinement (this is noted in [80, 2.1.4]). Here two related errors for measuring the ability of a functional map to transform functions are introduced. It is proved that the most simple functional map transforms functions with the least error. The alternatives to the Laplacian eigenbasis are used to construct functional maps. These functional maps are then tested for their ability to transform the function types used in the previous chapter.

5.1 Background: Functional Maps

This section provides an introduction to the original method of functional maps as introduced by Ovsjanikov et al [5] and details some further developments.

Definition 5.1.1: Let $\mathcal{F}(N, \mathbb{R})$ denote the set of real-valued functions $f : N \rightarrow \mathbb{R}$ on the mesh N , and let $\mathcal{F}(P, \mathbb{R})$ denote the set of real-valued functions $g : P \rightarrow \mathbb{R}$ on the mesh P . Let $T : N \rightarrow P$ be a bijective map between N and P , then given $f : N \rightarrow \mathbb{R}$, the corresponding function $g : P \rightarrow \mathbb{R}$ is defined by the composition $g = f \circ T^{-1}$. This can be written as $T_F : \mathcal{F}(N, \mathbb{R}) \rightarrow \mathcal{F}(P, \mathbb{R})$, where $T_F(f) = g$. The map T_F is called a **functional representation** or **functional map**.

The function space of a mesh N can be equipped with an orthonormal basis $\{\phi_i\}$ so that functions can be represented as linear combinations of basis functions:

$$f = \sum_i a_i \phi_i.$$

Then $T_F(f) = T_F(\sum_i a_i \phi_i) = \sum_i a_i T_F(\phi_i)$.

If the mesh P has function space with basis functions $\{\psi_j\}$ then $T_F(\phi_i) = \sum_j c_{ij} \psi_j$ for some c_{ij} , so

$$T_F(f) = \sum_i a_i \sum_j c_{ij} \psi_j = \sum_j \sum_i a_i c_{ij} \psi_j.$$

Note, c_{ij} is the j -th coefficient of $T_F(\phi_i)$ in the basis $\{\psi_j\}$. If $\{\psi_j\}$ is orthonormal with respect to an inner product $\langle \cdot, \cdot \rangle$, then

$$c_{ij} = \langle T_F(\phi_i), \psi_j \rangle. \quad (5.1.1)$$

In the matrix notation of previous chapters

$$C = (\Pi^T \Phi)^T M \Psi,$$

where Φ is that matrix with ϕ_i as columns; Π is the permutation induced by T ; and Ψ is the M -orthogonal matrix with ψ_j as columns.

Definition 5.1.2: A matrix C representing a functional map is called a **functional map matrix**.

There are several important properties. Let $T : N \rightarrow P$ be a bijection then:

- (1) T can be recovered from T_F . For a point $a \in N$ consider the indicator function $f : N \rightarrow \mathbb{R}$ where $f(a) = 1$ and $f(x) = 0, \forall x \neq a$. The image of this function is given by $g = T_F(f) = f \circ T^{-1}$ such that $g(y) = 0$ whenever $T^{-1}(y) \neq a$ and $g(y) = 1$ when $T^{-1}(y) = a$. Since T is a bijection $T^{-1}(y) = a$ for a unique y . This means that g acts as an indicator function for $T(a)$ and that if T_F is known T can be reconstructed by considering indicator functions for each point.
- (2) T_F is a linear map between function spaces.
- (3) For f represented by a vector a of coefficients a_i , and $g = T_F(f)$ represented by vector of coefficients b then

$$b_j = \sum_i a_i c_{ij},$$

where c_{ij} is independent of f , as it is determined by the basis functions.

- (4) Let $T : N \rightarrow P$ be an area-preserving bijection. If basis functions on P are A_P -orthogonal, where A_P denotes the area matrix, then the matrix C associated to the functional representation of T must be orthonormal. That is, the matrix associated to the functional representation of the bijective map $T^{-1} : P \rightarrow N$ is given by C^T .

Proof. Denote by Φ the matrix formed by the taking the vectors of ordered basis $\{\phi_i\}$ as columns, and denote by Ψ the matrix formed similarly via the basis $\{\psi_i\}$. Recall from section 3.4 that T can be represented by permutation matrix Π and that $f \circ T^{-1}$ can be written as $\Pi^T \mathbf{f}$ where \mathbf{f} is a vector representing the function $f : N \rightarrow \mathbb{R}$. Assume that Ψ is A_P -orthogonal, then the functional map matrix C is given by

$$\begin{aligned} C &= (\Pi^T \Phi)^T A_P \Psi \\ &= \Phi^T \Pi A_P \Psi. \end{aligned} \tag{5.1.2}$$

Since T is area-preserving, the area matrix A_P is given by $A_P = \Pi^T A_N \Pi$ (see equation (3.4.43)), so

$$C = \Phi^T A_N \Pi \Psi.$$

Now let $S := T^{-1} : P \rightarrow N$ be the inverse bijection, represented by Π^T and so the reverse functional map is represented by matrix D where

$$D = \Psi^T \Pi^T A_N \Phi,$$

which is exactly C^T . Since $S \circ T$ is the identity map it must be that $D^T C^T = CD = I$ and so $CC^T = C^T C = I$. That is, C is orthonormal. \square

- (5) For a set of Laplacian eigenfunctions with no repeated eigenvalues and an isometry $T : N \rightarrow P$, C is a diagonal matrix.

Proof. Recall from proposition 3.4.11 that when T is an isometry $\Psi = \Pi^T \Phi$.

So, via equation (5.1.2),

$$\begin{aligned} C &= \Phi^T \Pi A_P \Pi^T \Phi \\ &= \Phi^T A_N \Phi, \text{ since } T \text{ is an isometry,} \\ &= I, \text{ by the properties of the Laplacian eigenfunctions.} \end{aligned}$$

□

For a near-isometry C is close to being diagonal, with more variation around the higher-frequency eigenfunctions. This leads to a funnel shaped region along the main diagonal of C , as displayed in figure 5.1.

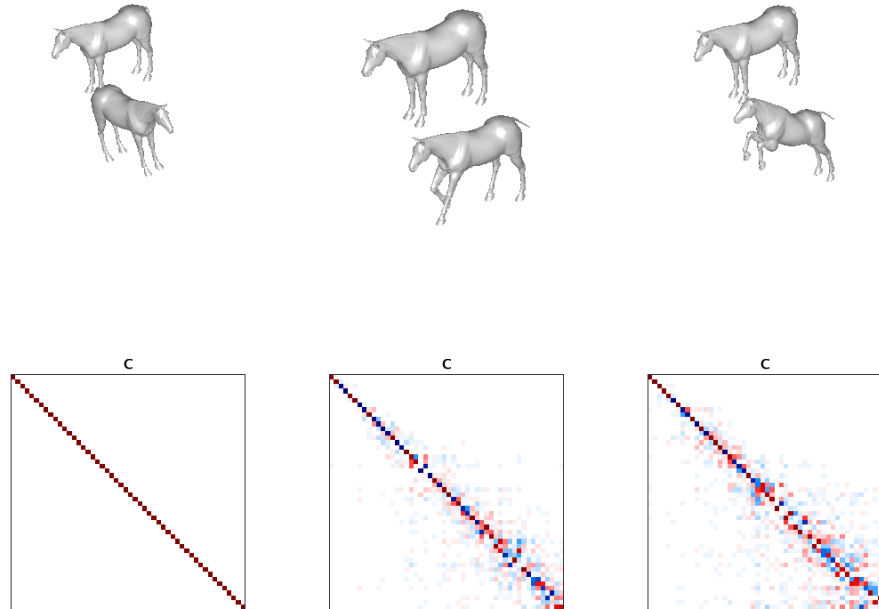


Figure 5.1: Functional map matrices between horses calculated using 50 Laplacian eigenfunctions, and a known bijection between vertex sets.

Notation 5.1.3: From here the matrix C is used to represent a functional map based on a map $N \rightarrow P$; the matrix D is used to represent a functional map in

the opposite direction, $P \rightarrow N$. The variable f is used as a scalar, representing the number of functions.

5.1.1 The original Ovsjanikov formulation

Let N be a mesh with n vertices and let P be a mesh with p vertices. Let $\{\phi_i\}$ be a basis for $\mathcal{F}(N, \mathbb{R})$ stored as columns of the matrix Φ and let $\{\psi_i\}$ be a basis for $\mathcal{F}(P, \mathbb{R})$ stored as columns of the matrix Ψ . Let \mathbf{F} be an $n \times f$ matrix formed by f vectors representing functions $N \rightarrow \mathbb{R}$. Similarly let \mathbf{G} be an $p \times f$ matrix formed by f vectors representing functions $P \rightarrow \mathbb{R}$. Written in terms of the bases $\{\phi_i\}$ and $\{\psi_i\}$

$$\mathbf{F} = \Phi A \quad \text{and} \quad \mathbf{G} = \Psi B$$

where $A \in \mathbb{R}^{n \times f}$ and $B \in \mathbb{R}^{p \times f}$ are matrices of basis coefficients.

Let C be the matrix representing the functional map $T_F : \mathcal{F}(N, \mathbb{R}) \rightarrow \mathcal{F}(P, \mathbb{R})$ then, if the functions in \mathbf{F} and \mathbf{G} are known to correspond,

$$\mathbf{G} = \Psi C^T A.$$

Then, given two corresponding matrices of known functions $\mathbf{F} = \Phi A$ and $\mathbf{G} = \Psi B$ the functional map matrix C can be found by solving

$$\arg \min_C \|B - C^T A\|_F. \quad (5.1.3)$$

This can be solved via the method of least squares with

$$C = (AA^T)^{-1} AB^T.$$

Note that to apply theorem 2.4.1 the matrix A must be such that $f > n$ and must be full rank. This is a natural consequence of a well-chosen set of functions. A rank deficient matrix A will occur when the columns of A are not linearly independent, i.e. when a function is repeated or a sum of other functions in the set. In practice functional maps are not calculated between function spaces but instead between lower dimensional subspaces spanned by a small number of basis functions (see

section 2.5.1), reducing the number of functions required to construct A . It's also important to note that the number of basis functions used can differ between the meshes.

In the original Ovsjanikov work this formulation was augmented by a commutativity condition: Let $\mathcal{O}_N : \mathcal{F}(N, \mathbb{R}) \rightarrow \mathcal{F}(N, \mathbb{R})$ and $\mathcal{O}_P : \mathcal{F}(P, \mathbb{R}) \rightarrow \mathcal{F}(P, \mathbb{R})$ be operators on the respective function spaces and let $T : N \rightarrow P$ be an isometry which defines a functional map $T_F : \mathcal{F}(N, \mathbb{R}) \rightarrow \mathcal{F}(P, \mathbb{R})$ via $T_F(f) := f \circ T^{-1}$ for $f \in \mathcal{F}(N, \mathbb{R})$. Then the operators commute with T if

$$T_F(\mathcal{O}_N(f)) = \mathcal{O}_P(T_F(f)), \quad \forall f \in \mathcal{F}(N, \mathbb{R}). \quad (5.1.4)$$

The operator \mathcal{O}_N can be represented by matrix $\bar{\mathcal{O}}_N$. The functions given by $\bar{\mathcal{O}}_N\Phi$ can be represented in the basis Φ by some matrix of coefficients S , with $\bar{\mathcal{O}}_N\Phi = \Phi S$. Similarly, let $\bar{\mathcal{O}}_P$ be the matrix representation of the operator \mathcal{O}_P . Then the functions given by $\bar{\mathcal{O}}_P\Psi$ can be represented in the basis Φ via matrix of coefficients R , with $\bar{\mathcal{O}}_P\Psi = \Psi R$. Equation (5.1.4) can then be written in matrix form as

$$\Psi C^T S = \bar{\mathcal{O}}_P \Psi C = \Psi R C^T.$$

The commutativity constraint can be written in matrix form as

$$C^T S = R C^T \quad (5.1.5)$$

with reference only to the coefficient matrices and the functional map matrix. This leads to the extended matching problem

$$\arg \min_C \|B - C^T A\|_F + \|C^T S - R C^T\|_F. \quad (5.1.6)$$

Again, this can be solved via the method of least squares as the problem can be rewritten as an over-determined system for the entries of C .

The Frobenius norm $\|M\|_F$ is minimised when the matrix M is the zero matrix, hence a matrix C is sought such that

$$A^T C = B^T, \quad (5.1.7)$$

$$S^T C - C R^T = 0. \quad (5.1.8)$$

Equation 5.1.7 is called the **function constraints** and equation 5.1.8 is called the **operator constraints**. Each of these conditions gives a system of equations for the entries of C . Converting the matrix equations into explicit linear systems requires use of the Kronecker product and the vec operator:

Definition 5.1.4: The **vec operator** $\text{vec} : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{mn}$ transforms an $m \times n$ matrix M with n columns denoted M_i into a vector by stacking columns:

$$\text{vec}(M) = \begin{bmatrix} M_1 \\ \vdots \\ M_n \end{bmatrix}$$

Definition 5.1.5: Let A be an $m \times n$ matrix and let B be a $p \times q$ matrix, then the **Kronecker product of A and B** is the $mp \times nq$ matrix

$$A \otimes B = \begin{bmatrix} A_{11}B & \dots & A_{1n}B \\ \vdots & \ddots & \vdots \\ A_{m1}B & \dots & A_{mn}B \end{bmatrix}.$$

Lemma 5.1.6: [19, equation 520] Let A and B be as above, and let X be a $n \times p$ matrix. Then, $\text{vec}(AXB) = (B^T \otimes A) \text{vec}(X)$.

Proof. Let the columns of B be denoted by B_i , so that $B = [B_1 \dots B_q]$. Similarly, let $X = [X_1 \dots X_p]$ and let $(B_k)_j$ denote the j -th component of the vector B_k . The k -th column of AXB is given by

$$\begin{aligned} AXB_k &= A \left(\sum_{j=1}^p X_j (B_k)_j \right) \\ &= \sum_{j=1}^p (B_k)_j AX_j, \text{ by properties of scalar and matrix multiplication} \\ &= \left[(B_k)_1 A \dots (B_k)_q A \right] \begin{bmatrix} X_1 \\ \vdots \\ X_p \end{bmatrix} \\ &= (B_k^T \otimes A) \text{vec}(X). \end{aligned}$$

Therefore, considering all columns of AXB ,

$$\begin{aligned} \text{vec}(AXB) &= \begin{bmatrix} (B_1^T \otimes A) \\ \vdots \\ (B_q^T \otimes A) \end{bmatrix} \text{vec}(X) \\ &= (B^T \otimes A) \text{vec}(X). \end{aligned}$$

□

Corollary 5.1.7: $\text{vec}(AX) = I_p \otimes A \text{vec}(X)$ and $\text{vec}(XB) = B^T \otimes I_n \text{vec}(X)$.

Assume that functions are written in truncated bases, using Φ_k and Ψ_k instead of Φ and Ψ . This means that

$$A, B \in \mathbb{R}^{k \times f} \quad \text{and} \quad C \in \mathbb{R}^{k \times k}.$$

The same sets of truncated basis functions are used to construct the operator constraints. So

$$R, S \in \mathbb{R}^{k \times k}.$$

Clearly this ensures an overdetermined system of equations as $f + k > k$.

The above corollaries allow the constraints (equations (5.1.7) and (5.1.8)) to be rewritten as

$$(I_k \otimes A^T) \text{vec}(C) = \text{vec}(B), \quad (5.1.9)$$

$$((I_k \otimes S^T) - (R \otimes I_k)) \text{vec}(C) = 0. \quad (5.1.10)$$

Note that $\text{vec}(B) \in \mathbb{R}^{kf}$ and the zero vector is in \mathbb{R}^{k^2} . Let $\alpha_F := (I_k \otimes A^T)$ and $\alpha_{Op} = ((I_k \otimes S^T) - (R \otimes I_k))$. The system of $k(f + k)$ linear equations

$$\begin{bmatrix} \alpha_F \\ \alpha_{Op} \end{bmatrix} \text{vec}(C) = \begin{bmatrix} \text{vec}(B) \\ 0 \end{bmatrix} \quad (5.1.11)$$

is then an overdetermined system for the entries of C . To simplify notation, let $\alpha := \begin{bmatrix} \alpha_F \\ \alpha_{Op} \end{bmatrix}$ and let $\beta := \begin{bmatrix} \text{vec}(B) \\ 0 \end{bmatrix}$. The aim is then to find a vector x which

minimises $\|\alpha x - \beta\|_F$, a least squares problem. The matrix α is full rank if A and S are full rank. In practice the backslash operator is used in MATLAB to solve least squares problems. See [106] for details of methods involving matrix decomposition to solve least squares problems.

Note that minimising $\|\alpha x - \beta\|_F$ as constructed above is not equivalent to the extended matching problem (equation (5.1.6)), but gives instead

$$\arg \min_C (\|B - C^T A\|_F^2 + \|C^T S - RC^T\|_F^2)^{\frac{1}{2}}. \quad (5.1.12)$$

5.1.2 Coupled Bases

For near-isometric shapes the matrix C has a diagonal structure due to the fact that when T is very close to being an isometry $\psi_i \approx \phi_i \circ T^{-1}$. This is not true when meshes are non-isometric. To maintain the diagonal structure in the functional map matrix, we seek a pair of new orthogonal function space bases $\{\hat{\phi}_i\}_{i=1}^{\hat{k}}$, $\{\hat{\psi}_i\}_{i=1}^{\hat{k}}$, where \hat{k} is the number of eigenfunctions used in the creation of the new basis. These new bases will produce a functional map matrix with the desired near-diagonal structure, and are constructed to be linear combinations of Laplacian eigenfunctions. The idea originates in [107], [108] and applied to functional maps in [109].

Let

$$\hat{\phi}_i = \sum_{j=1}^k q_{ji} \phi_j, \quad \hat{\psi}_i = \sum_{j=1}^k r_{ji} \psi_j \quad (5.1.13)$$

denote the new basis functions and \hat{k} be the number of basis functions in the new bases. That is,

$$\hat{\Phi} = \Phi_k Q \quad \text{and} \quad \hat{\Psi} = \Psi_k R$$

where the matrices Q and R are $k \times \hat{k}$ matrices of linear combination coefficients. The aim is to choose new bases so that they are A orthogonal, with $\hat{\Phi}^T A_N \hat{\Phi} = I_{\hat{k}}$ and $\hat{\Psi}^T A_P \hat{\Psi} = I_{\hat{k}}$. Given that $\Phi_k^T A_N \Phi_k = I_k$ the orthogonality condition holds if

$$\hat{\Phi}^T A_N \hat{\Phi} = (\Phi_k Q)^T A_N (\Phi_k Q) = Q^T \Phi_k^T A_N \Phi_k Q = Q^T Q = I_{\hat{k}}$$

(and similarly $R^T R = I_{\hat{k}}$).

It is also desirable that the new basis functions behave like eigenfunctions of the Laplacian. This happens if they minimise the Dirichlet energy, $\text{tr}(\hat{\Phi}^T W_N \hat{\Phi})$, where W_N is the weight matrix of the Laplacian $L_N = A_N^{-1} W_N$ (see section 2.5.2). Using the fact that $\hat{\Phi} = \Phi_k Q$,

$$\text{tr}(\hat{\Phi}^T W_N \hat{\Phi}) = \text{tr}(Q^T \Phi_k^T W_N \Phi_k Q) = \text{tr}(Q^T \Lambda_N Q),$$

by equation (2.5.7), where Λ_N is the diagonal matrix of the first k Laplacian eigenvalues. (Similarly, $\text{tr}(\hat{\Psi}^T W_P \hat{\Psi}) = \text{tr}(R^T \Lambda_P R)$.)

The following minimisation problem can be solved for the coefficient matrices Q and R which define the new bases:

$$\begin{aligned} \min_{Q,R} \text{tr}(Q^T \Lambda_N Q) + \text{tr}(R^T \Lambda_P R) + \mu \|Q^T A - R^T B\|_F \quad (5.1.14) \\ \text{s.t. } Q^T Q = I, R^T R = I. \end{aligned}$$

The matching problem can then be formulated as in equation (5.1.3) using the function coefficient matrices \hat{A} and \hat{B} .

5.1.3 Sparse Matching

Assume that the same number of basis functions are used to construct functions for each mesh, resulting in a square C matrix. The above methods assume the the correspondence is known between pairs of functions \mathbf{f} and \mathbf{g} and that they are arranged consistently in the coefficient matrices A and B . If the functions are not arranged consistently then there is a permutation on the columns of B which correspond to the columns of A . Let Π be the $f \times f$ matrix representing this permutation, then the functional constraint (equation 5.1.7) becomes

$$\Pi B^T = A^T C.$$

However, it could be that the number of functions differs between meshes. Given this set-up a possible method to find the functional map matrix is presented in [10].

Without loss of generality, let there be f functions \mathbf{f}_i defined on N and g functions \mathbf{g}_j defined on P with $f \leq g$. Now, instead of the one-to-one correspondences mentioned above, Π is an $f \times g$ matrix which has some zero columns. To calculate Π assume that all rows of A^T are matched to some row in B^T , and use a row-sparse $f \times l$ outlier matrix O to remove any mismatches. That is,

$$\Pi B^T = A^T C + O. \quad (5.1.15)$$

To find Π and C , keeping in mind that C should be sparse and nearly-diagonal, the aim is to minimise

$$\frac{1}{2} \|\Pi B^T - A^T C - O\|_F^2 + \lambda \|D \odot C\|_1 + \mu \|O^T\|_{2,1}. \quad (5.1.16)$$

where the norms are as defined in section 2.1. The matrix $D \in \mathbb{R}^{k \times k}$ is used to promote diagonal solutions. It consists of small valued D_{ij} close to the diagonal, with large valued D_{ij} further away from the diagonal. The symbol \odot denotes element-wise multiplication and the non-negative parameter λ controls the level of such promotion. In summary – the first term aims to minimise the difference between ΠB and $(AC + O)$; the second term promotes diagonal solutions via the weighted matrix D ; the final term controls the row-wise sparsity of the matrix O , via the non-negative parameter μ .

The minimisation problem (5.1.16) can be split into two problems. Firstly fix Π , resulting in

$$\min_{C, O} \frac{1}{2} \|\Pi B^T - A^T C - O\|_F^2 + \lambda \|D \odot C\|_1 + \mu \|O\|_{2,1}. \quad (5.1.17)$$

Secondly fix both C and O , resulting in

$$\max_{\Pi} \text{vec}((A^T C + O)B) \text{vec}(\Pi) \text{ s.t. } \begin{cases} \Pi \mathbf{1} = \mathbf{1}, \\ (\Pi^T \mathbf{1})_i = 1 \text{ for all } i = 1, \dots, f, \end{cases} \quad (5.1.18)$$

where $\mathbf{1}$ is the f -dimensional vector with each element equal to 1. Note that this maximisation arises from minimising a trace where the only non-constant term is $-\text{tr}(B^T \Pi^T (A^T C + O))$.

The following algorithm can be used to solve 5.1.17:

- 1: Fix Π , A , B , parameters λ and μ and step-size α .
- 2: Set initial values $O^0 = \Pi B^T$ and $C^0 = 0$.
- 3: **repeat**

$$\begin{aligned} C^{k+1} &= P_1\left(\left(I - \frac{1}{\alpha}AA^T\right)C^k - \frac{1}{\alpha}A(O^k - \Pi B^T)\right) \\ O^{k+1} &= P_2\left(\left(I - \frac{1}{\alpha}\right)O^k - \frac{1}{\alpha}(A^T C^k - \Pi B^T)\right) \end{aligned}$$

- 4: **until** convergence

where

$$\alpha \leq \text{the largest eigenvalue of } \begin{bmatrix} AA^T & A \\ A^T & I \end{bmatrix}, \quad (5.1.19)$$

$$P_1(C) = \max\{|C| - \frac{\lambda}{\alpha}W, 0\} \odot \text{sgn}(C), \quad (5.1.20)$$

and the i -th row of P_2 is given by

$$P_2(O) = \max\{\|o_i^T\|_2 - \frac{\mu}{\alpha}, 0\} \frac{o_i^T}{\|o_i^T\|_2}, \quad (5.1.21)$$

where o_i denotes the i -th of row O . The $P_i(\cdot)$ terms come from the use of a proximal gradient method [10, p.12]. (More details about the proximal operator can be found in section 3.2, see definition 3.2.11.) An acceleration step is also used.

The solution to (5.1.18) can be obtained via an inverted Hungarian algorithm. This is a method of finding an optimal cost, which allows the selection of the greatest trace [110]. These two part-solutions are then alternated between until an overall convergence is reached, solving the initial problem, (5.1.16).

5.1.4 Point-to-point maps from functional maps

Let function $\iota_x : N \rightarrow \mathbb{R}$ be an indicator function such that, for all $\bar{x} \in N$

$$\iota_x(\bar{x}) = \begin{cases} 0, & \text{if } \bar{x} \neq x, \\ 1, & \text{if } \bar{x} = x. \end{cases}$$

Let T_F be a functional map. Then $g = T_F(\iota_x)$ is a function on P and the point y^* , such that

$$y^* = \arg \max_{y \in P} g(y),$$

can be considered to be the corresponding point.

In matrix form, let $\iota_x = \Phi_k a$ be the indicator function represented as a vector of basis coefficients, then $\mathbf{g} = \Psi_l C^T a$ and the index of the corresponding point is given by $\max_i \mathbf{g}_i$.

To do this for every vertex of N requires $O(np)$ operations [5, section 6.1], expensive for large meshes. Instead, as in [5] consider the coefficients of indicator functions when represented in the function space basis. (Note that the paper references an alternative proof.)

Proposition 5.1.8: The indicator function for vertex v_j is given by

$$\iota_{v_j} = A\Phi r_j^T$$

where r_j denotes the j -th row of Φ .

Proof. First note that, given $n \times n$ Φ such that $\Phi^T A \Phi = I$, Φ is invertible and hence

$$A = (\Phi\Phi^T)^{-1}.$$

As A is positive definite, A is invertible so

$$A^{-1} = \Phi\Phi^T. \tag{5.1.22}$$

Note also that $(\iota_{v_j})_i = \delta_{ij}$. Then consider the i -th element of $A\Phi r_j^T$:

$$\begin{aligned} [A\Phi r_j^T]_i &= \sum_k (A\Phi)_{ik} (r_j^T)_k \\ &= \sum_k (A\Phi)_{ik} \Phi_{jk} \\ &= \sum_k A_{ii} \Phi_{ik} \Phi_{jk} \\ &= A_{ii} (\Phi\Phi^T)_{ij} \\ &= A_{ii} A_{ij}^{-1}, \text{ by equation (5.1.22),} \\ &= \delta_{ij}. \end{aligned}$$

□

Therefore the function given by

$$\mathbf{f} := A^{-1}\boldsymbol{\nu}_{v_j} = A^{-1}A\Phi r_j^T$$

(called the **scaled indicator function**) has corresponding function

$$\mathbf{g} := \Psi C^T r_j^T$$

on P . The matrix of all scaled indicator functions is given by

$$\mathbf{F} := A^{-1}I = (A^{-1}A)\Phi\Phi^T$$

which maps via the functional map to

$$\mathbf{G} := \Psi C^T \Phi^T.$$

Following the above argument for the construction of scaled indicator functions on N , the set of scaled indicator functions on P are given by $\Psi\Psi^T$. To find the point-to-point match, for each column in \mathbf{G} find the nearest column in $\Psi\Psi^T$. Let $\mathbf{x} = \Psi a$, $\mathbf{y} = \Psi b$ be vectors constructed as linear combination of basis vectors. Distance between vectors can be measured as distance between coefficient vectors as

$$\|\mathbf{x} - \mathbf{y}\|_A = \|a - b\|_2$$

(for proof see A.11). Therefore correspondence can be found by matching each column of $C^T\Phi^T$ with its nearest neighbour in Ψ^T . Note that since there are n columns of $C^T\Phi^T$ and p columns of Ψ^T this is not necessarily a bijection.

Definition 5.1.9: A **ground truth match** is a known matching between meshes.

The ground truth matching is usually defined as a point-to-point match between the vertex sets. Again, it may not be a bijection. Some datasets are provided with ground truth matches for testing purposes (e.g. [111], [112]).

5.1.5 Iterative Closest Point Refinement

Following [5], a given functional map C can be refined to make it closer to a point-to-point map. Assume that there is an underlying point-to-point map. Then it must be that each column of $C^T\Phi^T$ corresponds to a column of Ψ^T , where notation is as above.

Assume that the same number of basis functions are used to represent the (truncated) function spaces of the two meshes, and assume that the meshes are isometric. Then the matrix C is expected to be orthonormal (property (4)) so refinement of C can be given by the following algorithm:

- 1: **repeat**
- 2: Match columns x of $C^T\Phi^T$ to nearest column y in Ψ^T .
- 3: Find optimal orthonormal \tilde{C} minimising $\sum\|\tilde{C}x - y\|_2$ (via singular value decomposition).
- 4: Set $C = \tilde{C}$.
- 5: **until** convergence

Note that due to the potential for corresponding eigenfunctions to differ by a sign-flip this method cannot be used without an existing estimate for C . This method is analogous to the iterative closest point refinement (ICP) of [113], but takes place in the function space rather than attempting an alignment of the mesh vertices, and hence refinement is referred as being via ICP. In practice the refinement algorithm is run for a fixed number of iterations.

Figure 5.2 shows a functional map matrix calculated between horse0 and horse10, via the original Ovsjanikov formulation, then refined via ICP. Note how refinement makes the functional map matrix more like the “true” functional map matrix, as constructed in figure 5.1 (right).

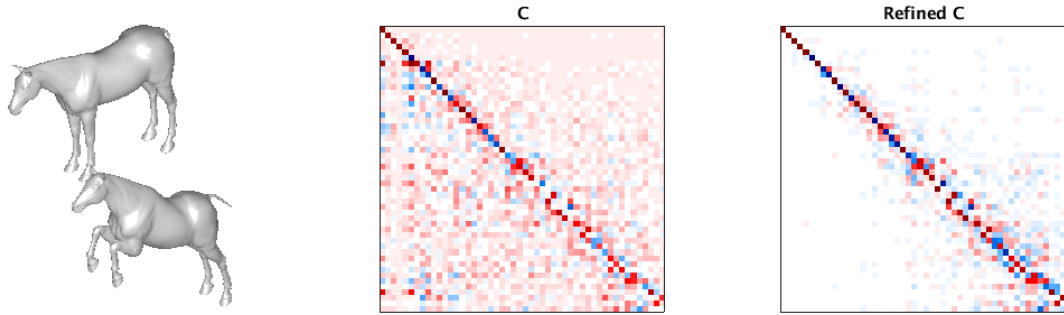


Figure 5.2: A functional map matrix and the resulting refinement via ICP. (Displayed is the upper-left 50×50 sub-matrix of the functional map calculated in chapter 5, using the LBO basis.)

5.1.6 Further work

There is a considerable amount of other work on the method of functional maps for shape correspondence problems. Reviews can be found in [114] and [115] (which also provides a good general introduction). One significant work is [116] which considers functional maps between shapes with holes or missing parts.

Alternative refinement methods with greater success in retrieving a point-to-point match can be found in [75] and [79], the latter permitting point-to-face matches.

5.2 Measuring Quality of a Functional Map

Let N be a mesh with n vertices and A_N -orthogonal (truncated) function space basis represented as columns of an $n \times k$ matrix Φ , where A_N is a (lumped) area matrix. Let \mathbf{F} be an $n \times f$ matrix with columns given by known functions evaluated on the vertices of N . Let ΦA be the reconstruction of \mathbf{F} in the basis Φ , where A is a $k \times f$ matrix of coefficients.

Similarly, let P be a mesh with p vertices and A_P -orthogonal (truncated) function space basis represented as columns of $p \times k$ matrix Ψ , where A_P is an area matrix.

Let \mathbf{G} be an $p \times f$ matrix with columns given by known functions evaluated on the vertices of P . Let ΨB be the reconstruction of \mathbf{G} in the basis Ψ , where B is a $k \times f$ matrix of coefficients.

Then, given a functional map between the (truncated) function spaces of N and P , represented by matrix C , the difference between the known functions \mathbf{G} and the transformed functions $\Psi C^T A$ can be measured in a simple way by taking the Frobenius norm,

$$Err_S(C) := \|\mathbf{G} - \Psi C^T A\|_F. \quad (5.2.23)$$

Definition 5.2.1: $Err_S(C)$ is called the **simple transformation error**.

An error based on the reconstruction error (see definition 4.2.2) can also be defined as

$$Err_T(C) := Err(\mathbf{G}, \Psi C^T A) = \frac{1}{p} \sum_{i=1}^p \sqrt{\sum_{j=1}^f |\mathbf{G}_{ij} - (\Psi C^T A)_{ij}|^2}. \quad (5.2.24)$$

This calculates the distance between the known functions and the functions obtained via the functional map at each vertex, then takes a mean.

Definition 5.2.2: $Err_T(C)$ is called the **function transformation error**.

The simple transformation error gives an average error between the two sets of functions on N , the known functions and the transformed functions. The function transformation error takes an average over the vertex set, so is skewed by sets of transformed functions which perform badly at specific vertices. That is, a low simple transformation error and a large function transformation error implies that the transformed functions are close to the known functions with large errors at specific vertices.

The simple transformation error and the function transformation error are related via the following lemma.

Lemma 5.2.3: Let $\mathcal{S} := Err_S(C)$ and let $\mathcal{T} := Err_T(C)$. Then

$$\mathcal{S}^2 = p^2 \mathcal{T}^2 - \sum_{i \neq j} \|E_i\|_2 \|E_j\|_2, \quad (5.2.25)$$

where E_i denotes the i -th row of $\mathbf{G} - \Psi C^T A$.

Proof. Define $E := (\mathbf{G} - \Psi C^T A)^T$. Note that the E_i are given by the *columns* of E . Then, since for any matrix M , $\|M\|_F = \|M^T\|_F$, $\mathcal{S} = \|E\|_F$. Consider the square of the Frobenius norm which can be expressed as a sum of ℓ_2 vector norms:

$$\mathcal{S}^2 = \|E\|_F^2 = \sum_{j=1}^p \|E_j\|_2^2.$$

The value \mathcal{T} can also be expressed in terms of the E_i , as

$$\mathcal{T} = \frac{1}{p} \sum_{i=1}^p \|E_i\|_2$$

and so

$$\begin{aligned} p^2 \mathcal{T}^2 &= \left(\sum_{i=1}^p \|E_i\|_2 \right)^2 \\ &= \sum_{i=1}^p \|E_i\|_2^2 + \sum_{i \neq j} \|E_i\|_2 \|E_j\|_2 \\ &= \mathcal{S}^2 + \sum_{i \neq j} \|E_i\|_2 \|E_j\|_2. \end{aligned}$$

Rearranging gives the result. □

When a ground truth match is known between two meshes, let this define a map $gt : N \rightarrow P$. Extract a point-to-point match $\rho : N \rightarrow P$ from the functional map (see section 5.1.4). Ideally the geodesic distance between $gt(v)$ and $\rho(v)$ for every vertex $v \in N$ would be calculated and averaged to obtain an error value. However, calculating geodesic distance on meshes is computationally expensive so instead approximate via *average geodesic distance units*, constructed by dividing the Euclidean distance between the points by the average edge length. An error based on the geodesic error (similar to that of [81, section 8.2]) can then be defined as

$$Err_{GD}(C) := \frac{1}{\text{average edge length}} \sum_{i=1}^n \|gt(v_i) - \rho(v_i)\|_2. \quad (5.2.26)$$

Definition 5.2.4: $Err_{GD}(C)$ is called the **geodesic functional map error**.

As the geodesic distance is used in existing works it is the obvious choice, even if taking an approximation. One work which uses an alternative method for comparison is [78] which uses the biharmonic distance [82].

5.2.1 Measuring Isometry Failure

Recall from the properties of functional maps, that given C representing $T_F : \mathcal{F}(N, \mathbb{R}) \rightarrow \mathcal{F}(P, \mathbb{R})$ the reverse functional map $T_F^{-1} : \mathcal{F}(P, \mathbb{R}) \rightarrow \mathcal{F}(N, \mathbb{R})$ is represented by C^T (property (4)). Let D be a functional map matrix representing the functional map in the opposite direction to that represented by C . Then the failure of the functional map to meet this property can be evaluated via $\|D - C^T\|_F$. Given that the property that $D = C^T$ relies on the assumption that the meshes are isometric this can be considered as a method of assessing how far from being isometric a pair of meshes are.

Definition 5.2.5: The value $\|D - C^T\|_F$ is referred to as the **C-orthogonality error**.

5.3 Functional map comparison experiments

A total of 300 functional maps were calculated (15 pairs of meshes (N, P) , 10 basis types, maps calculated $N \rightarrow P$ and $P \rightarrow N$). The basis types used were the basis types listed in section 4.3 as well as a set of compressed manifold modes calculated using ADMM as in section 3.2.2, with $\mu = 0.008$ (labelled CMM ADMM). The original Ovsjanikov formulation of the problem was used (see section 5.1.1), with functional constraints provided by a set of corresponding functions described below. Commutativity constraints were constructed using the weight matrix W of the calculated discrete Laplacian L . The weight matrix necessarily commutes with isometry (see definition 3.4.8), and by considering the commutation of the

weight matrix rather than the Laplacian results in commutativity constraints with a uniform weighting for each vertex. Functional map matrices were then refined using ICP (see section 5.1.5).

5.3.1 Choice of functional constraints

Point-based functions such as delta functions were not used in the set of functional constraints as the landmarked points were not guaranteed to correspond between meshes. Instead 30 heat kernel signature functions, 100 wave kernel signature functions and segment indicator functions were used to construct the functional constraints. Despite the poor performance of segment indicator functions in chapter 4 using such functions helps avoid errors due to symmetry. These functions were calculated as described in section 4.1. Segmentation indicator functions were rejected if the segmentation resulted in a single segment or if there was no clear correspondence between segments for a specific mesh pair. As mentioned in section 4.1, the fish and victoria meshes do not have segment indicator functions as the segmentation method resulted in a single segment; the chair meshes have segmentation functions removed from the set of descriptor functions as the segmentations did not have corresponding segments.

5.3.2 Expectations and Results

Refinement is likely to have a negative effect on the simple transformation error and hence also the function transformation error. To see this consider the most simple form of the functional map problem, where the functional map matrix C minimises $\|B - C^T A\|_F$.

Theorem 5.3.1: Let $B \in \mathbb{R}^{l \times f}$, $A \in \mathbb{R}^{k \times f}$, with A full rank and let C be such that

$$C = \arg \min_{X \in \mathbb{R}^{k \times l}} \|B - X^T A\|_F.$$

Let $\Psi \in \mathbb{R}^{n \times l}$ and let Ψ be M -orthogonal for positive definite $M \in \mathbb{R}^{n \times n}$. Then if Ψ is full rank, C also minimises

$$\arg \min_{X \in \mathbb{R}^{k \times l}} \|\Psi B - \Psi X^T A\|_F.$$

Proof. First note that if Ψ is such that $\Psi^T P \Psi = I$ (i.e. $M = I$) then proof is clear (see lemma A.10). However this does not simply generalise for the $M \neq I$ case.

Recall that via least squares minimisation (section 2.4.1) C is given by

$$C = (AA^T)^{-1}AB^T. \quad (5.3.27)$$

Now consider the matrix \bar{C} such that

$$\bar{C} = \arg \min_X \|\Psi B - \Psi X^T A\|_F.$$

To solve for \bar{C} first note that

$$\begin{aligned} \bar{C} &= \arg \min_X \|\Psi B - \Psi X^T A\|_F^2 \\ &= \text{tr}(B^T B) + \text{tr}(A^T X \Psi^T \Psi X^T A) - 2 \text{tr}(B^T \Psi^T \Psi X^T A) \\ &= \text{tr}(B^T B) + \text{tr}(\Psi X^T A A^T X \Psi^T) - 2 \text{tr}(X^T A B^T \Psi^T \Psi), \end{aligned}$$

by the cyclic property of trace. Differentiating with respect to X and equating with zero gives

$$2AA^T X \Psi^T \Psi - 2AB^T \Psi^T \Psi = 0 \text{ via [19, equations (103) and (116)].}$$

Rearranging gives

$$X \Psi^T \Psi = (AA^T)^{-1}AB^T \Psi^T \Psi.$$

Then, since $\Psi^T \Psi$ is invertible (see lemma A.1), multiply on the right by $(\Psi^T \Psi)^{-1}$ to get

$$X = (AA^T)^{-1}AB^T,$$

which is exactly C from equation 5.3.27. \square

That is, the C which is obtained as a functional map matrix is the same matrix which minimises the simple transformation error. Of course, when C is calculated with the additional commutativity constraints as well as the functional constraints this does not necessarily hold. Note that this reasoning is not dependent on the refinement being via ICP, so remains true for alternative refinement methods.

Figures 5.3 and 5.4 corroborate this expectation. The solid markers show the error calculated using the unrefined functional map matrix C and the unfilled markers show the error calculated using functional map matrices refined via ICP. Markers are slightly offset for ease of reading. The errors when using the unrefined matrices are less than the errors when using the refined matrices, showing that refinement has a negative effect on the accuracy of function reconstruction. The tight clustering of the unrefined errors also implies that there is no significant difference in the performance of each basis type.

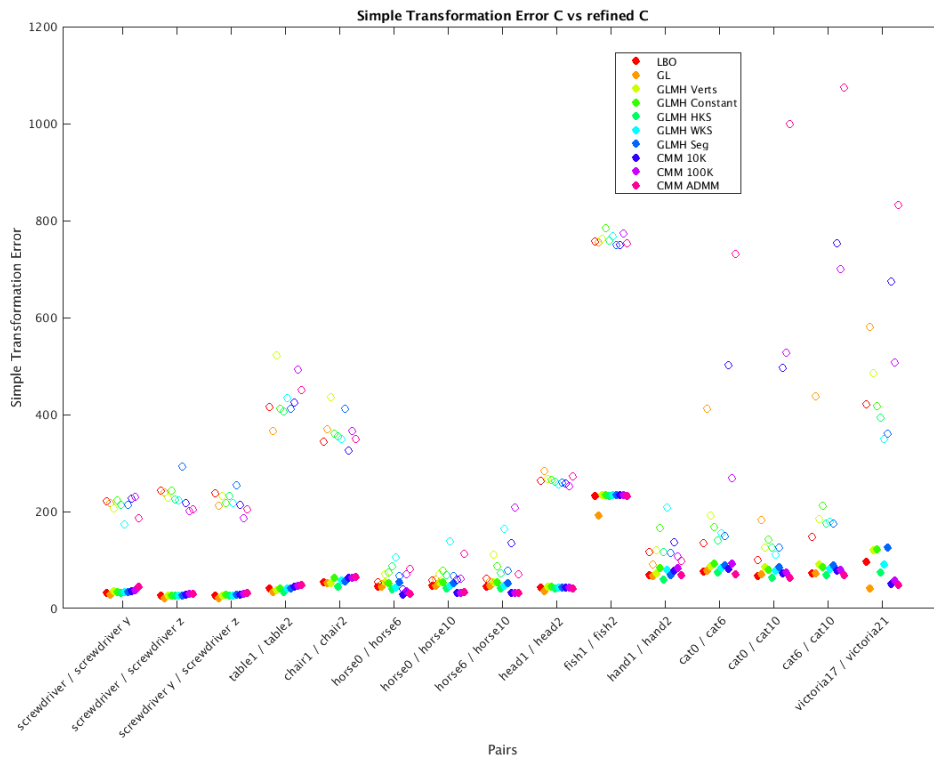


Figure 5.3: Simple transformation error – the effect of refinement.

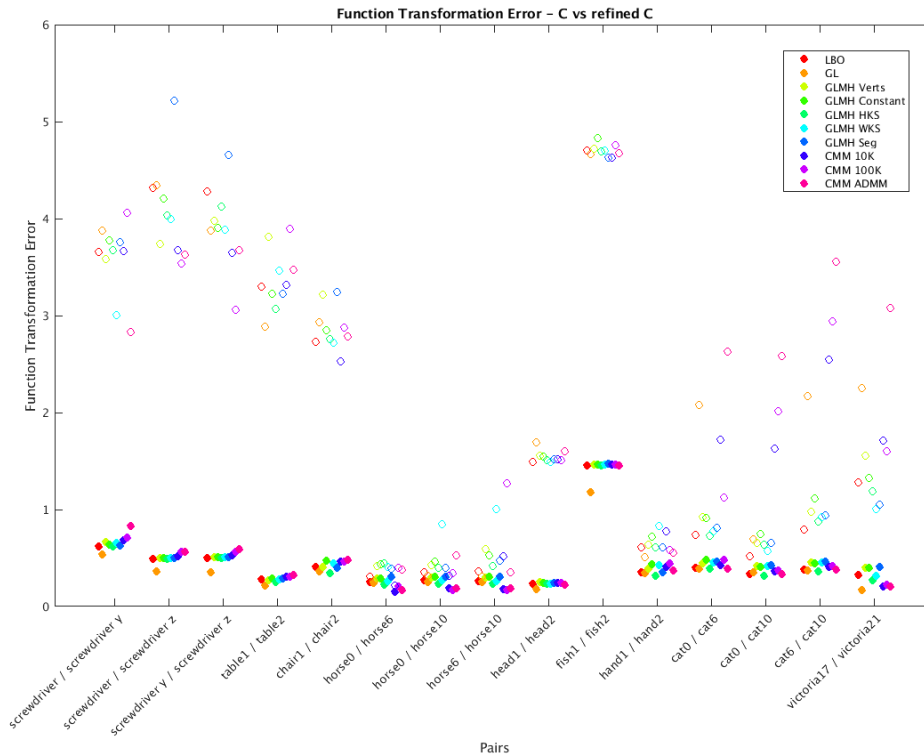


Figure 5.4: Function transformation error – the effect of refinement.

Theorem 5.3.2: Let $T : N \rightarrow P$ be an isometry between meshes. Let C the functional map matrix representing $T_F : \mathcal{F}(N, \mathbb{R}) \rightarrow \mathcal{F}(P, \mathbb{R})$ and let D be the matrix representing the inverse functional map $T_F^{-1} : \mathcal{F}(P, \mathbb{R}) \rightarrow \mathcal{F}(N, \mathbb{R})$. Then $Err_S(C) = Err_S(D)$.

Proof. Recall that when a pair of meshes are isometric and the functional map matrices are constructed using A -orthogonal bases then $D = C^T$ and both D and C are orthogonal (see property (4)). Therefore, due to the orthogonality of D ,

$$\begin{aligned}
 \|B - C^T A\|_F &= \|D^T(B - C^T A)\|_F, \text{ via lemma A.10,} \\
 &= \|D^T B - D^T C^T A\|_F \\
 &= \|D^T B - A\|_F, \text{ since } CC^T = I \text{ and } D^T = C, \\
 &= \|A - D^T B\|_F.
 \end{aligned}$$

Define

$$E := B - C^T A \quad \text{and} \quad H := A - D^T B \quad (5.3.28)$$

and note that

$$C^T H = -E. \quad (5.3.29)$$

Let Π be the permutation matrix representing T , then the isometry take functions Φ to functions $\Pi^T \Phi$ on P . Equally, in the language of functional maps Φ maps to ΨC^T so

$$\Pi^T \Phi = \Psi C^T \quad (5.3.30)$$

Now consider the simple transformation error:

$$\begin{aligned} Err_S(D) &= \|\Phi(A - D^T B)\|_F \\ &= \|\Phi H\|_F \\ &= \|\Pi^T \Psi C^T H\|_F, \text{ via equation (5.3.30)} \\ &= \|\Psi C^T H\|_F, \text{ via lemma A.10,} \\ &= \|\Psi E\|_F, \text{ via 5.3.29,} \\ &= Err_S(C). \end{aligned}$$

□

That is, for functional maps calculated in both directions between isometric meshes, the simple transformation errors are equal. Therefore, for near-isometric meshes it is expected that the transformation errors calculated in both directions are very similar. This is verified in figure 5.5 which compares the simple transformation error between the functional map C and the reverse functional map D . Solid markers denote the originally calculated C matrices and unfilled markers denote the D matrices calculated in the opposite direction. The solid and unfilled markers appear side-by-side for many of the meshes, with the horse meshes as the best example. Note that the meshes with greatest difference between the errors when the map direction is changed are the fish and head meshes. These are pairs which are not isometric – they do not have the same number of vertices.

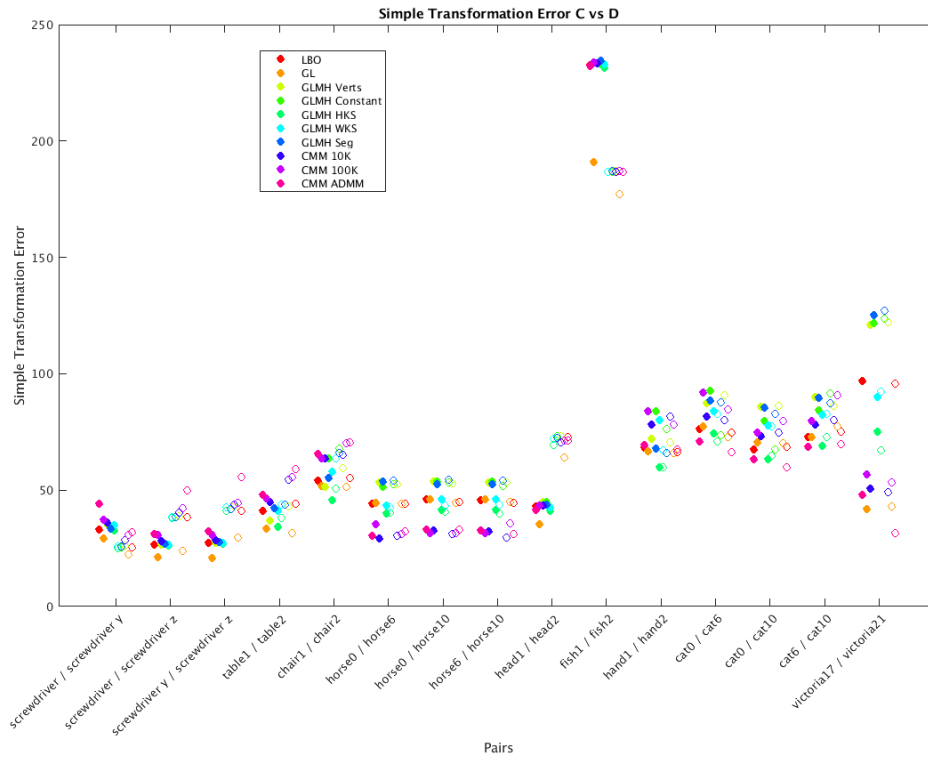


Figure 5.5: Simple transformation error – comparing functional map direction.

Figure 5.5 also allows a closer look at the basis type clusters. There is no clear trend in the performance of the basis types across the pairs, however trends do appear for related meshes – consider the performance of the CMM bases for the horse pairs, consistently resulting in the lowest error. This suggests that that different basis types may perform better than others for specific shape collections. Following refinement these trends are not as clear. This can be seen in figure 5.6 where solid markers denote the refined C matrices and unfilled markers denote refined D matrices. It can also be seen that although the errors have increased, refinement has brought to C and D errors together for the head and fish meshes. This is to be expected because the ICP refinement relies on the assumption that meshes are isometric.

Figure 5.7 plots the C -orthogonality error and shows extreme errors in the GL basis, particularly for the pairs of meshes known not to be near-isometric. This is because the graph Laplacian sees only the underlying graph structure of the

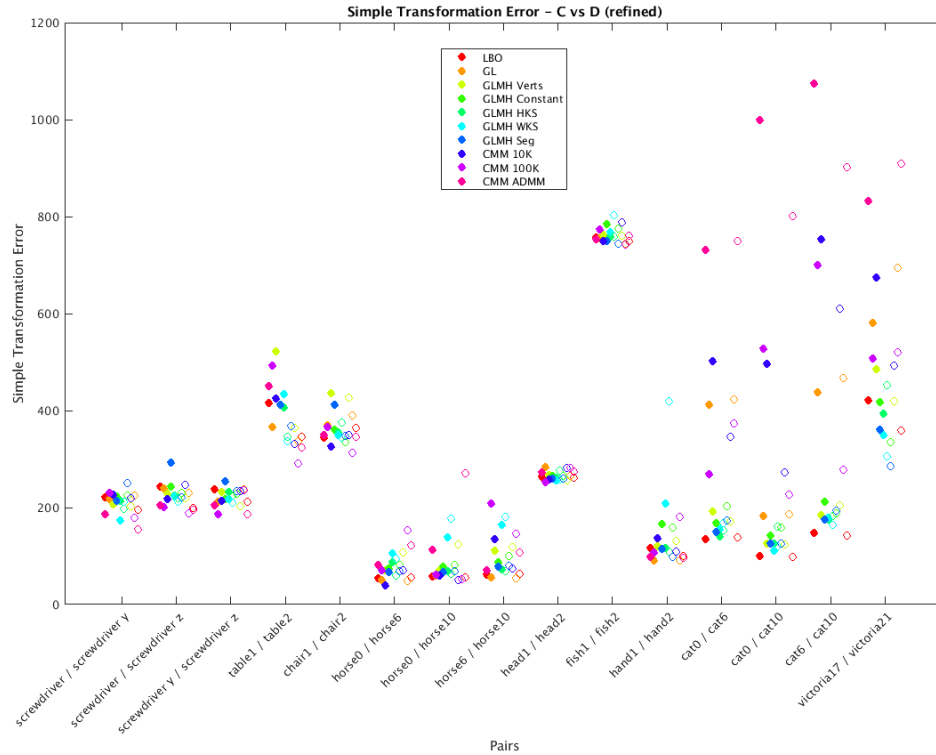


Figure 5.6: Simple transformation error – comparing refined functional map direction.

meshes. This provides a warning against testing only data sets where meshes are constructed with the same underlying graph structure, as they lead to improvement in results. Figure 5.8 shows the same data with the GL basis results removed. If C -orthogonality error is increased following refinement this indicates that the isometry assumption is flawed. Figure 5.9 shows the error after refinement. Note that following refinement functional maps calculated via the GL basis no longer stand out as performing poorly. As above there is no clear trend for a basis type which performs best, but there are micro-trends across the groups of mesh pairs, for example the superior performance of the LBO and GL bases for the horse meshes.

The aim of refinement is to improve the quality of a point-to-point map, measured via the geodesic functional map error, and hence the expectation is that refined functional map matrices reduce the geodesic functional map error. Figure 5.10 verifies this. The solid markers denote the geodesic functional map error using an

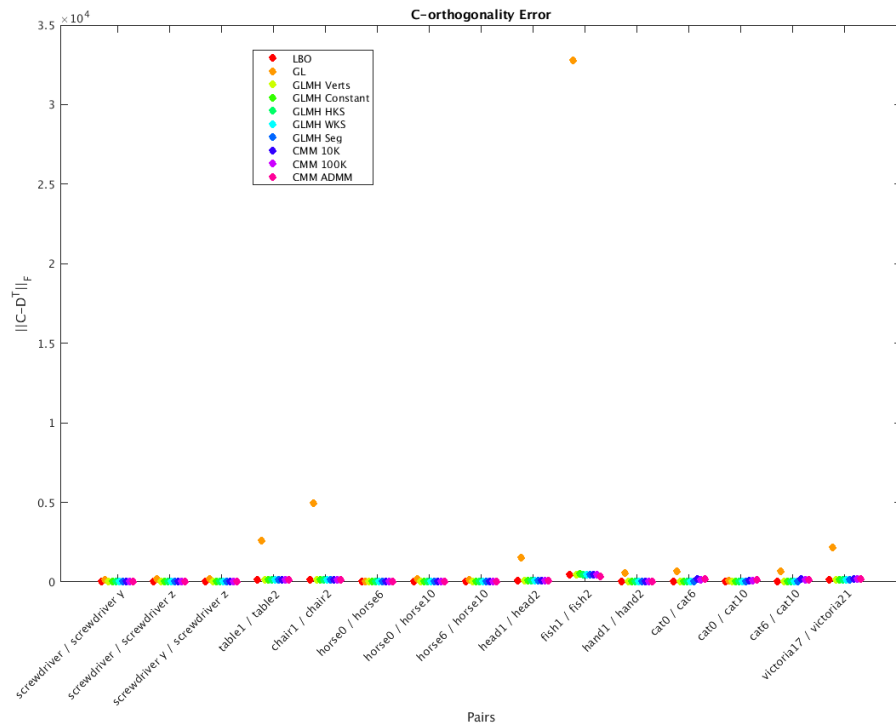


Figure 5.7: C -orthogonality error

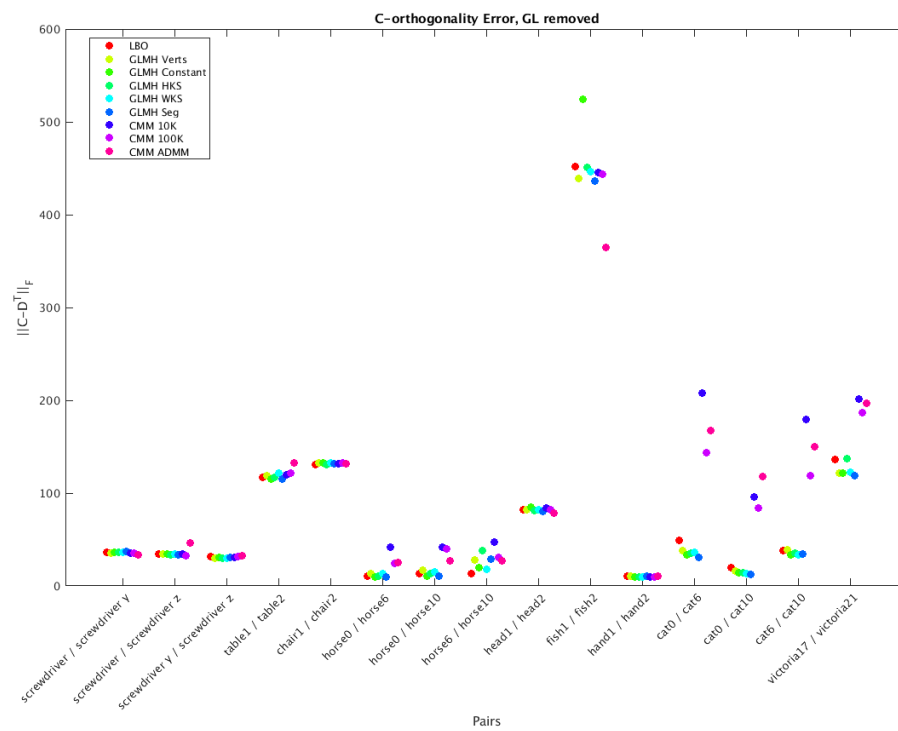


Figure 5.8: C -orthogonality error, GL basis removed.

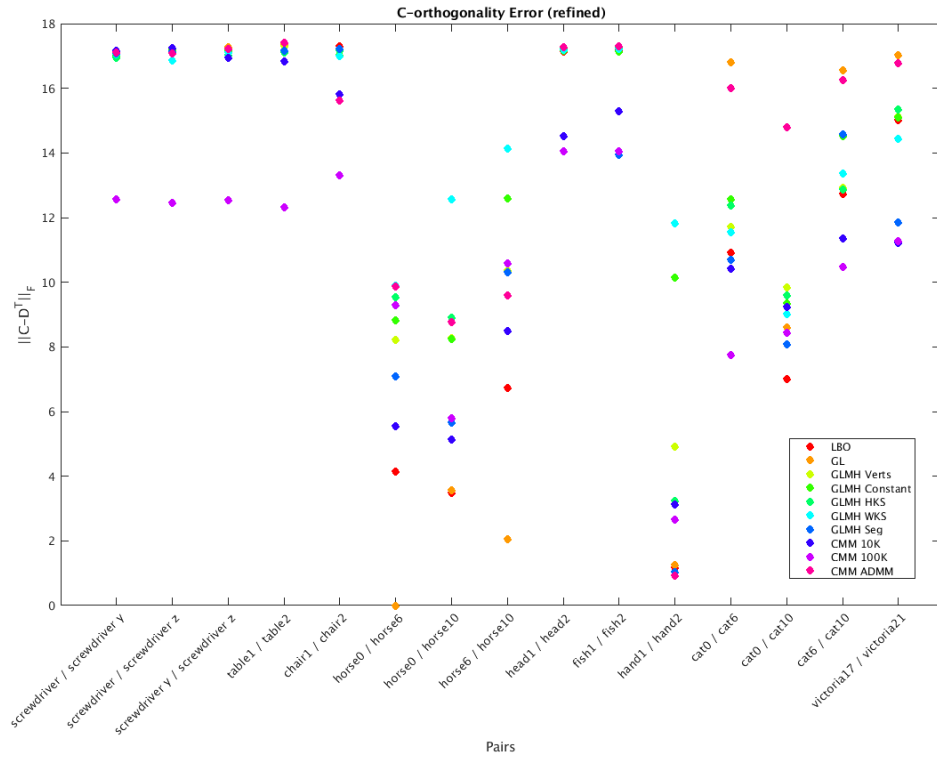


Figure 5.9: *C*-orthogonality error (refined)

unrefined C matrix, the unfilled markers denote error after refinement.

Figure 5.11 compares the geodesic functional map error for maps calculated in both directions, allowing a closer look at the performance of each basis type. Solid markers denote the refined C matrices and unfilled markers denote the refined D matrices calculated in the opposite direction. Once again there is no clear basis type which performs best across all pairs, but micro-trends appear for the groups of meshes – consider the performance of the GL basis for both the screwdriver meshes and the horse meshes, noting that it performs best for one but not the other.

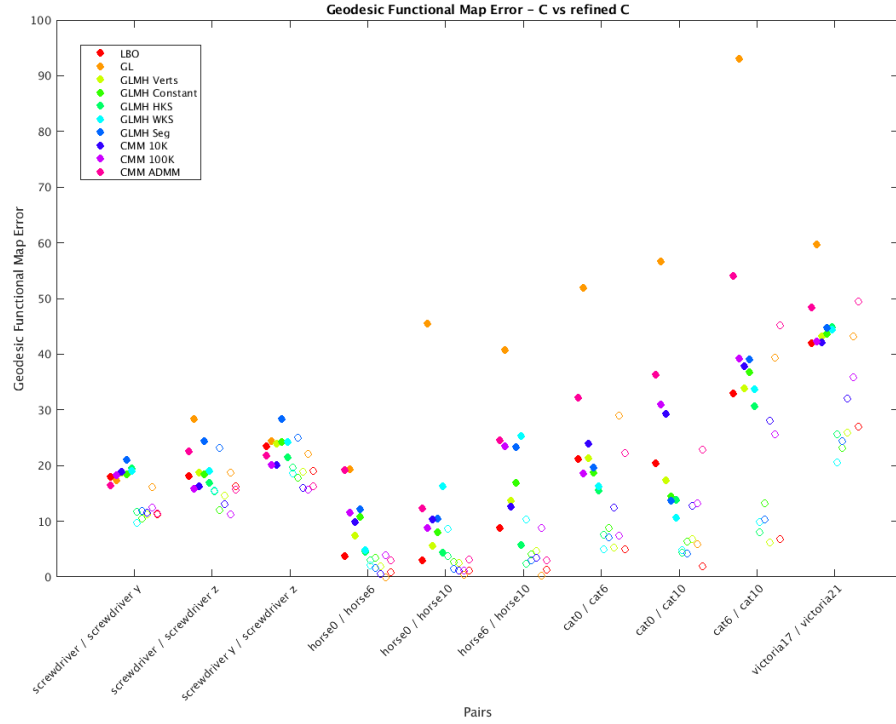


Figure 5.10: Geodesic functional map error – the effect of refinement.

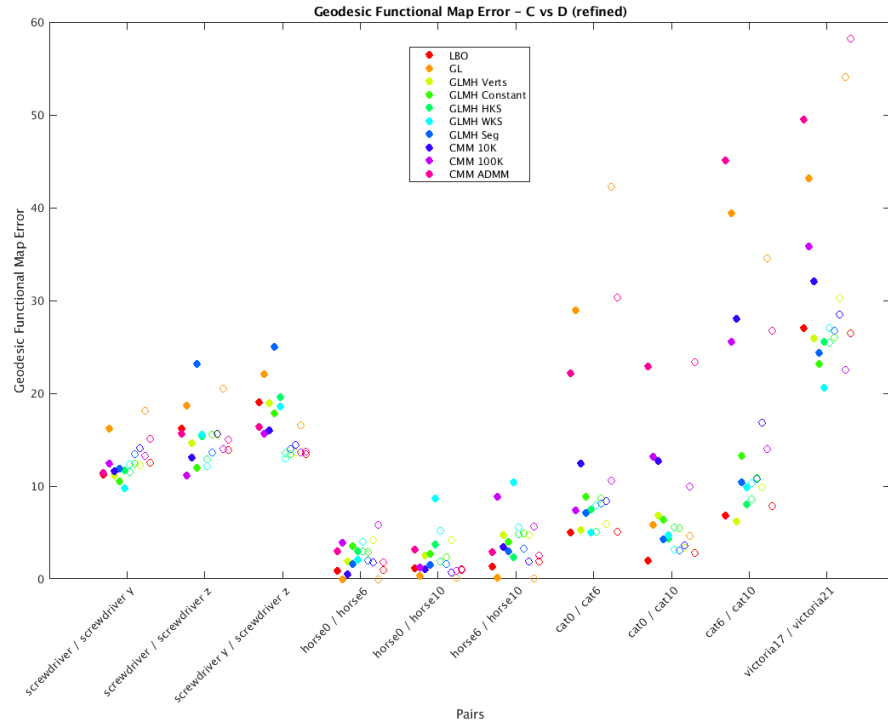


Figure 5.11: Geodesic functional map error – comparing refined functional map direction.

5.4 Summary and Future Work

Functional maps were calculated using a variety of bases. The ability of the functional maps to reconstruct the functions used in their construction was tested.

The key observations are as follows:

- Refinement negatively affects transformation error. This is because the least squares solution minimises transformation error. As a consequence any refinement leads to an increase in transformation error. As refinement is often used to improve geodesic error the result is that functions mapped using a refined functional map matrix will have an increased transformation error.
- There is no basis type which consistently reduces error across the whole mesh set.
- Some basis types may be better at reducing errors than others for specific shape collections. However, this comes with a warning that the experiments were unfairly skewed by meshes which have the same underlying graph structure.

In theorem 5.3.1 it was proved that the functions used in the construction of the functional map should transform with least error. There is more scope for investigation here as only the transformation and reconstruction of these functions was tested. How do the functional maps perform when transforming other functions?

Chapter 6

Optimisation on Generalised Stiefel Manifolds

The Stiefel manifold is defined via the set of $n \times k$ matrices X such that $X^T X = I$, [117, example 3.5.2]. In the $A = I$ case, the compressed manifold mode problem can be considered as an optimisation problem on a Stiefel manifold. This chapter gives an introduction to generalised Stiefel manifolds, defined via the set of $n \times k$ matrices X such that $X^T M X = I$, where M is symmetric positive definite. Known properties and results for the Stiefel manifold are generalised for the $M \neq I$ case.

Optimisation problems on the Stiefel manifold have been studied elsewhere. The Stiefel manifold is a frequent example in [117]; optimisation algorithms are given in [118]; accelerated algorithms are given in [119] and [120]. Optimisation requires a retraction from the tangent space to the manifold. The recent works of [121],[122] use a retraction based on the Cholesky decomposition. A proof is provided that this is a retraction in the general case, noting the deficiency in the reference provided in both [121] and [122].

Existing methods for optimisation on Stiefel manifolds also require the objective functions to be smooth [118, p.330] or differentiable [119, section 3.3]. Here a sequential method for optimisation on generalised Stiefel manifolds is presented. It is applied to the CMM problem, approximating via a smoothing of the ℓ_1 norm.

Section 6.5 details some possible approximations (smoothings) of the ℓ_1 norm to be used in the optimisation.

6.1 Generalised Stiefel Manifolds

This section follows a technical note [123], written to aid understanding of a notable work about optimisation on Stiefel manifolds [120]. The theory is adapted for the generalisation of Stiefel manifolds, altering the orthogonality constraints. It is assumed throughout that M is an $n \times n$ symmetric positive definite matrix. (Note that this means M^{-1} exists.) In addition, the following notation is used for symmetric and skew-symmetric matrices:

Definition 6.1.1: Let $\text{Sym}(n)$ denote the set of $n \times n$ symmetric matrices,

$$\text{Sym}(n) = \{W \in \mathbb{R}^{n \times n} : W = W^T\}. \quad (6.1.1)$$

Let $Y \in \mathbb{R}^{n \times n}$. Define the projection $\text{sym} : \mathbb{R}^{n \times n} \rightarrow \text{Sym}(n)$ to be

$$\text{sym}(Y) = \frac{1}{2}(Y + Y^T). \quad (6.1.2)$$

Definition 6.1.2: Let $\text{Skew}(n)$ denote the set of $n \times n$ skew-symmetric matrices,

$$\text{Skew}(n) = \{W \in \mathbb{R}^{n \times n} : W = -W^T\}. \quad (6.1.3)$$

Let $Y \in \mathbb{R}^{n \times n}$. Define the projection $\text{skew} : \mathbb{R}^{n \times n} \rightarrow \text{Skew}(n)$ to be

$$\text{skew}(Y) = \frac{1}{2}(Y - Y^T). \quad (6.1.4)$$

Remark 6.1.3: $\text{Sym}(n)$ is an $\frac{1}{2}n(n+1)$ dimensional vector space; $\text{Skew}(n)$ is an $\frac{1}{2}n(n-1)$ dimensional vector space.

Definition 6.1.4: The **generalised Stiefel manifold** is the set

$$\mathcal{S}_M := \{X \in \mathbb{R}^{n \times k} : X^T M X = I\}, \quad (6.1.5)$$

where M is an $n \times n$ symmetric, positive-definite matrix.

That this is a manifold of dimension $nk - \frac{1}{2}k(k+1)$ is a simple generalisation of [117, pp.26-27].

The notation \mathcal{S} without a subscript matrix M is used to denote the Stiefel manifold where $M = I$.

Denote by $T_X\mathcal{S}_M$ the tangent space at X to \mathcal{S}_M . The tangent space can be described by a specific relationship between tangent vectors and the basepoint X .

Lemma 6.1.5: [123, lemma 1, generalised] Any $Z \in T_X\mathcal{S}_M$ satisfies $Z^T MX + X^T MZ = 0$.

Proof. Let $Y : \mathbb{R} \rightarrow \mathcal{S}_M$ defined by $Y(t)$ be a differentiable curve in \mathcal{S}_M . Then $Y(t)^T MY(t) = I_k$ for all $t \in \mathbb{R}$. Differentiating with respect to t gives

$$Y'(t)^T MY(t) + Y(t)^T MY'(t) = 0.$$

Setting $Y(0) = X$ and $Y'(0) = Z$ gives $Z^T MX + X^T MZ = 0$. □

Note that this means that the matrix $Z^T MX$ is skew-symmetric as

$$Z^T MX = -X^T MZ = -(Z^T MX)^T.$$

Proposition 6.1.6: Let $X \in \mathcal{S}_M$ and $Z \in T_X\mathcal{S}_M$. Then

- (i) $(X + Z)^T M(X + Z)$ is a positive definite matrix,
- (ii) The tangent space $T_X\mathcal{S}_M$ intersects \mathcal{S}_M only at X .

Proof. (i) Firstly,

$$(X + Z)^T M(X + Z) = X^T MX + Z^T MX + X^T MZ + Z^T MZ.$$

Recall that $Z \in T_X\mathcal{S}_M$ means that $Z^T MX + X^T MZ = 0$ and that for any $X \in \mathcal{S}_M$, $X^T MX = I$. So

$$(X + Z)^T M(X + Z) = I + Z^T MZ. \tag{6.1.6}$$

Since M is positive definite, the matrix $Z^T M Z$ is positive semi-definite, therefore by equation (6.1.6)

$$\begin{aligned} v^T (X + Z)^T M (X + Z) v &= v^T v + v^T Z^T M Z v \\ &\geq 0 \end{aligned}$$

for all $v \in \mathbb{R}^k$, with equality if and only if $v = 0$. Hence $(X + Z)^T M (X + Z)$ is positive definite.

- (ii) Assume towards a contradiction that $\exists Z \neq 0$ with $X + Z \in \mathcal{S}_M$. The matrix $Z^T M Z$ has (ij) -th entry given by $z_i^T M z_j$, where z_i denotes the i -th column of Z . Since $Z \neq 0$ at least one of the columns z_i is non-zero. Then, at least one of the entries $z_i^T M z_i$ is non-zero as M is positive definite. Therefore, $Z^T M Z \neq 0$, and $(X + Z)^T M (X + Z) \neq I$, a contradiction. Hence, $T_X \mathcal{S}_M \cap \mathcal{S}_M = \{X\}$.

□

To give an alternative description of $T_X \mathcal{S}_M$ proposition 2.3 of [121] is generalised.

Proposition 6.1.7: [121, proposition 2.3, generalised] Let $X \in \mathcal{S}_M$, then

$$(i) \quad T_X \mathcal{S}_M = \{Z \in \mathbb{R}^{n \times k} : Z = W M X, W \in \text{Skew}(n)\},$$

$$(ii) \quad T_X \mathcal{S}_M = \{Z \in \mathbb{R}^{n \times k} : Z = M^{-1} W X, W \in \text{Skew}(n)\}.$$

Proof. Let $Z \in T_X \mathcal{S}_M$. Then define the projection

$$P_X := I - \frac{1}{2} X X^T M. \tag{6.1.7}$$

Define also the skew-symmetric matrix

$$W_Z := P_X Z X^T - X Z^T P_X^T. \tag{6.1.8}$$

Then

$$\begin{aligned}
W_Z MX &= P_X Z X^T M X - X Z^T P_X^T M X \\
&= P_X Z - X Z^T P_X^T M X \\
&= Z - \frac{1}{2} X X^T M Z - X Z^T M X + \frac{1}{2} X Z^T M X \\
&= Z - \frac{1}{2} X X^T M Z - \frac{1}{2} X Z^T M X \\
&= Z - \frac{1}{2} X (X^T M Z + Z^T M X) \\
&= Z, \text{ since } X^T M Z + Z^T M X = 0 \text{ as } Z \in T_X \mathcal{S}_M. \tag{6.1.9}
\end{aligned}$$

Hence $Z = W_Z M X$. Therefore $T_X \mathcal{S}_M \subseteq \{Z \in \mathbb{R}^{n \times k} : Z = W M X, W \in \text{Skew}(n)\}$.

Conversely, let Z be such that $Z = W M X$ for some $W \in \text{Skew}(n)$. Then

$$\begin{aligned}
Z^T M X + X^T M Z &= (W M X)^T M X + X^T M (W M X) \\
&= X^T M W^T M X + X^T M W M X \\
&= X^T M W^T M X - X^T M W^T M X, \text{ since } W = -W^T, \\
&= 0.
\end{aligned}$$

So $Z \in T_X \mathcal{S}_M$. Therefore $\{Z \in \mathbb{R}^{n \times k} : Z = W M X, W \in \text{Skew}(n)\} \subseteq T_M \mathcal{S}_M$ and hence $T_X \mathcal{S}_M = \{Z \in \mathbb{R}^{n \times k} : Z = W M X, W \in \text{Skew}(n)\}$.

The proof that $T_X \mathcal{S}_M = \{Z \in \mathbb{R}^{n \times k} : Z = M^{-1} W X, W \in \text{Skew}(n)\}$ is similar. \square

The matrices P_X and W_Z will appear again later.

Given $X \in \mathcal{S}_M$, the columns of X are M -orthonormal vectors in \mathbb{R}^n . Therefore, an additional set of $(n - k)$ vectors in \mathbb{R}^n can be constructed to be M -orthogonal to the columns of X . Let X_\perp denote the matrix with the additional vectors as columns. Note that the combined set of vectors forms an orthonormal basis for \mathbb{R}^n . Let $\begin{bmatrix} X & X_\perp \end{bmatrix}$ denote the $n \times n$ matrix formed by concatenating X and X_\perp .

Lemma 6.1.8: [123, lemma 2] The matrix $\begin{bmatrix} X & X_\perp \end{bmatrix}$ is an isomorphism of $\mathbb{R}^{n \times k}$.

Proof. (Tagare) Let $W = \begin{bmatrix} X & X_\perp \end{bmatrix}$. Since the columns of W form a basis for \mathbb{R}^n , W is invertible. Similarly $W^{-1}R \in \mathbb{R}^{n \times k} \quad \forall R$. Consider $P \in \mathbb{R}^{n \times p}$, $P \neq R$. Assume $WR = WP$. Multiplication on the left by W^{-1} gives $R = P$, a contradiction, therefore multiplication by W is injective. Surjectivity is clear as when $WR \neq WP$ then $R \neq P$. As matrix multiplication is a homomorphism the action of W is an automorphism. \square

Any element $U \in \mathbb{R}^{n \times k}$ can be written as $U = \begin{bmatrix} X & X_\perp \end{bmatrix} C$ where C is an $n \times k$ matrix. Let

$$C = \begin{bmatrix} A \\ B \end{bmatrix},$$

where $A \in \mathbb{R}^{k \times k}$ and $B \in \mathbb{R}^{(n-k) \times k}$. Then

$$U = \begin{bmatrix} X & X_\perp \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} = XA + X_\perp B. \quad (6.1.10)$$

Lemma 6.1.9: [123, lemma 3, generalised] A matrix $Z = XA + X_\perp B$ is in $T_X \mathcal{S}_M$ if and only if A is skew-symmetric.

Proof. Let $\Omega = \{Z \in \mathbb{R}^{n \times k} : Z = XA + X_\perp B, \quad A = -A^T\} \subseteq \mathbb{R}^{n \times k}$. The dimension of Ω is determined by the matrices A and B , hence $\dim(\Omega) = \frac{1}{2}k(k-1) + (n-k)k = nk - \frac{1}{2}k(k+1)$. Let $Z \in T_X \mathcal{S}_M$. Since $Z \in T_X \mathcal{S}_M \subseteq \mathbb{R}^{n \times k}$, Z can be expressed in the form $Z = XA + X_\perp B$ and satisfies the condition that $Z^T M X + X^T M Z = 0$. Then

$$\begin{aligned} 0 &= (A^T X^T + B^T X_\perp^T) M X + X^T M (XA + X_\perp B) \\ &= A^T X^T M X + B^T X_\perp^T M X + X^T M X A + X^T M X_\perp B \\ &= A^T + A \text{ since } X^T M X = I \text{ and } X^T M X_\perp = 0. \end{aligned}$$

Therefore A is skew-symmetric and $T_X \mathcal{S}_M \subseteq \Omega$. To show equality consider the dimension of $T_X \mathcal{S}_M$:

$$\dim(T_X \mathcal{S}_M) = \dim(\mathcal{S}_M) = nk - \frac{1}{2}k(k+1) = \dim(\Omega),$$

therefore $T_X \mathcal{S}_M = \Omega$. \square

Definition 6.1.10: Let $Z_1, Z_2 \in T_X \mathcal{S}_M$. Then define the **Euclidean inner product with respect to M** as

$$\langle Z_1, Z_2 \rangle_e = \text{tr}(Z_1^T M Z_2). \quad (6.1.11)$$

This is just the M -inner product, as defined in definition 2.1.5, but here the terminology of [118] and [120] is used to remain consistent with those works.

Consider $\langle Z, Z \rangle_e$ when $Z = XA + X_\perp B$:

$$\begin{aligned} \langle Z, Z \rangle_e &= \text{tr}(Z^T M Z) \\ &= \text{tr}((A^T X^T + B^T X_\perp^T) M (XA + X_\perp B)) \\ &= \text{tr}(A^T X^T M X A + A^T X^T M X_\perp B + B^T X_\perp^T M X A + B^T X_\perp^T M X_\perp B). \end{aligned}$$

Recall that $X^T M X = I$, $X^T M X_\perp = 0$, $X_\perp^T M X = 0$, $X_\perp^T M X_\perp = I$ and hence,

$$\begin{aligned} \langle Z, Z \rangle_e &= \text{tr}(A^T A + B^T B) \\ &= \sum_{i>j} 2A_{ij}^2 + \sum_{ij} B_{ij}^2. \end{aligned}$$

The Euclidean inner product gives twice the weight to independent entries of A , so consider a new inner product which weights coordinates equally:

Definition 6.1.11: Let $Z_1, Z_2 \in T_X \mathcal{S}_M$. The **(generalised) canonical inner product** is defined to be

$$\langle Z_1, Z_2 \rangle_c = \text{tr} \left(Z_1^T M \left(I - \frac{1}{2} X X^T M \right) Z_2 \right). \quad (6.1.12)$$

Check that this does indeed produce an equal weight for all coordinates: Let $Z =$

$XA + X_{\perp}B$ then

$$\begin{aligned}
\langle Z, Z \rangle_c &= \text{tr} \left((A^T X^T + B^T X_{\perp}^T) M \left(I_n - \frac{1}{2} X X^T M \right) (XA + X_{\perp}B) \right) \\
&= \text{tr} \left((A^T X^T M + B^T X_{\perp}^T M - \frac{1}{2} A^T X^T M) (XA + X_{\perp}B) \right) \\
&= \text{tr} \left(A^T A - \frac{1}{2} A^T A + B^T B \right) \\
&= \text{tr} \left(\frac{1}{2} A^T A + B^T B \right) \\
&= \sum_{i < j} A_{ij}^2 + \sum_{ij} B_{ij}^2.
\end{aligned}$$

Definition 6.1.12: [119, 3.1.1], [117, 4.1.1] Let \mathcal{M} be a smooth manifold. A **retraction** on \mathcal{M} is a smooth map R from the tangent bundle $T\mathcal{M}$ to \mathcal{M} , such that for all $X \in \mathcal{M}$, $Z \in T_X\mathcal{M}$ the restriction $R_X : T_X\mathcal{M} \rightarrow \mathcal{M}$ satisfies

(i) $R_X(0) = X$,

(ii) $\left. \frac{d}{d\tau} \right|_{\tau=0} R_X(\tau Z) = Z$, where $\tau \in \mathbb{R}$.

Definition 6.1.13: Let $X \in \mathbb{R}^{n \times k}$ and let M be an $n \times n$ symmetric positive definite matrix. The **Cayley Transform** is the map $\text{CT}_X : \text{Skew}(n) \rightarrow \mathbb{R}^{n \times k}$ defined by

$$\text{CT}_X(W) = (I - WM)^{-1}(I + WM)X, \quad (6.1.13)$$

where $W \in \text{Skew}(n)$.

Lemma 6.1.14: If $X \in \mathcal{S}_M$ and $W \in \text{Skew}(n)$ then $\text{CT}_X(W) \in \mathcal{S}_M$.

Proof. Let $W \in \text{Skew}(n)$. Firstly note that

$$(I + WM)^T = I - MW \quad \text{and} \quad (I - WM)^T = I + MW.$$

Secondly note that

$$\begin{aligned}
(I - WM)M^{-1}(I + MW) &= (M^{-1} - W)(I + MW) \\
&= M^{-1} - W + W - WMW \\
&= M^{-1} - WMW \\
&= (M^{-1} + W)(I - MW) \\
&= (I + WM)M^{-1}(I - MW). \tag{6.1.14}
\end{aligned}$$

Let $Q = (I - WM)^{-1}(I + WM)$. Then

$$\begin{aligned}
Q^T M Q &= (I + WM)^T (I - WM)^{-T} M (I - WM)^{-1} (I + WM) \\
&= (I - MW)(I + MW)^{-1} M (I - WM)^{-1} (I + WM) \\
&= (I - MW) \left((I - WM)M^{-1}(I + MW) \right)^{-1} (I + WM) \\
&= (I - MW) \left((I + WM)M^{-1}(I - MW) \right)^{-1} (I + WM), \text{ from (6.1.14)} \\
&= (I - MW)(I - MW)^{-1} M (I + WM)^{-1} (I + WM) \\
&= IMI \\
&= M.
\end{aligned}$$

Then, $\text{CT}_X(W) = QX$ so

$$\begin{aligned}
\text{CT}_X(W)^T M \text{CT}_X(W) &= (QX)^T M (QX) \\
&= X^T Q^T M Q X \\
&= X^T M X, \text{ since } Q^T M Q = M \\
&= I, \text{ since } X \in \mathcal{S}_M.
\end{aligned}$$

That is, $\text{CT}_X(W) \in \mathcal{S}_M$. □

Proposition 6.1.15: Let $X \in \mathcal{S}_M$ and let P_X, W_Z be as in equations (6.1.7),(6.1.8).

Then the map $\text{TCT}_X : T_X \mathcal{S}_M \rightarrow \mathcal{S}_M$ defined by

$$\text{TCT}_X(Z) = \text{CT}_X \left(\frac{1}{2} W_Z \right) = \text{CT}_X \left(\frac{1}{2} (P_X Z X^T - X Z^T P_X^T) \right), \tag{6.1.15}$$

is a retraction.

Proof. To show this check that TCT_X satisfies the conditions of the definition:

(i) $\text{TCT}_X(0) = \text{CT}_X(0) = I^{-1}IX = X.$

(ii) Let $W := \frac{1}{2}W_Z = \frac{1}{2}(P_X Z X^T - X Z^T P_X^T)$, then

$$\frac{d}{d\tau}\Big|_{\tau=0} \text{TCT}_X(\tau Z) = \frac{d}{d\tau}\Big|_{\tau=0} \text{CT}_X(\tau W).$$

Since $\frac{\partial}{\partial X} Y^{-1} = -Y^{-1} \cdot \frac{\partial}{\partial \tau} Y \cdot Y^{-1}$ [19, (59)] and $\frac{\partial}{\partial X}(I \pm \tau WM) = \pm WM$,

$$\begin{aligned} \frac{d}{d\tau}\Big|_{\tau=0} \text{TCT}_X(\tau Z) &= \left[-(I - \tau WM)^{-1}(-WM)(I - \tau WM)^{-1}(I + \tau WM)X \right. \\ &\quad \left. + (I - \tau WM)^{-1}(WM)X \right]_{\tau=0} \\ &= -I^{-1}(-WM)I^{-1}IX + I^{-1}WMX \\ &= 2WMX \\ &= W_Z M X \\ &= Z, \text{ via lemma 6.1.7, equation (6.1.9).} \end{aligned}$$

And, therefore, TCT_X is a retraction. □

Definition 6.1.16: The map TCT_X is called the **Cayley retraction**.

Use of the Cayley retraction can be computationally expensive due to the inversion of the $n \times n$ matrix $I + \frac{\tau}{2}W_Z$. This can be simplified by the following theorem:

Theorem 6.1.17: [19, (157)] (The **Woodbury matrix identity**) Let A be an $n \times n$ invertible matrix, let U be a $n \times k$ matrix and let V be a $k \times n$ matrix. Then,

$$(A + UV)^{-1} = A^{-1} - A^{-1}U(I_k + VA^{-1}U)^{-1}VA^{-1}. \quad (6.1.16)$$

A proof can be found in appendix G.

Then via the Woodbury matrix identity

$$(I - \frac{\tau}{2}W_Z M)^{-1} = I - U(I + VU)^{-1}V, \quad (6.1.17)$$

with

$$U = -\frac{\tau}{2} \begin{bmatrix} P_X Z & X \end{bmatrix} \quad \text{and} \quad V = \begin{bmatrix} X^T M \\ -Z^T P_X^T M \end{bmatrix}. \quad (6.1.18)$$

So

$$\begin{aligned} \text{CT}_X\left(\frac{\tau}{2}W_Z\right) &= (I - U(I + VU)^{-1}V)(I - UV)X \\ &= X - UVX - U(I + VU)^{-1}VX + U(I + VU)^{-1}VUVX \\ &= X - U(I + VU)^{-1}[(I + VU) + I - VU]V \\ &= X - 2U(I + VU)^{-1}VX. \end{aligned} \quad (6.1.19)$$

This is simpler to compute than equation (6.1.13) since $I + VU$ is a $2k \times 2k$ matrix and k is chosen to be much smaller than n in our applications.

Another retraction which has been used in some recent work ([121],[122]) is a retraction based on the Cholesky decomposition of a symmetric positive definite matrix. These papers claim that there exists a retraction to \mathcal{S} based on the QR decomposition with an argument to generalise to \mathcal{S}_M via the Cholesky decomposition. (For details on matrix decomposition see section 2.2.) Both provide citation to Absil [117] which does not provide a proof that the QR decomposition can be used to define a retraction onto \mathcal{S} . Absil proves that there is a retraction based on the QR decomposition to the orthogonal group and claims a retraction on to \mathcal{S} . The proof below uses an alternative approach.

First let

$$\mathbb{P} := \{X \in \mathbb{R}^{n \times n} : X \text{ symmetric positive definite}\}$$

and let

$$\mathbb{L} := \{X \in \mathbb{R}^{n \times n} : X \text{ lower triangular with positive diagonal elements}\}.$$

Proposition 6.1.18: The map $R_X^{\text{Chol}} : T_X \mathcal{S}_M \rightarrow \mathcal{S}_M$ defined by

$$R_X^{\text{Chol}}(Z) = (X + Z)C^{-1}, \quad (6.1.20)$$

where $C = \text{Chol}((X + Z)^T M (X + Z))^T$, is a retraction.

Proof. First, check that $R_X^{\text{Chol}}(Z) \in \mathcal{S}_M$:

$$\begin{aligned} R_X^{\text{Chol}}(Z)^T M R_X^{\text{Chol}}(Z) &= (C^{-1})^T (X + Z)^T M (X + Z) C^{-1} \\ &= (C^{-1})^T C^T C C^{-1}, \text{ since } C^T C = (X + Z)^T M (X + Z), \\ &= I. \end{aligned}$$

Hence $R_X^{\text{Chol}}(Z) \in \mathcal{S}_M$. Next check the properties of a retraction:

- (i) $R_X^{\text{Chol}}(0) = X C^{-1}$, where $C^{-1} = \text{Chol}(X^T M X)$. Since $X \in \mathcal{S}_M$, $X^T M X = I$. The Cholesky decomposition of I is given by I and hence $R_X^{\text{Chol}}(0) = X$.
- (ii) Let $F : \mathbb{R} \rightarrow \mathbb{P}$ be defined by $F(\tau) = (X + \tau Z)^T M (X + \tau Z)$ which is equal to $I + \tau^2 Z^T M Z$ since $Z \in T_X \mathcal{S}_M$. The $(X + \tau Z)^T M (X + \tau Z)$ is symmetric positive definite by 6.1.6. Then $F'(\tau) = 2\tau Z^T M Z$, $F''(\tau) = 2Z^T M Z$ and $F^{(n)}(\tau) = 0$ for all $n \geq 3$. The matrix $F(\tau)$ has entries $F(\tau)_{ij} = \delta_{ij} + \tau^2 (Z^T M Z)_{ij}$, a polynomial of τ .

Let $H : \mathbb{R} \rightarrow \mathbb{L}$ be defined by $H(\tau) = \text{Chol}(F(\tau))$. That is, $F(\tau) = H(\tau)^T H(\tau)$. The decomposition $\text{Chol}(F(\tau))$ is defined element-wise on the matrix $F(\tau)$, via the functions in 2.2.7 (and according to the forced order of construction used in algorithm 1). Given that elements of $F(\tau)$ are polynomials in τ the functions constructing the elements L_{ij} are analytic since matrix derivatives are also taken element-wise H can be expressed using the Taylor series expansion:

$$H(\tau) = H(0) + H'(0) \tau + \frac{H''(\tau)}{2!} \tau^2 + \frac{H^{(3)}(\tau)}{3!} \tau^3 + \dots$$

Then equate τ coefficients of $F(\tau)$ and $H(\tau)^T H(\tau)$:

$$\begin{aligned} F(\tau) &= (H(0)^T + H'(0)^T \tau + \dots)(H(0)^T + H'(0)^T \tau + \dots) \\ I + \tau^2 Z^T M Z &= H(0)^T H(0) + H(0)^T H'(0)^T \tau + H'(0)^T H(0)^T \tau \\ &\quad + \text{higher order } \tau \text{ terms,} \end{aligned}$$

giving $I = H(0)^T H(0)$ and $0 = H(0)^T H'(0)^T + H'(0)^T H(0)$. Since $H(0)$ is upper triangular it must be that $H(0) = I$, and so it follows that $H'(0) = 0$.

Therefore

$$H(\tau) = I + \frac{H''(0)}{2!}\tau^2 + \frac{H^{(3)}(0)}{3!}\tau^3 + \dots$$

Let $\bar{H}(\tau) := H(\tau)^{-1}$, with Taylor expansion

$$\bar{H}(\tau) = \bar{H}(0) + \bar{H}'(0)\tau + \frac{\bar{H}''(0)}{2!}\tau^2 + \frac{\bar{H}^{(3)}(0)}{3!}\tau^3 + \dots$$

Then, since $I = H(\tau)H(\tau)^{-1} = H(\tau)\bar{H}(\tau)$,

$$\begin{aligned} I &= \left(I + \frac{H''(0)}{2!}\tau^2 + \frac{H^{(3)}(0)}{3!}\tau^3 + \dots \right) \left(\bar{H}(0) + \bar{H}'(0)\tau + \frac{\bar{H}''(0)}{2!}\tau^2 + \dots \right) \\ &= \bar{H}(0) + \bar{H}'(0)\tau + \frac{\bar{H}''(0)}{2!}\tau^2 + \frac{H''(0)\bar{H}(0)}{2!}\tau^2 + \text{higher order } \tau \text{ terms.} \end{aligned}$$

By comparing coefficients, $\bar{H}(0) = I$ and $\bar{H}'(0) = 0$.

Now the second property of a retraction is simple to check:

$$\begin{aligned} \left. \frac{d}{d\tau} \right|_{\tau=0} R_X^{\text{Chol}}(\tau Z) &= \left. \frac{d}{d\tau} \right|_{\tau=0} (X + \tau Z)\bar{H}(\tau) \\ &= Z\bar{H}(\tau)|_{\tau=0} + (X + \tau Z) \left[\left. \frac{d}{d\tau} \bar{H}(\tau) \right]_{\tau=0} \right. \\ &= Z\bar{H}(0) + (X + \tau Z)\bar{H}'(0) \\ &= Z. \end{aligned}$$

Therefore R_X^{Chol} is a retraction onto \mathcal{S}_M . □

Definition 6.1.19: The map R_X^{Chol} is called the **Cholesky retraction**.

6.2 Background: Optimisation on Manifolds

Let $\mathcal{M} \subseteq \mathbb{R}^n$ be a manifold and let $F : \mathbb{R}^n \rightarrow \mathbb{R}$ be a real-valued function. Consider the constrained optimisation problem

$$\min_X F(X) \text{ subject to } X \in \mathcal{M}. \quad (6.2.21)$$

Let the gradient of F at X be denoted by

$$\nabla F(X) = G := \frac{\partial F(X)}{\partial X} = \left[\frac{\partial F(X)}{\partial X_i} \right].$$

However, most usually it is the gradient of the restricted function $F|_{\mathcal{M}} : \mathcal{M} \rightarrow \mathbb{R}$ which is required.

Definition 6.2.1: Let $(\mathcal{M}, g) \subseteq \mathbb{R}^{n \times k}$ be a Riemannian manifold and let $F : \mathcal{M} \rightarrow \mathbb{R}$ be a smooth real-valued function. Let $Z \in T_X \mathcal{M}$. The **directional derivative of F at X in direction Z** is defined to be

$$\mathcal{D}_Z F(X) := \langle G, Z \rangle_g \quad [120, 1.4]. \quad (6.2.22)$$

Definition 6.2.2: [117, 3.31] Let $(\mathcal{N}, h) \subseteq (\mathcal{M}, g) \subseteq \mathbb{R}^{n \times k}$ and let $F|_{\mathcal{N}} : \mathcal{N} \rightarrow \mathbb{R}$ be the function F restricted to \mathcal{N} . Let $X \in \mathcal{N}$. Then the **restricted gradient** $\nabla F|_{\mathcal{N}}(X)$ is the unique element of $T_X \mathcal{N}$ satisfying, for all $Z \in T_X \mathcal{N} \subseteq T_X \mathcal{M}$,

$$\langle \nabla F|_{\mathcal{N}}(X), Z \rangle_h = \mathcal{D}_Z F(X) = \langle G, Z \rangle_g. \quad (6.2.23)$$

When $\mathcal{N} \subseteq \mathcal{M}$ is a Riemannian submanifold, with the induced metric, the following lemma can be used:

Lemma 6.2.3: [117, p.48 (3.37)] Let $f : \mathcal{M} \rightarrow \mathbb{R}$ be a real-valued function on Riemannian manifold (\mathcal{M}, g) . Consider a Riemannian submanifold $\mathcal{N} \subseteq \mathcal{M}$, with induced metric h . Let $F|_{\mathcal{N}} : \mathcal{N} \rightarrow \mathbb{R}$ be the restriction of F to \mathcal{N} and let $X \in \mathcal{N}$. Then the gradient $\nabla F|_{\mathcal{N}}(X)$ is given by the projection of the gradient $\nabla F(X)$ into the tangent space $T_X \mathcal{N}$. That is, for projection $\text{proj}_{T_X \mathcal{N}} : T_X \mathcal{M} \rightarrow T_X \mathcal{N}$,

$$\nabla F|_{\mathcal{N}}(X) = \text{proj}_{T_X \mathcal{N}}(\nabla F(X)). \quad (6.2.24)$$

To minimise a function $F|_{\mathcal{M}}$ consider a gradient descent algorithm. Gradient descent algorithms iteratively define new, improved points from old via a descent direction and a step size.

Definition 6.2.4: Let \mathcal{M} be a manifold and let $F : \mathcal{M} \rightarrow \mathbb{R}$ be a differentiable real valued function. A **gradient descent algorithm** is an algorithm which iteratively

searches for a minimiser of F . Given a retraction $R : T\mathcal{M} \rightarrow \mathcal{M}$ and an initial start point $X_0 \in \mathcal{M}$, new points X_{p+1} are defined iteratively by

$$X_{p+1} := R_{X_p}(\tau_p d_p). \quad (6.2.25)$$

The scalar τ_p is called the **step size**. The **descent direction** d_p is an element of $T_{X_p}\mathcal{M}$ such that $\langle d_p, G \rangle < 0$ where $\langle \cdot, \cdot \rangle$ denotes the inner product on $T_{X_p}\mathcal{M}$ and G is the gradient of F .

Figure 6.1 illustrates this, with the vector lying in the tangent space representing the descent direction, and the red arrow representing the retraction from the tangent space back down to the manifold.

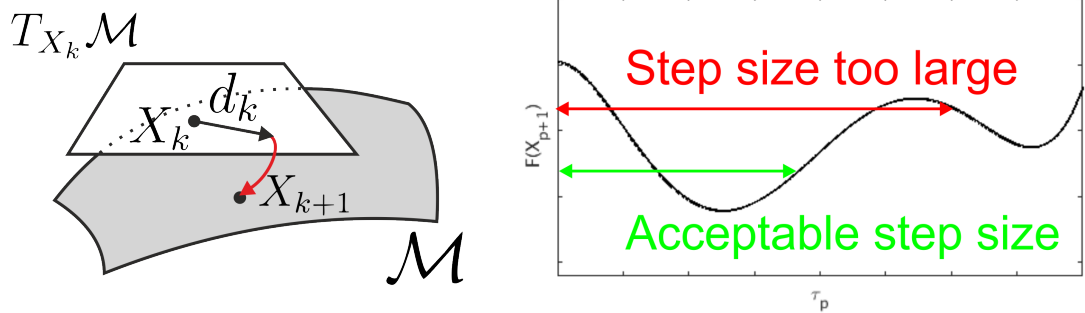


Figure 6.1: Gradient descent on a manifold, via retraction.

Figure 6.2: The importance of step size choice.

The gradient G gives a direction of greatest change in the value of F , based at X . Movement in the direction of $-G$ is, therefore, an efficient method of searching for a minimum. This is the direction of *steepest* descent. A gradient descent algorithm for minimisation of a real-valued function evaluated on a manifold is given in algorithm 4.

Of course, care must be taken when choosing how far to move in the descent direction. A step size which is too small will result in a slow decrease of the objective function but too large a step size may pass over a local minimum. Step size choice is illustrated in figure 6.2, which plots $F(X_{p+1})$ as step-size T_p varies, given fixed descent direction. This restriction to a search in a fixed direction is known as a *line search*.

Algorithm 4 Gradient descent on a manifold \mathcal{M}

- 1: Given a function $F : \mathcal{M} \rightarrow \mathbb{R}$ and retraction R onto \mathcal{M} ,
 - 2: Set convergence tolerance ε .
 - 3: Generate random $x_0 \in \mathcal{M}$.
 - 4: Set $k = 0$.
 - 5: **while** $\|\nabla F(x)\| > \varepsilon$ **do**
 - 6: Select descent direction d .
 - 7: Select step size τ .
 - 8: Move to new iterate $x_{k+1} = R_x(\tau d)$.
 - 9: $k \leftarrow k + 1$
 - 10: **end while**
-

Definition 6.2.5: Let $f : \mathcal{M} \rightarrow \mathbb{R}$ be a real-valued function. Let $x_p \in \mathcal{M}$, $d \in T_{x_p}\mathcal{M}$, $\tau \in \mathbb{R}$. A **line search** is an algorithm which searches for a scalar τ such that the value of the function f satisfies some condition for the new iterate x_{p+1} . For a **monotone line search** this condition is that the value of f decreases at x_{p+1} .

In general, solving for an optimum τ is expensive, and so instead end the search when a step size which meets certain conditions is found. The line search algorithm below uses conditions on the gradient of f at x_{p+1} and the decrease in the value of f between the old and new points.

To ensure that the decrease in the value of the function is sufficient, ensure that τ is such that

$$f(x_{p+1}) \leq f(x_p) + c_1\tau\langle\nabla f(x_p), d\rangle,$$

where $c_1 \in (0, 1)$. That is, the decrease of f is proportional to the step size and the rate of change of f at x_p in the direction d [124, p.33].

Consider the function $h : \mathbb{R} \rightarrow \mathbb{R}$, defined by $h(\tau) = f(x_p + \tau d)$. To ensure that τ is close to being a stationary point of h use a condition on the gradient:

$$|\nabla f(x_{p+1})^T d| \leq c_2|\nabla f(x_p)^T d|,$$

where $c_2 \in (0, 1)$. This ensures that the gradient at the new point x_{p+1} is closer to zero than the gradient at the initial point x_p . Figure 6.3 shows how the constraints

aid selection of step size by plotting the curve $h(\tau)$ and the constraints. The sufficient decrease condition is met whenever the curve is beneath the red line; the gradient condition is met in regions denoted by the blue line. Regions indicated by the green line are regions which meet both conditions.

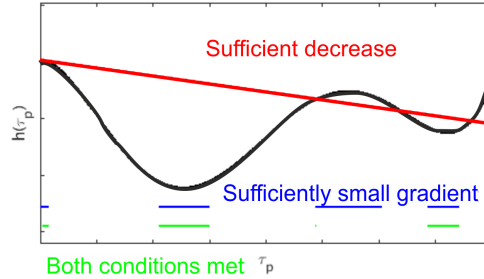


Figure 6.3: Line search conditions

Definition 6.2.6: The **strong Armijo-Wolfe conditions** are satisfied when τ is such that

$$f(x_{p+1}) \leq f(x_p) + c_1 \tau \langle \nabla f(x_p), d \rangle, \quad (6.2.26)$$

$$|\langle \nabla f(x_{p+1}), d \rangle| \leq c_2 |\langle \nabla f(x_p), d \rangle|, \quad (6.2.27)$$

where $0 < c_1 < c_2 < 1$. In practice, $c_1 = 10^{-4}$ [124, p.33] and $c_2 = 0.9$ [124, p.34]. Condition (6.2.26) is called the **Armijo condition**.

A backtracking line search is a line search method which decreases an initial step size τ_0 until a suitable τ is found. An algorithm for a backtracking line search using the strong Armijo-Wolfe conditions is given in algorithm 5. To combine the line search with the gradient descent algorithm, insert algorithm 5 at line 2 of algorithm 4. Sometimes the additional constraint that the new step size τ_p is less than or equal to the previous step size τ_{p-1} is used. This aids the backtracking line search by giving an initial step size.

Algorithm 5 Backtracking line search

- 1: Given initial point x , direction d and step size τ_0 ,
 - 2: Set $k = 0$.
 - 3: Set a scale factor ρ used to decrease the value of τ_k , $0 < \rho < 1$.
 - 4: Set condition parameters $0 < c_1 < c_2 < 1$.
 - 5: Set $x_\tau = R_x(\tau_k d)$.
 - 6: **while** strong Armijo-Wolfe conditions not satisfied **do**
 - 7: $\tau_{k+1} \leftarrow \rho \tau_k$
 - 8: $x_\tau \leftarrow R_x(\tau_{k+1} d)$.
 - 9: $k \leftarrow k + 1$
 - 10: **end while**
-

6.3 Optimisation on Generalised Stiefel Manifolds

Returning to the setting of generalised Stiefel manifolds, let $F : \mathbb{R}^{n \times k} \rightarrow \mathbb{R}$ be a smooth real-valued function and consider the constrained optimisation problem

$$\min_{X \in \mathbb{R}^{n \times k}} F(X) \text{ subject to } X^T M X = I. \quad (6.3.28)$$

Then, using the results from the previous section, we can find the gradient of the restricted function $F|_{\mathcal{S}_M}$.

Consider $\mathbb{R}^{n \times k}$ as a Riemannian manifold with metric defined via the Euclidean inner product

$$\langle X, Y \rangle_{\mathbb{R}} = \text{tr}(X^T M Y), \quad (6.3.29)$$

for $X, Y \in T_p \mathbb{R}^{n \times k} \simeq \mathbb{R}^{n \times k}$, $p \in \mathbb{R}^{n \times k}$. Consider \mathcal{S}_M as a submanifold and let F have gradient $G := \left[\frac{\partial F(X)}{\partial X_{ij}} \right]$. Then, to transform G into a restricted gradient a projection from $\mathbb{R}^{n \times k}$ to $T_X \mathcal{S}_M$ is required.

Definition 6.3.1: Let $Y \in \mathbb{R}^{n \times k}$. The **orthogonal projection** $\text{proj}_{T_X \mathcal{S}_M} : \mathbb{R}^{n \times k} \rightarrow$

$T_X\mathcal{S}_M$ is defined by [117, p.81]

$$\begin{aligned}\text{proj}_{T_X\mathcal{S}_M}(Y) &= Y - X \text{sym}(X^T M Y) \\ &= Y - \frac{X}{2}(X^T M Y + Y^T M X).\end{aligned}\tag{6.3.30}$$

It can be checked for both the Euclidean and canonical inner products that the inner product $\langle X \text{sym}(X^T M Y), Z \rangle = 0$ for all $Z \in T_X\mathcal{S}_M$, and so it is justified to describe the above projection as orthogonal.

So, for $X \in \mathcal{S}_M$ and a matrix function $F : \mathbb{R}^{n \times k} \rightarrow \mathbb{R}$ the restricted gradient on the (generalised) Stiefel manifold endowed with the Euclidean metric is

$$\nabla F|_{\mathcal{S}_M}(X) = G - X \text{sym}(X^T M G)\tag{6.3.31}$$

and the restricted gradient on the (generalised) Stiefel manifold endowed with the canonical metric is

$$\nabla F|_{\mathcal{S}_M}(X) = G - X G^T M X.\tag{6.3.32}$$

For completeness the calculations which verify these claims are below. Recall from definition 6.2.2 that $\nabla F|_{\mathcal{S}_M}(X)$ is the unique element of $T_X\mathcal{S}_M$ satisfying $\langle \nabla F|_{\mathcal{S}_M}(X), Z \rangle = \text{tr}(G^T M Z)$ for all $Z \in T_X\mathcal{S}_M$, where $\langle \cdot, \cdot \rangle$ denotes the inner product on $T_X\mathcal{S}_M$. It is checked that $\langle \nabla F|_{\mathcal{S}_M}(X), Z \rangle = \text{tr}(G^T M Z)$ and that $\nabla F|_{\mathcal{S}_M}(X) \in T_X\mathcal{S}_M$ via lemma 6.1.5.

Let $\hat{Z} = G - X \text{sym}(X^T M G)$. Then

$$\begin{aligned}\langle \hat{Z}, Z \rangle_e &= \text{tr}\left((G - X \text{sym}(X^T M G))^T M Z\right) \\ &= \text{tr}(G^T M Z) - \text{tr}\left(\text{sym}(X^T M G)^T X^T M Z\right) \\ &= \text{tr}(G^T M Z),\end{aligned}$$

as $X^T M Z = -Z^T M X$ and the trace of the product of a symmetric and a skew-symmetric matrix is zero (for proof see lemma A.12). The projection $\hat{Z} \in T_X\mathcal{S}_M$ since $\hat{Z}^T M X + X^T M \hat{Z} = G^T M X - \text{sym}(X^T M G)^T + X^T M G - \text{sym}(X^T M G) = 0$.

Let $\hat{Z} = G - XG^T M X$. Then

$$\begin{aligned} \langle \hat{Z}, Z \rangle_e &= \text{tr} \left((G^T - X^T M G X^T) (M Z - \frac{1}{2} M X X^T M Z) \right) \\ &= \text{tr}(G^T M Z) - \frac{1}{2} \text{tr}(G^T M X X^T M Z) - \frac{1}{2} \text{tr}(X^T M G X^T M Z) \\ &= \text{tr}(G^T M Z), \end{aligned}$$

since $X^T M Z = -Z^T M X$ and by exploiting the properties of trace. The projection $\hat{Z} \in T_X \mathcal{S}_M$ since $\hat{Z}^T M X + X^T M \hat{Z} = G^T M X - X^T M G + X^T M G - G^T M X = 0$.

There are two significant existing works using gradient descent on Stiefel manifolds ($M = I$),

- Siegel’s algorithm “Accelerated Gradient Descent with Function Restart Scheme” [119, Algorithm 1] which uses Nesterov acceleration [125],[126].
- Wen & Yin’s algorithm “Curvilinear Search method with BB steps” [120] which uses Barzilai-Borwein steps to accelerate [127],[128].

Both methods follow a similar structure: use a line search to find a step size (see algorithm 5) then use the Cayley transform (see definitions 6.1.13, 6.1.16 and proposition 6.1.15) to define a new point. Both methods also include an additional acceleration step and can be adapted for use on generalised Stiefel manifolds using the results in section 6.1.

6.4 Sequential Optimisation on \mathcal{S}_M

This section considers optimisation on generalised Stiefel manifolds for a specific type of function which can be written as a sum of functions acting on matrix columns. A sequential algorithm for optimisation is presented in algorithm 6. This is based on [119, Algorithm 1], a gradient descent algorithm for optimisation on the Stiefel manifold, using the Cayley retraction. As discussed in section 6.1 this requires inversion of a $2k \times 2k$ matrix, and although k is generally chosen to be much smaller

than n in applications this could still be quite large. In an aim to reduce the dimensionality of the problem the sequential algorithm searches for a solution on an ellipsoid formed by the intersection of a larger ellipsoid and a linear subspace of \mathbb{R}^n .

Let $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$, $i = 1, \dots, k$ and define $F : \mathbb{R}^{n \times k} \rightarrow \mathbb{R}$ by

$$F(\bar{\Phi}) = \sum_{i=1}^k f_i(\phi_i), \quad (6.4.33)$$

where ϕ_i is the i -th column of the matrix $\bar{\Phi} \in \mathbb{R}^{n \times k}$.

Examples 6.4.1: Let A and W be $n \times n$ matrices and define $f : \mathbb{R}^n \rightarrow \mathbb{R}$ by

(i) $f(\phi) = \phi^T W \phi$, then F as defined in (6.4.33) can be written as $F(\bar{\Phi}) = \text{tr}(\bar{\Phi}^T W \bar{\Phi})$.

(ii) $f(\phi) = \phi^T W \phi + \|A\phi\|_1$, then F as defined in (6.4.33) can be written as $F(\bar{\Phi}) = \text{tr}(\bar{\Phi}^T W \bar{\Phi}) + \|A\bar{\Phi}\|_1$.

Recall, the aim is to find a matrix $\bar{\Phi} \in \mathbb{R}^{n \times k}$ minimising a function $F : \mathbb{R}^{n \times k} \rightarrow \mathbb{R}$ such that $\bar{\Phi}^T M \bar{\Phi} = I$. Instead of seeking the $n \times k$ matrix minimiser, consider a sequential construction, searching for columns of $\bar{\Phi}$ one by one: Let $\phi_m \in \mathbb{R}^n$ denote the m -th vector found via the following optimisation and let Φ denote the matrix $[\phi_1, \dots, \phi_m]$. Then the $(m+1)$ -th column-vector is found by solving

$$\arg \min_{\phi_{m+1}} F(\phi_{m+1}) \text{ subject to } \begin{cases} \phi_{m+1}^T M \Phi = 0, \\ \phi_{m+1}^T M \phi_{m+1} = 1. \end{cases} \quad (6.4.34)$$

To do this, begin by considering the generalised Stiefel manifold with $k = 1$. That is, $\mathcal{S}_M = \{x \in \mathbb{R}^n : x^T M x = 1\}$. Compare this to definition 3.2.8 and the following proposition is clear:

Proposition 6.4.2: The generalised Stiefel manifold $\mathcal{S}_M \subset \mathbb{R}^n$ is an $(n-1)$ -dimensional ellipsoid. That is $\mathcal{S}_M = \mathbb{E}_M$.

Then define

$$V_{\Phi} := \{x \in \mathbb{R}^n : x^T M \Phi = 0\}.$$

Lemma 6.4.3: Let M be an $n \times n$ symmetric positive definite matrix and let $\Phi = [\phi_1 \dots \phi_m]$ be an $n \times m$ matrix formed by orthonormal vectors ϕ_i (i.e. $\Phi^T M \Phi = I_m$). Then, V_Φ is an $(n - m)$ -dimensional linear subspace of \mathbb{R}^n .

Proof. By definition V_Φ is the null space of the matrix $M\Phi$, hence V_Φ is a linear subspace of \mathbb{R}^n . To calculate the dimension consider the rank-nullity theorem:

$$\text{rank}(M\Phi) + \text{null}(M\Phi) = n.$$

Since Φ has orthogonal columns, $\text{rank}(\Phi) = m$. The rank of M is n since all eigenvalues of positive definite matrices are non-negative, and the rank of a symmetric matrix is equal to the number of non-zero eigenvalues. Note that $\text{rank}(M\Phi) = \text{rank}(\Phi) = m$ (for proof see lemma A.13). Hence, via the rank-nullity theorem, the dimension of $V_\Phi = \text{Null}(M\Phi) = n - \text{Rank}(M\Phi) = n - m$. \square

Remark 6.4.4: The set of points $V_\Phi \cap \mathbb{E}_M$ is an $(n - m - 1)$ -dimensional ellipsoid. To see this, let $\{v_i\}$ be a set of basis vectors for V_Φ , represented as columns of a matrix V . Then any vector in V_Φ can be written as Vb where $b \in \mathbb{R}^{n-m}$, so $V_\Phi \cap \mathbb{E}_M = \{b \in \mathbb{R}^{n-m} : b^T V^T M V b = 1\} = \mathbb{E}_{V^T M V}$. Note that the matrix $V^T M V$ is symmetric positive definite: symmetry is due to the symmetry of M ; positive definite-ness follows from V being full rank (so $Vx = 0 \iff x = 0$) and M being positive definite.

The minimisation problem (6.4.34) can be rephrased as find

$$\arg \min_{\phi_{m+1}} F(\phi_{m+1}) \text{ subject to } \phi_{m+1} \in V_\Phi \cap \mathbb{E}_M. \quad (6.4.35)$$

The new aim is to minimise the function F restricted to the space $V_\Phi \cap \mathbb{E}_M$. Denote the restricted function by $F|_{V \cap E} : V_\Phi \cap \mathbb{E}_M \rightarrow \mathbb{R}$. The optimisation problem (6.4.35) can then be written as solve

$$\arg \min_{\phi_{m+1}} F|_{V \cap E}(\phi_{m+1}). \quad (6.4.36)$$

To restrict to the submanifold $V_\Phi \cap \mathbb{E}_M \subseteq \mathbb{R}^n$ a projection $\mathbb{R}^n \rightarrow V_\Phi \cap \mathbb{E}_M$ is required, along with a projection into the tangent space. To construct these first consider projection into the linear subspace V_Φ :

Definition 6.4.5: The **M-orthogonal projection** onto V_Φ , $\text{proj}_{V,M} : \mathbb{R}^n \rightarrow V_\Phi$, is defined by

$$\text{proj}_{V,M}(x) = x - \sum_{l=1}^m (x^T M \phi_l) \phi_l. \quad (6.4.37)$$

This takes a vector and removes any parts which are parallel to the columns of Φ . To check that $\text{proj}_{V,M}(x) \in V_\Phi$ consider $\text{proj}_{V,M}(x)^T M \Phi$:

$$\begin{aligned} \text{proj}_{V,M}(x)^T M \Phi &= \left(x^T - \sum_{l=1}^m (x^T M \phi_l) \phi_l^T \right) M \Phi \\ &= x^T M \Phi - \sum_{l=1}^m (x^T M \phi_l) \phi_l^T M \Phi \\ &= x^T M \Phi - \sum_{l=1}^m (x^T M \phi_l) e_l, \text{ where } e_l \text{ is the standard basis vector,} \\ &= x^T M \Phi - x^T M \Phi \\ &= 0, \text{ as expected.} \end{aligned}$$

Remark 6.4.6: It is useful to note that

$$\begin{aligned} \text{proj}_{V,M}(x) &= x - \sum_{l=1}^m (x^T M \phi_l) \phi_l \\ &= x - \left(\sum_{l=1}^m \phi_l \phi_l^T \right) M x. \end{aligned}$$

The matrix given by $\phi_1 \phi_1^T + \dots + \phi_m \phi_m^T$ has entries given by $\sum_{l=1}^m \Phi_{il} \Phi_{jl} = \sum_{l=1}^m \Phi_{il} (\Phi^T)_{lj}$ since $\Phi_{ij} = (\phi_j)_i$ and hence,

$$\text{proj}_{V,M}(x) = x - \Phi \Phi^T M x = (I - \Phi \Phi^T M) x. \quad (6.4.38)$$

That is, the projection is given by the linear transformation described by the matrix $(I - \Phi \Phi^T M)$. This can be used to help describe the tangent space of the ellipsoid $V_\Phi \cap \mathbb{E}_M$ at a point x .

Definition 6.4.7: Two submanifolds \mathcal{P}, \mathcal{Q} of a manifold \mathcal{M} are **transverse** if for all $x \in \mathcal{P} \cap \mathcal{Q}$, the span of the tangent spaces to the submanifolds at x is equal to the tangent space of \mathcal{M} at x . That is, $T_x \mathcal{P} + T_x \mathcal{Q} = T_x \mathcal{M}$.

Lemma 6.4.8: Consider the spaces V_Φ and \mathbb{E}_M as submanifolds of \mathbb{R}^n . Then V_Φ and \mathbb{E}_M are transverse.

Proof. Let $x \in V_\Phi$ and recall that V_Φ is an $(n - m)$ -dimensional linear subspace of \mathbb{R}^n . Therefore

$$T_x V_\Phi = V_\Phi = \{z \in \mathbb{R}^n : z^T M \Phi = 0\}. \quad (6.4.39)$$

Similarly, let $x \in \mathbb{E}_M$ and recall that \mathbb{E}_M is an $(n - 1)$ -dimensional ellipsoid. Therefore the tangent space $T_x \mathbb{E}_M$ is given by the $(n - 1)$ -dimensional linear subspace of \mathbb{R}^n orthogonal to x . That is,

$$T_x \mathbb{E}_M = \{z \in \mathbb{R}^n : z^T M x = 0\}. \quad (6.4.40)$$

Now, let $x \in V_\Phi \cap \mathbb{E}_M$ and consider the tangent spaces at x : since $x \in T_x V_\Phi$ and $T_x \mathbb{E}_M$ is the $(n - 1)$ -dimensional linear subspace of \mathbb{R}^n orthogonal to x , $T_x \mathcal{P} + T_x \mathcal{Q} = \mathbb{R}^n$. Therefore, since $T_x \mathbb{R}^n = \mathbb{R}^n$, V_Φ and \mathbb{E}_M are transverse. \square

Lemma 6.4.9: Let \mathcal{P}, \mathcal{Q} be transverse submanifolds of a manifold \mathcal{M} and let $x \in \mathcal{P} \cap \mathcal{Q}$. Then, $T_x \mathcal{P} \cap T_x \mathcal{Q} = T_x(\mathcal{P} \cap \mathcal{Q})$. [129, p.203]

Let $x \in V_\Phi \cap \mathbb{E}_M$, then the above lemma can be applied to describe the tangent space $T_x(V_\Phi \cap \mathbb{E}_M)$. Recall that any $z \in \mathbb{R}^n$ can be projected into V_Φ via transformation by the matrix $(I - \Phi \Phi^T M)$ (see remark 6.4.6). Similarly, any $z \in \mathbb{R}^n$ can be projected into the linear subspace $T_x \mathbb{E}_M$ via transformation by the matrix $(I - x x^T M)$.

These transformations commute:

$$\begin{aligned} (I - \Phi \Phi^T M)(I - x x^T M) &= I - \Phi \Phi^T M - x x^T M + \Phi \Phi^T M x x^T M \\ &= I - \Phi \Phi^T M - x x^T M, \text{ since } \Phi^T M x = 0, \\ &= (I - x x^T M)(I - \Phi \Phi^T M). \end{aligned}$$

Let

$$P := I - \Phi \Phi^T M - x x^T M. \quad (6.4.41)$$

Then any $z \in \mathbb{R}^n$ can be projected into both $T_x V_\Phi$ and $T_x \mathbb{E}_M$ via Pz , and so $Pz \in T_x V_\Phi \cap T_x \mathbb{E}_M = T_x(V_\Phi \cap \mathbb{E}_M)$ by lemma 6.4.9. In summary:

$$z \in T_x(V_\Phi \cap \mathbb{E}_M) \text{ if both } \Phi^T Mz = 0 \text{ and } x^T Mz. \quad (6.4.42)$$

The projection P is used to translate information about the function F to information about the restricted function $F|_{V \cap E}$. In particular, the gradient of $F|_{V \cap E}$ at x is equal to the gradient of F projected in to the tangent space $T_x(V_\Phi \cap \mathbb{E}_m)$, see [117, 3.37].

Definition 6.4.10: Let $F : \mathbb{R}^n \rightarrow \mathbb{R}$ be a differentiable function, and let $x \in \mathbb{R}^n$. Then define $g \in \mathbb{R}^n$ as the $n \times 1$ vector with entries

$$g_i := \frac{\partial}{\partial x_i} F(x). \quad (6.4.43)$$

The vector g represents the **gradient** of F at x .

Let $V_\Phi \cap \mathbb{E}_M$ be endowed with either the Euclidean or canonical metric. Then the gradient of $F|_{V \cap E}$ at $x \in V_\Phi \cap \mathbb{E}_M$ is given by

$$\nabla F|_{V \cap E}(x) = Pg. \quad (6.4.44)$$

Again, for completeness the calculations to verify this claim are given here. It is clear from the construction of P that $Pg \in T_x(V_\Phi \cap \mathbb{E}_M)$ so all that is required is to check that $\langle Pg, z \rangle = \text{tr}(g^T Mz)$ for all $z \in T_x(V_\Phi \cap \mathbb{E}_M)$. For the Euclidean metric,

$$\begin{aligned} \langle Pg, z \rangle_e &= \text{tr}(g^T P^T Mz) \\ &= \text{tr}(g^T Mz) - \text{tr}(g^T M\Phi\Phi^T Mz) - \text{tr}(g^T Mxx^T Mz) \\ &= \text{tr}(g^T Mz), \text{ since } \Phi^T Mz = 0 \text{ and } x^T Mz = 0 \text{ as } z \in T_x(V_\Phi \cap \mathbb{E}_M). \end{aligned}$$

For the canonical metric,

$$\begin{aligned} \langle Pg, z \rangle_c &= \text{tr}\left(g^T P^T \left(Mz - \frac{1}{2}Mxx^T Mz\right)\right) \\ &= \text{tr}(g^T Mz) - \frac{1}{2} \text{tr}(g^T P^T Mxx^T Mz), \text{ since } \text{tr}(g^T P^T Mz) = \text{tr}(g^T Mz), \\ &= \text{tr}(g^T Mz), \text{ since } x^T Mz = 0, \text{ as } z \in T_x(V_\Phi \cap \mathbb{E}_M). \end{aligned}$$

Remark 6.4.11: Note that Additional notes on the matrix P can be found in appendix F.

Definition 6.4.12: Let $\text{proj}_E : \mathbb{R}^n \rightarrow \mathbb{E}_M$ be the projection on to the ellipsoid defined by

$$\text{proj}_E(x) = \frac{x}{\sqrt{x^T M x}}. \quad (6.4.45)$$

This takes a vector in \mathbb{R}^n and scales it to lie on the ellipsoid \mathbb{E}_M , hence is referred to as **radial projection**.

Let $\text{proj}_{V \cap E} : \mathbb{R}^n \rightarrow V_\Phi \cap \mathbb{E}_M$ be defined by

$$\text{proj}_{V \cap E}(x) = \text{proj}_E(\text{proj}_{V,M}(x)). \quad (6.4.46)$$

It is clear that $\text{proj}_{V \cap E}(x) \in \mathbb{E}_M$ but to check that it also lies in V_Φ consider $\text{proj}_{V \cap E}(x)^T M \Phi$. Let $y = \text{proj}_{V,M}(x)$, so $y^T M \Phi = 0$. Then

$$\text{proj}_E(y)^T M \Phi = \frac{y^T}{\sqrt{y^T M y}} M \Phi = \left(\frac{1}{\sqrt{y^T M y}} \right) y^T M \Phi = 0, \text{ since } y \in V_\Phi.$$

And so $\text{proj}_{V \cap E} \in V_\Phi$.

The fact that the projection $\text{proj}_{V,M}$ is linear leads to the following useful corollary:

Corollary 6.4.13: If $x \in V_\Phi$ and $y \in \mathbb{R}^n$, then

$$\text{proj}_{V \cap E}(x + \alpha y) = \text{proj}_E(x + \alpha \text{proj}_{V,M}(y)).$$

Proof. Consider the projection on to $V_\Phi \cap \mathbb{E}_M$:

$$\begin{aligned} \text{proj}_{V \cap E}(x + \alpha y) &= \text{proj}_E(\text{proj}_{V,M}(x + \alpha y)) \\ &= \text{proj}_E(\text{proj}_{V,M}(x) + \alpha \text{proj}_{V,M}(y)), \text{ since } \text{proj}_{V,M} \text{ is linear,} \\ &= \text{proj}_E(x + \alpha \text{proj}_{V,M}(y)), \text{ since } x \in V_\Phi \Rightarrow \text{proj}_{V,M}(x) = x. \end{aligned}$$

□

Remark 6.4.14: Note that, in general, $\text{proj}_{V,M}(\text{proj}_E(x)) \notin V_\Phi \cap \mathbb{E}_M$.

Proposition 6.4.15: Let $x \in \mathbb{E}_M$ and $z \in T_x \mathbb{E}_M$, then the map $R_x^{\text{proj}} : T_x \mathbb{E}_M \rightarrow \mathbb{E}_M$ defined by

$$R_x^{\text{proj}}(z) = \text{proj}_E(x + z) \quad (6.4.47)$$

is a retraction.

Proof. It is clear that $R_x^{\text{proj}}(z) \in \mathbb{E}_M$, so check the properties of a retraction:

(i) $R_x^{\text{proj}}(0) = \text{proj}_E(x) = x$ since $x \in \mathbb{E}_M$.

(ii)

$$\begin{aligned} & \left. \frac{d}{d\tau} \right|_{\tau=0} R_x^{\text{proj}}(x + \tau z) \\ &= \left. \frac{d}{d\tau} \right|_{\tau=0} (x + \tau z) \left((x + \tau z)^T M(x + \tau z) \right)^{-\frac{1}{2}} \\ &= z \left((x + \tau z)^T M(x + \tau z) \right)^{-\frac{1}{2}} \\ &\quad - \frac{1}{2} (x + \tau z) \left((x + \tau z)^T M(x + \tau z) \right)^{-\frac{3}{2}} (z^T M(x + \tau z) + (x + \tau z)^T Mz) \Big|_{\tau=0} \\ &= z (x^T Mx)^{-\frac{1}{2}} - \frac{1}{2} x (x^T Mx)^{-\frac{3}{2}} (z^T Mx + x^T Mz) \\ &= z, \text{ since } x^T Mx = 1 \text{ and } z^T Mx = x^T Mz = 0 \text{ as } z \in T_x \mathbb{E}_M. \end{aligned}$$

□

Remark 6.4.16: When $x \in V_\Phi \cap \mathbb{E}_M$ and $\tau z \in T_x(V_\Phi \cap \mathbb{E}_M)$ the retracted point $R_x^{\text{proj}}(\tau z) \in V_\Phi \cap \mathbb{E}_M$ and so R_x^{proj} can be used as a retraction onto $V_\Phi \cap \mathbb{E}_M$.

Definition 6.4.17: The map R_x^{proj} is called the **radial retraction**.

Remark 6.4.18: Let L be the straight line between the point $x + \tau z$ and the origin $0 \in \mathbb{R}^{n \times k}$. Then the projection $\text{proj}_E(x + \tau z)$ will project the point $x + \tau z$ to the point of intersection between the ellipsoid E_M and the line L , hence the name *radial* retraction. There will be an accumulation point at the point of intersection between the ellipsoid \mathbb{E}_M and the parallel line \hat{L} , where \hat{L} is the straight line in the direction of L which passes through 0. Figure 6.4 shows this on a 2-dimensional ellipsoid (ellipse).

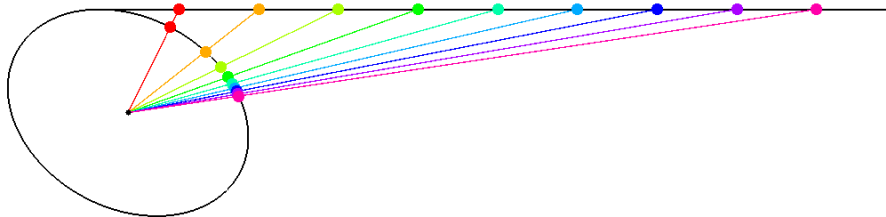


Figure 6.4: Projection onto the ellipse via a line through the centre.

Consider the following alternative: Any $y \in T_x \mathbb{E}_M$ can be written as $y = \tau \hat{y}$ where $\tau > 0$ and $\hat{y} = \frac{y}{\|y\|}$. From here it is assumed that d is such a unit vector.

Proposition 6.4.19: Let $\mathbb{E}_M \subset \mathbb{R}^n$ be an $(n-1)$ -dimensional ellipsoid, as described above, and let $x \in \mathbb{E}_M$. The map $\tilde{R}_x : T_x \mathbb{E}_M \rightarrow \mathbb{E}_M$ defined by [117, 4.31]

$$\tilde{R}_x(\tau d) = \cos(\tau) x + \sin(\tau) d \quad (6.4.48)$$

is a retraction.

Proof. To check first ensure that $\tilde{R}_x(\tau d) \in \mathbb{E}_M$:

$$\tilde{R}_x(\tau d)^T M \tilde{R}_x(\tau d) = \cos^2(\tau) x^T M x + \sin^2(\tau) d^T M d + 2 \cos(\tau) \sin(\tau) x^T M d.$$

Then, since $x^T M x = d^T M d = 1$ and $x^T M d = 0$ as $d \in T_x \mathbb{E}_M$ (see equation (6.4.40)),

$$\tilde{R}_x(\tau d)^T M \tilde{R}_x(\tau d) = \cos^2(\tau) + \sin^2(\tau) = 1.$$

Hence, $\tilde{R}_x(\tau d) \in V_\Phi \cap \mathbb{E}_M$. Then check the properties for a retraction:

- (i) $\tilde{R}_x(0) = \cos(0) x + \sin(0) d = 0$.
- (ii) $\frac{d}{d\tau} \Big|_{\tau=0} \tilde{R}_x(\tau d) = [-\sin(\tau) x + \cos(\tau) d]_{\tau=0} = d$.

So \tilde{R}_x is a retraction on \mathbb{E}_M . □

Definition 6.4.20: The retraction \tilde{R}_x is called the **periodic retraction**.

Remark 6.4.21: When $x \in V_\Phi \cap \mathbb{E}_M$ and $\tau d \in T_x(V_\Phi \cap \mathbb{E}_M)$ the retracted point $\tilde{R}_x(\tau d) \in V_\Phi \cap \mathbb{E}_M$. Therefore \tilde{R}_x can be used as a retraction onto $V_\Phi \cap \mathbb{E}_M$.

Remark 6.4.22: The retraction is called *periodic* because $\tilde{R}_x(\tau d) = \tilde{R}_x((2\pi n + \tau)d)$ for $n \in \mathbb{N}$. This follows from the periodic qualities of \cos and \sin and is illustrated in figure 6.5 for an ellipsoid in \mathbb{R}^2 .

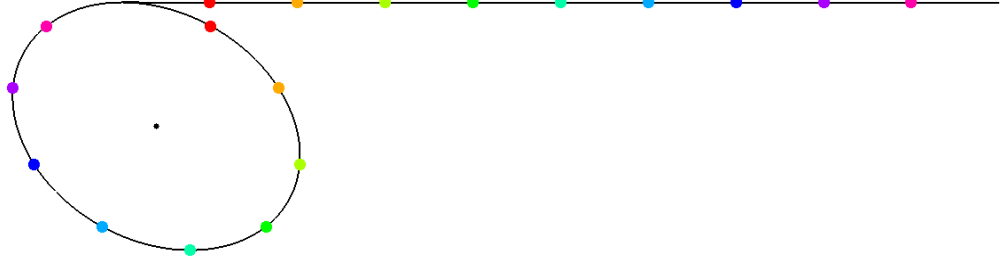


Figure 6.5: Projection onto the ellipse via periodic retraction.

Remark 6.4.23: The periodic retraction does not extend simply to generalised Stiefel manifolds of the form $\mathcal{S}_M = \{X \in \mathbb{R}^{n \times k} : X^T M X = I_k\}$, with $k > 1$. The obvious extension to generalised Stiefel manifolds is given by the function from $T_X \mathcal{S}_M$ defined by

$$\tilde{R}_X(Y) = \cos(\|Y\|) X + \frac{\sin(\|Y\|)}{\|Y\|} Y,$$

where $\|Y\| = \text{tr}(Y^T M Y)^{\frac{1}{2}}$, for $Y \in T_X \mathcal{S}_M$. The hope is that $\tilde{R}_X(Y) \in \mathcal{S}_M$. That is, that $\tilde{R}_X(Y)^T M \tilde{R}_X(Y) = I_k$, but

$$\begin{aligned} \tilde{R}_X(Y)^T M \tilde{R}_X(Y) &= \left(\cos(\|Y\|) X^T + \frac{\sin(\|Y\|)}{\|Y\|} Y^T \right) \left(\cos(\|Y\|) M X + \frac{\sin(\|Y\|)}{\|Y\|} M Y \right) \\ &= \cos^2(\|Y\|) X^T M X + \frac{\sin^2(\|Y\|)}{\|Y\|^2} Y^T M Y \\ &\quad + \frac{\cos(\|Y\|) \sin(\|Y\|)}{\|Y\|} (X^T M Y + Y^T M X). \end{aligned}$$

Then since $Y \in T_X \mathcal{S}_M$, $X^T M Y + Y^T M X = 0$,

$$\tilde{R}_X(Y)^T M \tilde{R}_X(Y) = \cos^2(\|Y\|) I_k + \frac{\sin^2(\|Y\|)}{\|Y\|^2} Y^T M Y.$$

This is equal to I_k if and only if

$$\frac{Y^T M Y}{\|Y\|^2} = I_k. \tag{6.4.49}$$

Recall that $\|Y\|^2 = \text{tr}(Y^T M Y)$ so rearranging equation (6.4.49) gives $Y^T M Y = \text{tr}(Y^T M Y) I_k$. Taking the trace of both sides gives $\text{tr}(Y^T M Y) = k \text{tr}(Y^T M Y)$, which only holds when $k = 1$. Therefore the function only takes points in $T_X \mathcal{S}_M$ to \mathcal{S}_M when $\mathcal{S}_M \subset \mathbb{R}^n$. That is, when $\mathcal{S}_M = \mathbb{E}_M$.

Proposition 6.4.24: Let the function $h : \mathbb{R} \rightarrow \mathbb{R}$ be defined by $h(\tau) = F(\tilde{R}_x(\tau d))$ for some fixed $x \in V_\Phi \cap \mathbb{E}_M$, $d \in T_x(V_\Phi \cap \mathbb{E}_M)$ with $d^T M d = 1$. If F is such that $F(x) = F(-x)$ then $h(\tau)$ is π -periodic.

Proof. Consider $h(\tau + \pi) = F(\tilde{R}_x((\tau + \pi)d))$.

$$\begin{aligned} \tilde{R}_x((\tau + \pi)d) &= \cos(\tau + \pi) x + \sin(\tau + \pi) d, \text{ since } d^T M d = 1, \\ &= -\cos(\tau) x - \sin(\tau) d \\ &= -\tilde{R}_x(\tau d). \end{aligned}$$

Then, since $F(x) = F(-x)$, $h(\tau + \pi) = F(-\tilde{R}_x(\tau d)) = -F(\tilde{R}_x(\tau d)) = h(\tau)$, so h is π -periodic. \square

Corollary 6.4.25: Assume d is a descent direction. Then for a backtracking line search on the function $h(\tau)$ the initial step need not be larger than $\pi/2$.

An algorithm for sequential gradient descent is given in algorithm 6. In practice the optimisation is accelerated via a function restart scheme. A maximum number of iterations is set, along with an additional convergence condition – if the distance between a point x_k and the new point x_{k+1} is sufficiently small the algorithm is deemed to have converged.

Algorithm 6 Sequential gradient descent on \mathcal{S}_M

- 1: Set convergence tolerance ε .
 - 2: Set $\Phi = 0_{n \times 1}$.
 - 3: Set $p = 1$
 - 4: **while** $p \leq k$ **do**
 - 5: Generate random $x \in V_\Phi \cap \mathbb{E}_M$.
 - 6: **while** $\|\nabla F|_{V \cap E}(x)\| > \varepsilon$ **do**
 - 7: Set descent direction $d = -\nabla F|_{V \cap E}(x)$.
 - 8: Perform line search to find step size τ for next iterate.
 - 9: Use $\tau_0 = \pi/2$.
 - 10: Use iterates $x_\tau = \tilde{R}_x(\tau d)$.
 - 11: $x \leftarrow \tilde{R}_x(\tau d)$.
 - 12: **end while**
 - 13: **if** $p = 1$ **then**
 - 14: $\Phi \leftarrow x$
 - 15: **else**
 - 16: $\Phi \leftarrow [\Phi, x]$
 - 17: **end if**
 - 18: **end while**
-

6.5 Approximating Sequential Compressed Manifold Modes

The method of sequential optimisation on the generalised Stiefel manifold can be used to approximate compressed manifold modes. Recall that compressed modes ϕ minimise $\text{tr}(\phi^T W \phi) + \|A\phi\|_1$ with $\phi^T A \phi = 1$ and each mode orthogonal to every other mode. (A and W are the area and weight matrix of a discrete Laplacian operator.) To approximate compressed modes a differentiable function which approximates the ℓ_1 norm is required.

To find such a function first recall that for $X \in \mathbb{R}^{n \times k}$, $\|X\|_1 = \sum_{ij} |X_{ij}|$. The ℓ_1 norm can then be approximated by a continuously differentiable function which approximates the absolute value.

Definition 6.5.1: A function $\bar{f} : \mathbb{R} \times (0, \infty) \rightarrow \mathbb{R}$ is a **smoothing of the ℓ_1 norm** if

- (i) $\bar{f}(0, \varepsilon) = 0$, for all $\varepsilon \in (0, \infty)$,
- (ii) $\bar{f}(x, \varepsilon) \rightarrow |x|$ as $\varepsilon \rightarrow 0$,
- (iii) $\frac{\partial}{\partial x} \bar{f}(x, \varepsilon) \rightarrow \text{sgn}(x)$ as $\varepsilon \rightarrow 0$.

For ease of notation, define $f_{*,\varepsilon}(x) := \bar{f}(x, \varepsilon)$. A list of viable functions $f_{*,\varepsilon}$ and their first partial derivatives (with respect to x) is given in table 6.1. The functions are illustrated in figures 6.6 and 6.7, with figure 6.7 showing an extreme close up about zero. (Note that the subscript ε is omitted in the function names.) The parameters ε and δ are both set to be 10^{-3} . Figure 6.8 shows the function gradients. The function f_0 denotes the absolute value, included for reference. Note also that when x is very small the absolute difference between $f_0(x)$ and $f_{*,\varepsilon}(x)$ becomes even smaller. Figure 6.9 demonstrates this, plotting values of $|f_{*,\varepsilon}(x) - \text{abs}(x)|$ for $x \in [10^{-12}, 10^{10}]$.

The error in computational accuracy due to machine rounding errors can be bounded by a value called *the machine epsilon*. As the parameter ε decreases below the

machine epsilon calculations become inaccurate and so ε can be replaced with zero. Therefore, assuming a very small ε the absolute value function f_0 can be used rather than a smoothed approximation, and hence this function is included in some of the following figures. Figure 6.10 shows the first 10 approximated modes on the ScapeMan001 mesh (12500 vertices), via a selection of the ℓ_1 smoothings. Algorithm parameters are detailed in the figure caption. The approximated CMMs were ordered by compressed eigenvalue (see section 3.2.3).

The approximated modes are not consistent. That is, between approximations the modes vary. There is not always a clear correspondence between modes, nor is there a consistent ordering. In the classical eigenfunction case inconsistent ordering occurs when eigenvalues are very close, a consequence of symmetries in the shape. To check if this is the case for these approximated modes the compressed eigenvalues are plotted in figure 6.11. Consider the mode supported on the left foot which appears as the 7th mode for f_0 ; the 2nd mode for f_1 ; the 2nd mode for f_3 ; and the 1st mode for f_9 . If the approximated modes exhibit the same behaviour as the Laplacian eigenfunctions for modes with similar eigenvalue then it would be expected that the compressed eigenfunctions corresponding to these modes is very similar but this is not the case.

Recall that when the parameter $\mu = 0$ the output matrix $\bar{\Phi}$ should be the matrix with the first k eigenfunctions as columns. Figure 6.12 shows eigenfunctions calculated on the homer mesh (5103 vertices) via the `eigs` function in MATLAB and then the first 10 sequential eigenfunctions calculated using the sequential gradient descent on \mathcal{S}_M algorithm. Again, algorithm parameters are detailed in the figure caption. Comparing the rows of the figure shows a much greater consistency between the sequentially approximated eigenfunctions and the `eigs` eigenfunctions. This suggests that the inconsistency between methods for CMMs is a consequence of the difficulty in calculating CMMs rather than a flaw of the algorithm.

Table 6.1: Smoothings of the l_1 norm.

Name	$f_{*,\varepsilon}(x)$	$\frac{\partial}{\partial x} f_{*,\varepsilon}(x)$
$f_0(x)$	$ x $	$\text{sgn}(x)$
$f_{1,\varepsilon}(x)$	$ x + \frac{\varepsilon}{1+ x } - \varepsilon$	$\text{sgn}(x) - \frac{\varepsilon \text{sgn}(x)}{(1+ x)^2}$
$f_{2,\varepsilon}(x)$	$ x - \frac{\varepsilon}{2(1+ x)} - \varepsilon \log(\varepsilon + x) + \frac{\varepsilon}{2} + \varepsilon \log(\varepsilon)$	$\text{sgn}(x) + \frac{\varepsilon \text{sgn}(x)}{2(1+ x)^2} - \frac{\varepsilon \text{sgn}(x)}{\varepsilon+ x }$
$f_{3,\varepsilon}(x)$	$ x - \varepsilon \log(\varepsilon + x) + \varepsilon \log(\varepsilon)$	$\text{sgn}(x) - \frac{\varepsilon \text{sgn}(x)}{\varepsilon+ x }$
$f_{4,\varepsilon}(x)$	$ x + \varepsilon \exp(- x) - \varepsilon$	$\text{sgn}(x) - \varepsilon \text{sgn}(x) \exp(- x)$
$f_{5,\varepsilon}(x)$	$ x + \varepsilon \exp(-\varepsilon x) - \varepsilon$	$\text{sgn}(x) - \varepsilon^2 \text{sgn}(x) \exp(-\varepsilon x)$
$f_{6,\varepsilon}(x)$	$\begin{cases} \frac{x^2}{2\delta}, & \text{if } x \leq \delta \\ x - \frac{\delta}{2}, & \text{if } x > \delta \end{cases}$	$\begin{cases} \frac{x}{\delta}, & \text{if } x \leq \delta \\ \text{sgn}(x), & \text{if } x > \delta \end{cases}$
$f_{7,\varepsilon}(x)$	$\begin{cases} \frac{\varepsilon+\delta}{\delta} \left(x + \varepsilon \log\left(\frac{\varepsilon}{\varepsilon+ x }\right) \right), & \text{if } x \leq \delta \\ x + \varepsilon \left(1 + \frac{\varepsilon+\delta}{\delta} \log\left(\frac{\delta}{\varepsilon+\delta}\right) \right), & \text{if } x > \delta \end{cases}$	$\begin{cases} \frac{\varepsilon+\delta}{\delta} \left(\text{sgn}(x) - \frac{\varepsilon \text{sgn}(x)}{\varepsilon+ x } \right), & \text{if } x \leq \delta \\ \text{sgn}(x), & \text{if } x > \delta \end{cases}$
$f_{8,\varepsilon}(x)$	$\varepsilon \log(\cosh(\frac{x}{\varepsilon}))$	$\tanh(\frac{x}{\varepsilon})$
$f_{9,\varepsilon}(x)$	$ x - \delta\varepsilon \log(\varepsilon + x) + \delta\varepsilon \log(\varepsilon)$	$\text{sgn}(x) - \frac{\delta\varepsilon \text{sgn}(x)}{\varepsilon+ x }$

Functions 1-5 are based on functions found in table III in the paper [130]. Function 6 is the Moreau-Yosida envelope where δ controls the width of the envelope [119, (5.1.3)]. Function 7 is an adaptation of the Moreau-Yosida envelope via function 1. Function 8 is based on the Green potential function [130, Table II]. Function 9 is based on function 3, with an additional parameter to reduce the size of the smoothing terms.

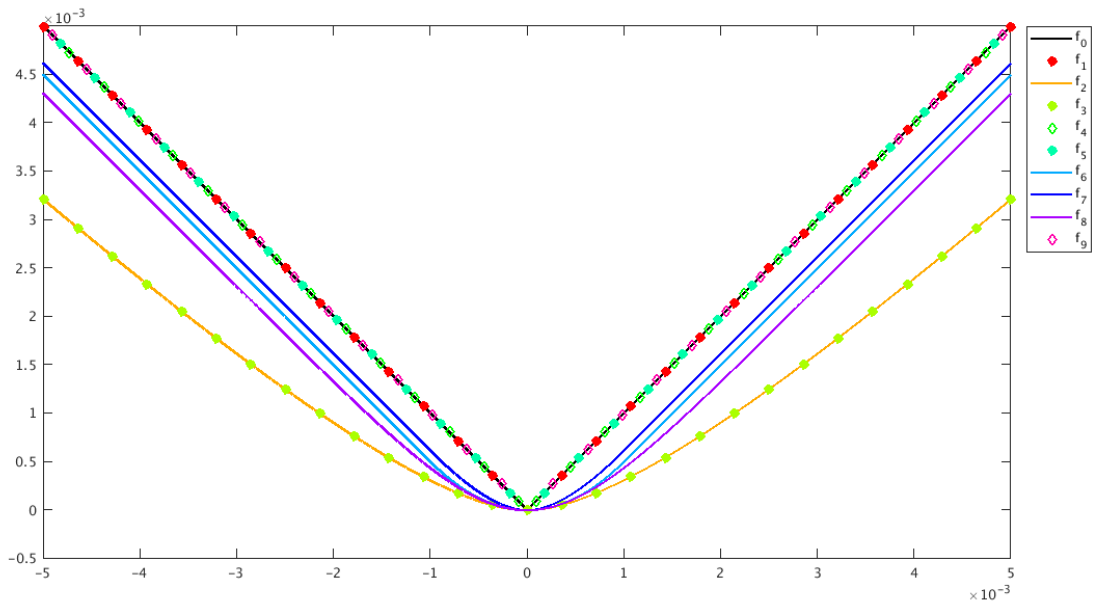


Figure 6.6: Smoothed approximations of the modulus function.

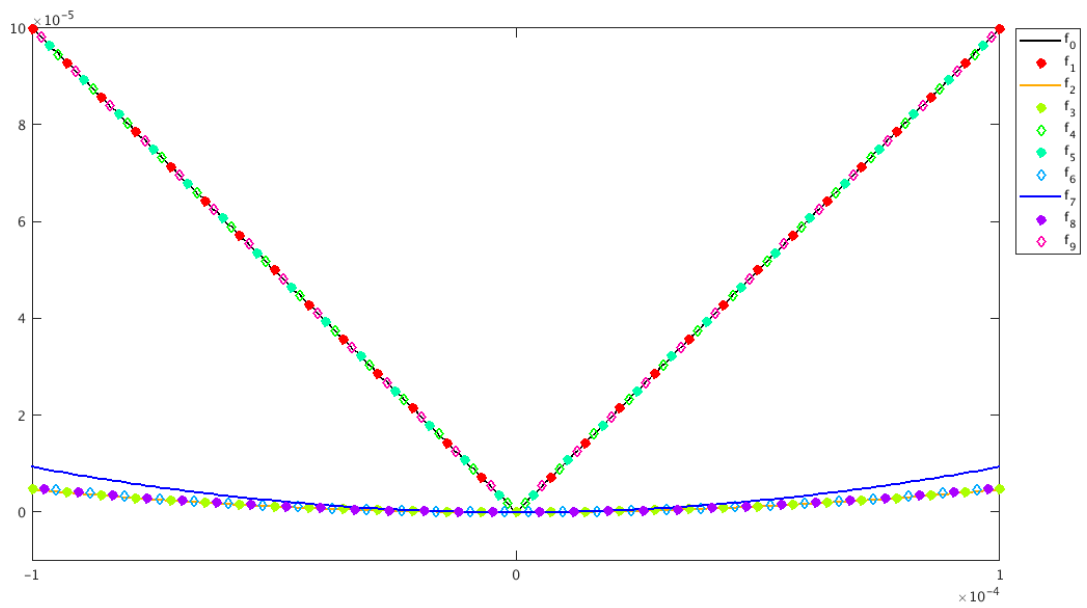


Figure 6.7: Smoothed approximations of the modulus function close to zero.

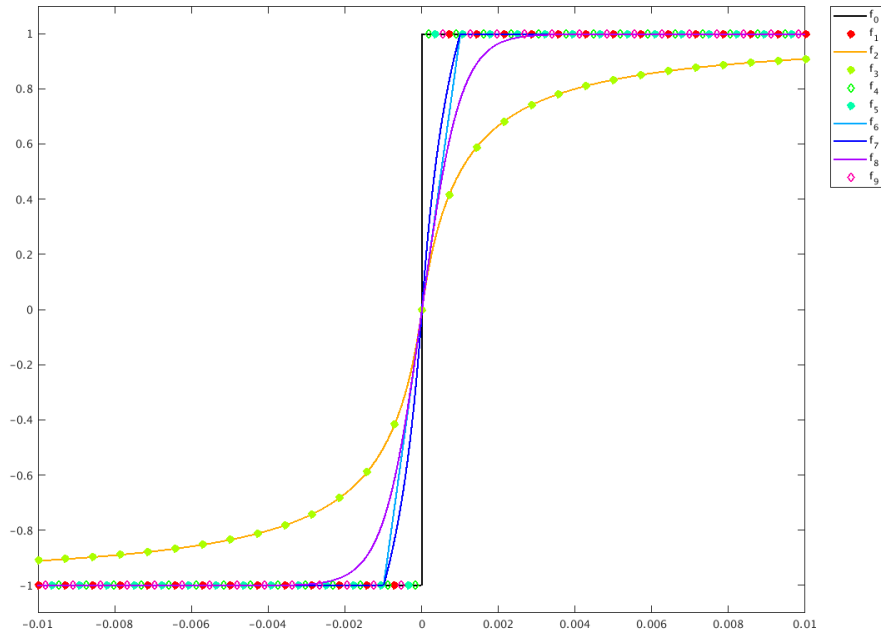


Figure 6.8: Gradients of smoothed approximations to the modulus function.

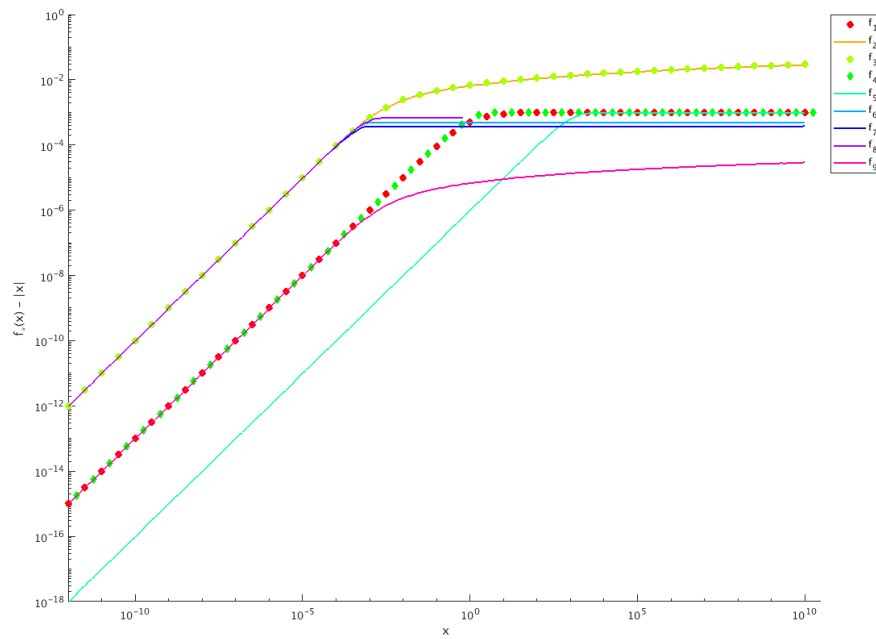


Figure 6.9: The effect of small x values on the difference between $f_{*,\epsilon}$ and $|x|$. Note that $f_{8,\epsilon}$ becomes undefined due to a computational error. When $\frac{x}{\epsilon}$ is large, $\cosh\left(\frac{x}{\epsilon}\right) = \infty$ and so $\log\left(\cosh\left(\frac{x}{\epsilon}\right)\right)$ also evaluates to be ∞ .

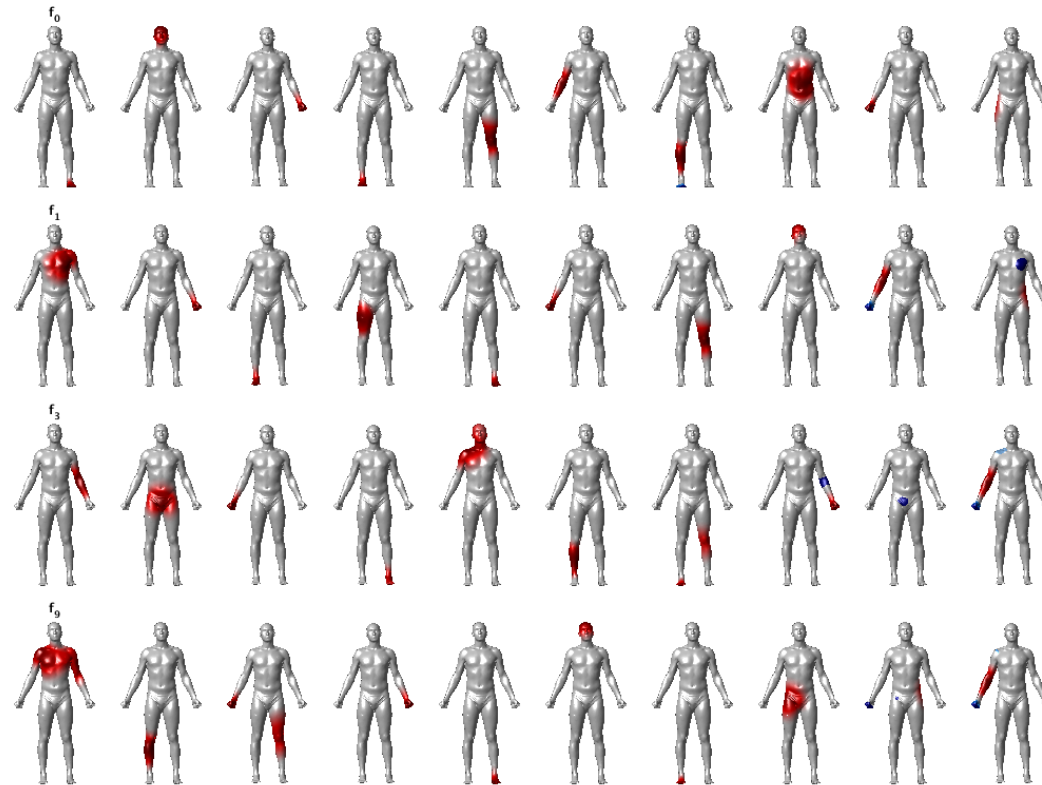


Figure 6.10: Various approximation of CMMs. Algorithm parameters: $\mu = 0.1$, maximum number of iterations = 10000, convergence tolerance = 10^{-12} , uses the canonical metric.

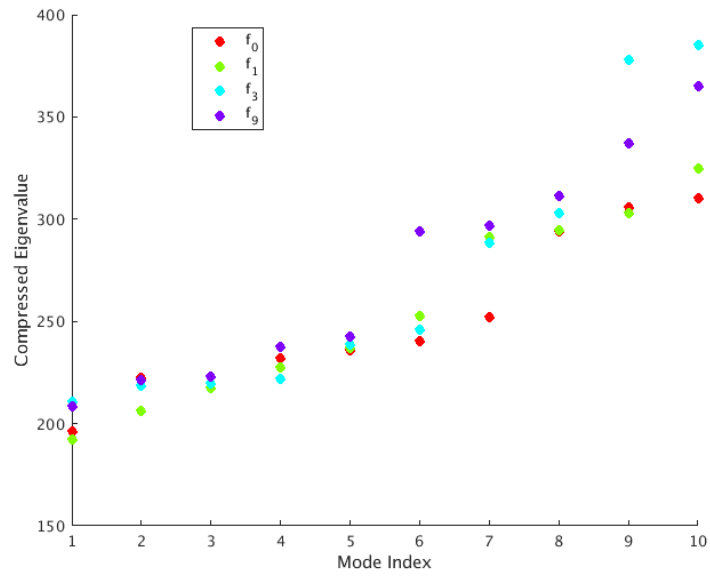


Figure 6.11: Compressed eigenvalues for the modes in figure 6.10

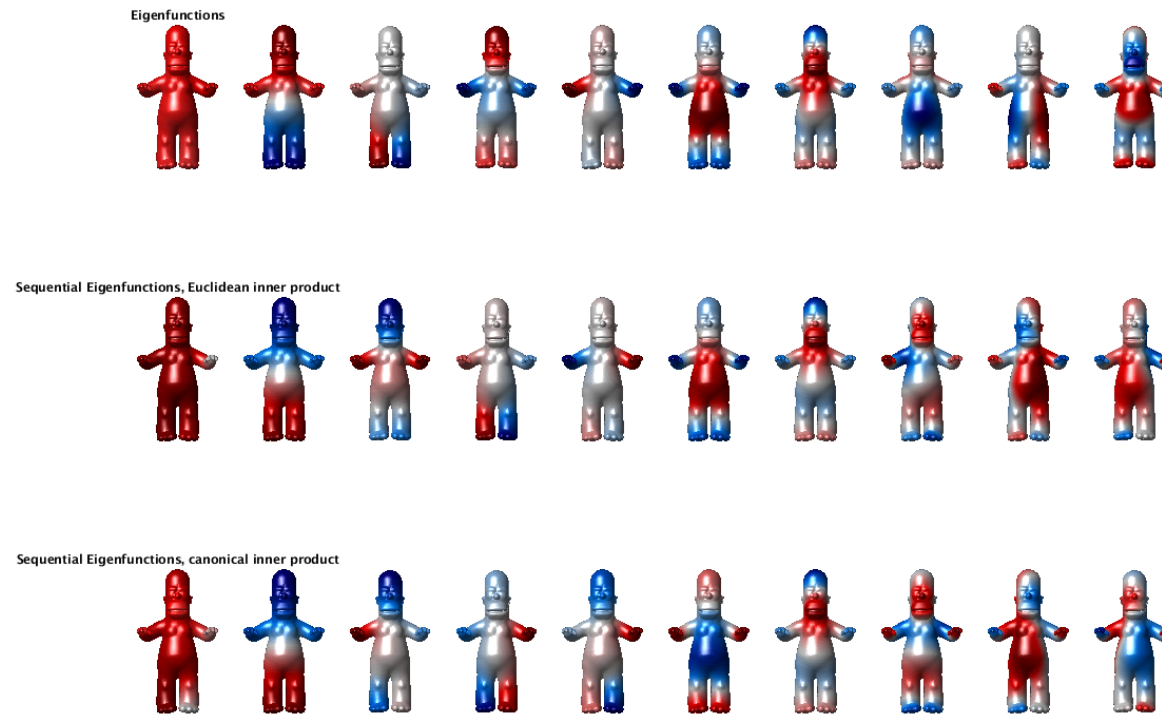


Figure 6.12: A comparison of eigs eigenfunctions and sequential eigenfunctions on the homer mesh. Algorithm parameters: $\mu = 0$, max. iterations = 10000, convergence tolerance = 10^{-12} .

6.6 Summary and Future Work

Generalised manifold harmonics where M is symmetric and invertible can be found as solutions to an optimisation problem on the generalised Stiefel manifold. Section 6.1 developed the theory of Steifel manifolds for the generalised Stiefel manifold case, including a proof that the Cholesky retraction really is a retraction.

A sequential algorithm (algorithm 6) was presented. This gives a potential method for the calculation of generalised manifold harmonics, however it requires an objective function which can be written as a sum of matrix columns.

The sequential algorithm was applied to the calculation of compressed manifold modes. This required a smoothing of the ℓ_1 term and a set of possible smoothings were suggested. The functions produced by the optimisation were sparse, however the results were inconsistent. Different smoothings produced different functions. Further work is required to understand this difference.

Chapter 7

Fast Approximation of Compressed Manifold Modes

The recent paper ‘Fast Approximation of Laplace-Beltrami Eigenfunctions’ [14] provides a method of reducing the dimension of the Laplacian eigenfunctions, by restricting to a sample of the mesh vertices. The solution to the restricted eigenvalue problem can then be used to provide an approximation of the Laplacian eigenbasis. The reduction in dimension leads to faster calculation times, at the expense of loss in accuracy. This chapter provides a summary of the original work and applies the same method to the calculation of compressed manifold modes. Proof that restricted eigenfunctions commute with discrete isometry is also given.

7.1 Fast Approximation of Laplacian Eigenfunctions

This section details the method introduced in the original work referenced above. The general idea is to sample a mesh, construct eigenfunctions on the sample and map back to the mesh to get an approximation of the eigenfunctions on the mesh.

Let N be a mesh with n vertices and real-valued function space $\mathcal{F}(N, \mathbb{R})$, then the

method has three steps:

1. Sample the mesh and produce a set of locally supported functions which span a subspace of the function space.
2. Restrict the eigenproblem to the subspace and solve.
3. Lift the solutions to N , producing an approximation of the Laplacian eigenfunctions.

Constructing a subspace

Let $d \ll n$ be the number of vertices used in a sampling of N . In [14] a sampling method based on Poisson-disk sampling is used which ensures that randomly sampled points remain further than some distance ε from each other [131],[132]. Let $S = \{s_i \in V : i = 1, \dots, d\}$ be the set of vertices in the sample. Then, create a set of locally supported functions \tilde{u}_i centred at s_i . In [14] these functions are defined over the vertex set $\mathcal{V} = \{v_1, \dots, v_n\}$ by

$$\tilde{u}_i(v_j) = \begin{cases} \frac{2}{\rho^3}r^3 - \frac{3}{\rho^2} + 1, & \text{when } r \leq \rho \\ 0, & \text{when } r > \rho \end{cases} \quad (7.1.1)$$

where r is the distance between s_i and v_j , and ρ is a distance which bounds the radius of support of the function. The distance r is calculated using a Euclidean correction of the geodesic distance as calculated via Dijkstra's algorithm: the geodesic distance between two nearby points on a surface can be approximated by the Euclidean distance. The parameter ρ is chosen to ensure that each function is supported on a specified number of other sample vertices. In [14] ρ is chosen so that the local functions \tilde{u}_i are supported on a minimum of 7 and maximum of 15 vertices other than s_i .

The matrix U containing the set of d locally supported functions as columns is constructed by normalising the rows of the matrix \tilde{U} which has the functions u_i as columns. That is,

$$U_{ij} = \frac{1}{\sum_{k=1}^d \tilde{U}_{ik}} \tilde{U}_{ij} = \frac{1}{\sum_{k=1}^d \tilde{u}_k(v_i)} \tilde{u}_j(v_i).$$

This ensures that the sum of the local functions evaluated at each vertex equals 1, constructing a partition of unity. The local functions u_i are defined via

$$u_j(v_i) := U_{ij}.$$

Remark 7.1.1: The combination of poisson-disk sampling and choice of polynomial (equation (7.1.1)) used to construct the local functions guards against the matrix U being rank deficient. However it is not guaranteed that U is full-rank. From here it is assumed that U is full rank.

The subspace eigenproblem

Recall from section 2.5.2 that the eigenfunctions of the Laplacian $L = A^{-1}W$ on N are columns of the matrix solution to the optimisation problem

$$\arg \min_{\Phi} \text{tr}(\Phi^T W \Phi) \quad \text{subject to} \quad \Phi^T A \Phi = I_k,$$

where k is the number of eigenfunctions which are sought. The columns ϕ_i of Φ correspond to eigenvalues λ_i such that $W\phi_i = \lambda_i A\phi_i$.

This eigenproblem can be restricted to the subspace spanned by the functions u_i . This is done by restricting both the weight and area matrices,

$$\bar{A} := U^T A U \quad \text{and} \quad \bar{W} := U^T W U. \quad (7.1.2)$$

The restriction of the eigenproblem to the space spanned by U is then given by

$$\arg \min_{\bar{\phi}} \text{tr}(\bar{\phi}^T \bar{W} \bar{\phi}) \quad \text{subject to} \quad \bar{\phi}^T \bar{A} \bar{\phi} = I_k.$$

The number of eigenfunctions k is chosen to be $k = d/2$. This restricted eigenproblem is called the **subspace eigenproblem**, the eigenfunctions $\bar{\phi}_i$ are called **subspace eigenfunctions** and their corresponding eigenvalues $\bar{\lambda}_i$ are called **restricted eigenvalues**.

The restricted eigenfunctions

The subspace eigenfunctions can be lifted to functions $\bar{\Phi}_i \in \mathbb{R}^n$ on the mesh N by $\bar{\Phi}_i = U\bar{\phi}_i$. These functions are called **restricted eigenfunctions**.

The restricted eigenfunctions have several important properties:

Lemma 7.1.2: [14, lemma 1] The restricted eigenfunctions are orthogonal with respect to A .

Proof. Let $\bar{\Phi}_i = U\bar{\phi}_i$ and $\bar{\Phi}_j = U\bar{\phi}_j$ be restricted eigenfunctions, then

$$\begin{aligned} \langle \bar{\Phi}_i, \bar{\Phi}_j \rangle_A &= \bar{\Phi}_i^T A \bar{\Phi}_j, \\ &= \bar{\phi}_i^T U^T A U \bar{\phi}_j, \text{ by the construction of the restricted eigenfunctions,} \\ &= \bar{\phi}_i^T \bar{A} \bar{\phi}_j, \text{ by the construction of } \bar{A}, \\ &= \delta_{ij}, \text{ since } \bar{\phi}_i^T \bar{A} \bar{\phi}_i = I_k. \end{aligned}$$

□

Lemma 7.1.3: [14, lemma 2] The Dirichlet energy of the restricted eigenfunction $\bar{\Phi}_i$ is given by the restricted eigenvalue $\bar{\lambda}_i$.

Proof. Recall from definition 2.5.32 that the Dirichlet energy of a function represented by vector $\mathbf{f} \in \mathbb{R}^n$ is given by $E = \mathbf{f}^T W \mathbf{f}$. Then the Dirichlet energy of a restricted eigenfunction $\bar{\Phi}_i$ is given by

$$\begin{aligned} \bar{\Phi}_i^T W \bar{\Phi}_i &= \bar{\phi}_i^T U^T W U \bar{\phi}_i, \text{ by the construction of the restricted eigenfunctions,} \\ &= \bar{\phi}_i^T \bar{W} \bar{\phi}_i, \text{ by the construction of } \bar{W}, \\ &= \bar{\lambda}_i \bar{\phi}_i^T \bar{A} \bar{\phi}_i, \text{ by the properties of the subspace eigenfunctions,} \\ &= \bar{\lambda}_i, \text{ since } \bar{\phi}_i^T \bar{A} \bar{\phi}_i = 1. \end{aligned}$$

□

The above lemma provides justification for the terminology *restricted eigenvalues*. Consequently, if restricted eigenvalues $\bar{\lambda}_i$ are close to (unrestricted) eigenvalues

λ_i then the Dirichlet energies of the corresponding restricted and unrestricted eigenfunctions are also close.

Figures 7.2 and 7.1 show a comparison of the fast approximation method applied to the fandisk mesh. A total of 500 restricted eigenfunctions were calculated. Figure 7.3 shows the comparison in time taken to solve the eigenproblem via `eigs` and the time taken to solve the restricted eigenproblem via `eigs`. The total time to perform the fast approximation algorithm is also included – this adds the time taken to sample, construct a matrix of local functions and lift to the restricted eigenfunctions.

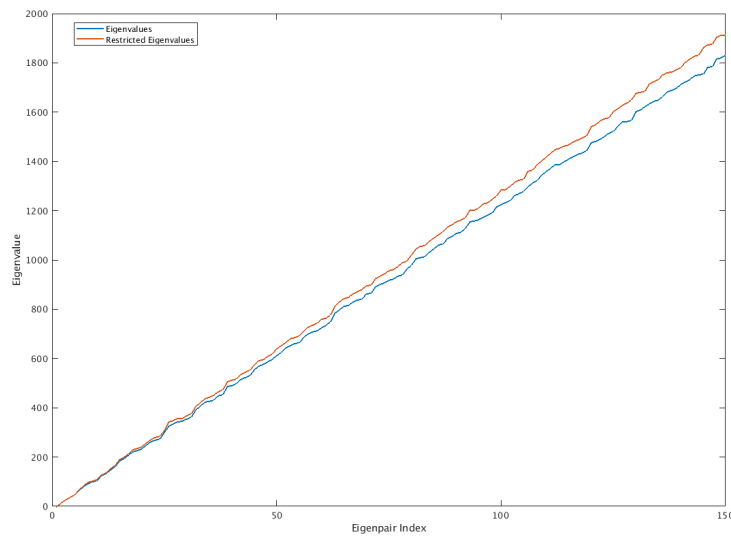


Figure 7.1: A comparison of `eigs` eigenvalues and restricted `eigs` eigenvalues for the fandisk mesh.

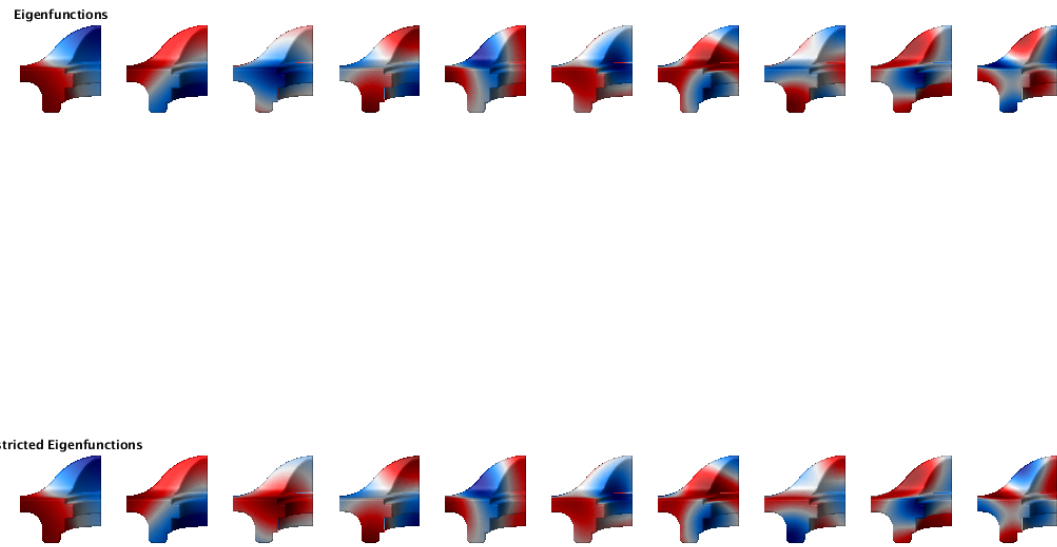


Figure 7.2: A comparison of `eigs` eigenfunctions and restricted `eigs` eigenfunctions on the fandisk mesh.

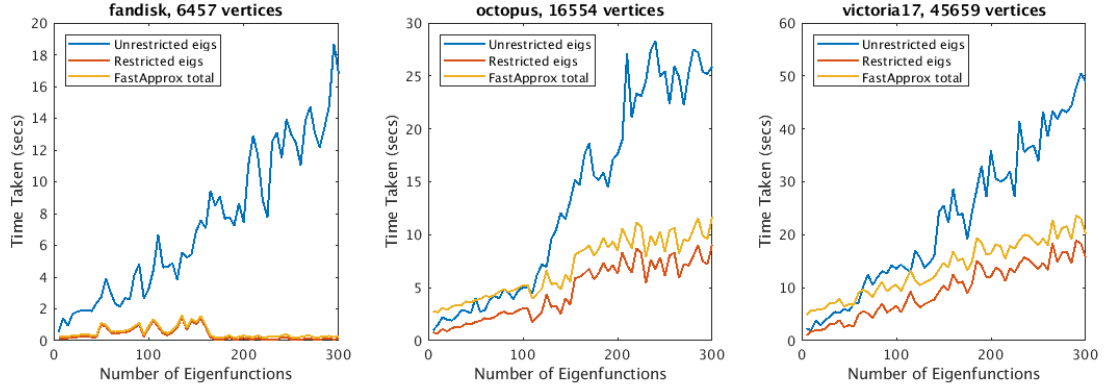


Figure 7.3: A comparison of `eigs` and the fast approximation algorithm timings for three meshes.

Proposition 7.1.4: Restricted eigenfunctions commute with isometry.

Proof. Let Π be the permutation matrix representing an isometry between two meshes N and P . Let U_N be a matrix of locally supported functions. The subspace eigenfunctions $\bar{\Phi}_N$ are such that

$$\bar{\Phi}_N = \arg \min_X \text{tr}(X^T \bar{W}_N X) \quad \text{subject to } X^T \bar{A}_N X = I,$$

where \bar{W}_N and \bar{A}_N are the restricted area and weight matrices.

The isometry maps any function \mathbf{f} defined as a vector on N to the vector $\Pi^T \mathbf{f}$ on P . So $U_P = \Pi^T U_N$. Then, the restricted area and weight matrices on P are given by

$$\begin{aligned} \bar{W}_P &= U_P^T W_P U_P \\ &= U_N^T \Pi \Pi^T W_N \Pi \Pi^T U_N, \text{ since } \Pi \text{ represents an isometry,} \\ &= \bar{W}_N, \end{aligned} \tag{7.1.3}$$

and

$$\bar{A}_P = \bar{A}_N, \tag{7.1.4}$$

by the same reasoning.

The subspace eigenfunctions $\bar{\Phi}_P$ on P are such that

$$\bar{\Phi}_P = \arg \min_X \text{tr}(X^T \bar{W}_P X) \text{ subject to } X^T \bar{A}_P X = I,$$

That is, via equations (7.1.3) and (7.1.4),

$$\bar{\Phi}_P = \arg \min_X \text{tr}(X^T \bar{W}_N X) \text{ subject to } X^T \bar{A}_N X = I,$$

which is exactly the subspace eigenproblem on N . Hence,

$$\bar{\Phi}_P = \bar{\Phi}_N$$

and the restricted eigenvalues on P are given by

$$\bar{\Lambda}_P = \bar{\Lambda}_N.$$

Therefore the restricted eigenfunctions on P are given by

$$\begin{aligned} \Phi_P &= U_P \bar{\Phi}_P \\ &= \Pi^T U_N \bar{\Phi}_N \\ &= \Pi^T \Phi_N. \end{aligned}$$

□

7.2 Extension to GLMHs

The method of fast approximation can be applied to approximating generalised localised manifold harmonics (definition 3.3.9). Recall that these are GMHs which arise as solutions to an eigenvalue problem (as in theorem 3.3.7). As noted in [14], the method adapts because there is no need for the matrices involved to necessarily be the components A, W of a Laplacian $L = A^{-1}W$, as detailed above.

Let M be symmetric and invertible, then problems of the form

$$\arg \min_{\Psi} \text{tr}(\Psi Q \Psi) \text{ subject to } \Psi^T M \Psi = I,$$

Algorithm 7 FastApprox for GLMHs

1: Given GLMH problem of the form

$$\arg \min_{\Psi} \operatorname{tr}(\Psi^T Q \Psi) \quad \text{subject to } \Psi^T M \Psi = I_{k \times k}$$

2: Take a sample and construct matrix of locally supported functions U .

3: Restrict the GLMH problem to

$$\arg \min_X \operatorname{tr}(X^T \bar{Q} X) \quad \text{subject to } X^T \bar{M} X = I_{k \times k}$$

where $\bar{Q} = U^T Q U$ and $\bar{M} = U^T M U$.

4: Solve the restricted GLMH eigenvalue problem.

5: Calculate the restricted eigenfunctions $\bar{\Psi} = U X$.

have solutions given by the eigenfunctions satisfying $(Q + Q^T) \psi_i = \lambda_i M \psi_i$, where the λ_i are the k smallest eigenvalues. These eigenfunctions can be approximated using the fast approximation method, see algorithm 7.

The proof of proposition 7.1.4 can be extended to include generalised localised manifold harmonics with Q and M matrices related by isometry as in theorem 3.5.5.

7.3 Fast Approximation of Compressed Manifold Modes

The idea of solving problems in a lower dimensional subspace can be combined with ADMM to calculate approximations of compressed manifold modes. Recall that the CMM problem is given by

$$\arg \min_{\Psi} \operatorname{tr}(\Psi^T W \Psi) + \mu \|A \Psi\|_1, \quad \text{subject to } \Psi^T A \Psi = I,$$

where $W, A \in \mathbb{R}^{n \times n}$ are the weight and area matrices of a discrete Laplacian, $\Psi \in \mathbb{R}^{n \times k}$ and $\mu \in \mathbb{R}$.

Let matrix U be a matrix of d locally supported functions. Then the CMM problem can be restricted to the subspace spanned by columns of U . The restricted problem is given by

$$\arg \min_X \operatorname{tr}(X^T \bar{W} X) + \mu \|AU X\|_1, \text{ subject to } X^T \bar{A} X = I, \quad (7.3.5)$$

where $\bar{W}, \bar{A} \in \mathbb{R}^{d \times d}$ are as in equation 7.1.2.

Remark 7.3.1: There is no need to scale the sparsity parameter μ as in the restricted problem the ℓ_1 norm continues to act on an $n \times k$ matrix.

Problem (7.3.5) can then be split via the ADMM algorithm of [9], with the $f(X)$ part given by

$$\iota(X) = \begin{cases} 0, & \text{if } X^T \bar{A} X = I \\ \infty, & \text{if } X^T \bar{A} X \neq I. \end{cases}$$

The $g(Z)$ part is given by setting

$$Z = \begin{bmatrix} E \\ S \end{bmatrix}, \quad g(Z) = \begin{bmatrix} \operatorname{tr}(E^T \bar{W} E) \\ \mu \|AS\|_1 \end{bmatrix};$$

and the linear condition is given by

$$\begin{bmatrix} U \\ U \end{bmatrix} X + \begin{bmatrix} -U & 0 \\ 0 & -I \end{bmatrix} \begin{bmatrix} E \\ S \end{bmatrix} = 0.$$

Note that $X \in \mathbb{R}^{d \times k}$, $E \in \mathbb{R}^{d \times k}$ and $S \in \mathbb{R}^{n \times k}$. Given an initial X_0 , set $E_0 = X_0$, $S_0 = UX_0$, $U_0^E \in \mathbb{R}^{n \times k}$ and $U_0^S = 0 \in \mathbb{R}^{n \times k}$. The iterative steps of the algorithm are then given by

$$\text{The } X \text{ step: } X_{k+1} := \arg \min_X \iota(X) + \frac{\rho}{2} \left\| \begin{bmatrix} UX \\ UX \end{bmatrix} - \begin{bmatrix} UE_k \\ S_k \end{bmatrix} + \begin{bmatrix} UU_k^E \\ U_k^S \end{bmatrix} \right\|_F^2, \quad (7.3.6)$$

$$\text{The } E \text{ step: } E_{k+1} := \arg \min_E \operatorname{tr}(E^T \bar{W} E) + \frac{\rho}{2} \|UX_{k+1} - UE + UU_k^E\|_F^2, \quad (7.3.7)$$

$$\text{The } S \text{ step: } S_{k+1} := \arg \min_S \mu \|AS\|_1 + \frac{\rho}{2} \|UX_{k+1} - S + U_k^S\|_F^2, \quad (7.3.8)$$

$$\text{The } U \text{ step: } U_{k+1}^E := U_k^E + X_{k+1} - E_{k+1},$$

$$U_{k+1}^S := U_k^S + UX_{k+1} - S_{k+1}.$$

Although the overall aim is reduce the dimension of the matrices in each step, the S step dimension does not change. This is because a solution is not known when restricting to the lower dimensional space – we only have a result for multiplication by a diagonal matrix (see section 3.2.2, theorem 3.2.13). This leads to the novel situation of the variables U^E and U^S being different sizes.

The X Step: To solve (7.3.6) first rearrange to be of the same form as equation (3.2.20). That is, the problem becomes

$$\arg \min_{X \in \mathbb{R}^{d \times k}} \|UX - Y\|_F^2 \text{ subject to } X^T \bar{A}X = I,$$

where $Y = \frac{1}{2}(UE_k - UU_k^E + S_k - U_k^S)$.

Due to the multiplication by U this cannot be solved in the same way as before – the differentiation of the Lagrange equation with respect to X leads to the equation

$$U^T UX + \frac{1}{2}(\bar{A}X\Lambda + \bar{A}X\Lambda) = U^T Y$$

which can't be rearranged into an expression for X . Instead, approximate a solution by solving for the matrix UX and use least squares minimisation to extract the matrix X . This looks as if it is a problem in $\mathbb{R}^{n \times d}$ but with some manipulation the solution can be written as product of smaller matrices.

Substitute $\hat{X} = UX$ and recall from section 3.2.2, equation 3.2.24 that the solution to

$$\arg \min_{\hat{X} \in \mathbb{R}^{n \times k}} \|\hat{X} - Y\|_F^2 \text{ subject to } \hat{X}^T A \hat{X} = I,$$

when $A = rI$ is given by

$$\hat{X} = \frac{1}{\sqrt{r}} YVD^{-\frac{1}{2}}V^T \quad (7.3.9)$$

where Y is as above and VDV^T is the singular value decomposition of $Y^T Y$.

But, $S_k - U_k^S$ can be approximated in the subspace spanned by the columns of U . Let $\tilde{S} \in \mathbb{R}^{d \times k}$ assume that $U\tilde{S} = S_k - U_k^S$, then

$$\begin{aligned} U^T U \tilde{S} &= U^T (S_k - U_k^S) \\ \tilde{S} &= (U^T U)^{-1} U^T (S_k - U_k^S). \end{aligned}$$

Of course, this is not really equality, but the least squares solution of section 2.4.1. Note that the existence of $(U^T U)^{-1}$ follows from the assumption that U is full rank. Then let Y be approximated by \tilde{Y} with

$$\begin{aligned}\tilde{Y} &:= \frac{1}{2}(U E_k - U U_k^E + U \tilde{S}) \\ &= \frac{1}{2}U(E_k - U_k^E + \tilde{S}).\end{aligned}$$

Then

$$\tilde{Y}^T \tilde{Y} = \frac{1}{4}(E_k - U_k^E + \tilde{S})^T U^T U (E_k - U_k^E + \tilde{S}).$$

Let $\tilde{V} \tilde{D} \tilde{V}^T$ be the svd of the $k \times k$ matrix $(E_k - U_k^E + \tilde{S})^T U^T U (E_k - U_k^E + \tilde{S})$, so

$$\tilde{Y}^T \tilde{Y} = \frac{1}{4} \tilde{V} \tilde{D}^{\frac{1}{2}} \tilde{V}^T$$

and

$$(\tilde{Y}^T \tilde{Y})^{-\frac{1}{2}} = 2 \tilde{V} \tilde{D}^{-\frac{1}{2}} \tilde{V}^T.$$

Then, via equation (7.3.9), UX can be approximated by

$$\begin{aligned}UX &\approx \frac{2}{\sqrt{r}} \tilde{Y} \tilde{V} \tilde{D}^{-\frac{1}{2}} \tilde{V}^T \\ &= \frac{1}{\sqrt{r}} U(E_k - U_k^E + \tilde{S}) \tilde{V} \tilde{D}^{-\frac{1}{2}} \tilde{V}^T.\end{aligned}$$

Using the least squares minimisation discussed above, X can be approximated by multiplying on the left by $(U^T U)^{-1} U^T$, giving

$$\begin{aligned}X &\approx \frac{1}{\sqrt{r}} (U^T U)^{-1} U^T U (E_k - U_k^E + \tilde{S}) \tilde{V} \tilde{D}^{-\frac{1}{2}} \tilde{V}^T \\ &= \frac{1}{\sqrt{r}} (E_k - U_k^E + \tilde{S}) \tilde{V} \tilde{D}^{-\frac{1}{2}} \tilde{V}^T.\end{aligned}\tag{7.3.10}$$

The E Step: Problem (7.3.7) can be solved in a similar way to problem (3.2.16).

Let $\hat{E} := UX_{k+1} + UU_k^E$, then

$$\begin{aligned}E_k &= \arg \min_E \operatorname{tr}(E^T W E) + \frac{\rho}{2} \|\hat{E} - U E\|_F^2 \\ &= \arg \min_E \operatorname{tr}(E^T W E) + \frac{\rho}{2} \left(\operatorname{tr}(\hat{E}^T \hat{E}) + \operatorname{tr}(E^T U^T U E) - 2 \operatorname{tr}(\hat{E}^T U E) \right).\end{aligned}$$

Differentiating with respect to E and equating with zero gives

$$0 = 2WE + \frac{\rho}{2} \left(2U^T U E - 2U^T \hat{E} \right).$$

Rearranging gives

$$(2W + \rho U^T U)E = \rho U^T \hat{E}.$$

And so

$$\begin{aligned} E &= \rho(2\bar{W} + \rho U^T U)^{-1} U^T (UX_{k+1} + UU_k^E) \\ &= \rho(2\bar{W} + \rho U^T U)^{-1} U^T U (X_{k+1} + U_k^E). \end{aligned} \quad (7.3.11)$$

That $(2\bar{W} + \rho U^T U)^{-1}$ exists is due to it being the sum of a positive semi-definite matrix and a positive definite matrix (recall $\rho > 0$).

This is the step which is most greatly affected by the lowering of dimensions. In the original ADMM for calculating CMMs the E step requires the inversion of an $n \times n$ matrix. Here the size of the matrix inverse is $d \times d$, where d is the number of samples taken to construct the set of local basis functions. It is assumed that $d \ll n$, and so the time taken to calculate the inverse will be reduced. Note, however the additional multiplication by $U^T U$ and that in the original ADMM solution E is given by

$$E = \rho(2W + \rho I)^{-1} (\Psi_{k+1} + U_k^E),$$

where the matrix $2W + \rho I$ is much sparser than the matrix $2\bar{W} + \rho U^T U$. Figure 7.4 compares the sparsity of the restricted and unrestricted matrices. Sparsity is given as (number of zero elements/number of elements).

The S Step: The solution for S can be found in the same way as previously, with $\hat{S} = UX_{k+1} + U_k^S$. That is,

$$S_{ij} = \text{sgn}(\hat{S}_{ij}) \max\{|\hat{S}_{ij}| - \frac{\mu r}{\rho}, 0\}, \quad (7.3.12)$$

since it is assumed that $A = rI$. Again note the the dimensions have not been reduced when compared with the original ADMM.

To summarise, the algorithm for the fast approximation of compressed manifold modes is given in algorithm 8.

Algorithm 8 Fast Approximation of CMMs

- 1: Given $X_0 \in \mathbb{R}^{d \times k}$, $\mu \in \mathbb{R}$, restricted weight matrix $\bar{W} \in \mathbb{R}^{d \times d}$, matrix of local functions $U \in \mathbb{R}^{n \times d}$
- 2: Set $E_0 = X_0$, $S_0 = UX_0$, $U_0^E = 0 \in \mathbb{R}^{d \times k}$, $U_0^S = 0 \in \mathbb{R}^{n \times k}$
- 3: Set regularisation parameter $\rho > 0$
- 4: Compute and store $U^T U$, $(U^T U)^{-1} U^T$
- 5: **repeat**

$$\tilde{S} \leftarrow (U^T U)^{-1} U^T (S_k - U_k^S)$$

$$Z \leftarrow E_k - U_k^E - \tilde{S}$$

$$\tilde{V} \tilde{D} \tilde{V}^T = Z^T U^T U Z, \text{ the svd}$$

$$X_{k+1} \leftarrow \sqrt{n} Z \tilde{V} \tilde{D}^{-\frac{1}{2}} \tilde{V}^T$$

$$E_{k+1} \leftarrow \rho (2\bar{W} + \rho U^T U)^{-1} U^T U (X_{k+1} + U_k^E)$$

$$(S_{k+1})_{ij} \leftarrow \text{sgn}((UX_{k+1} + U_k^S)_{ij}) \max\{|(UX_{k+1} + U_k^S)_{ij}| - \frac{\mu}{\rho n}, 0\}$$

$$U_{k+1}^E \leftarrow U_k^E + X_{k+1} - E_{k+1}$$

$$U_{k+1}^S \leftarrow U_k^S + UX_{k+1} - S_{k+1}$$

- 6: **until** convergence
-

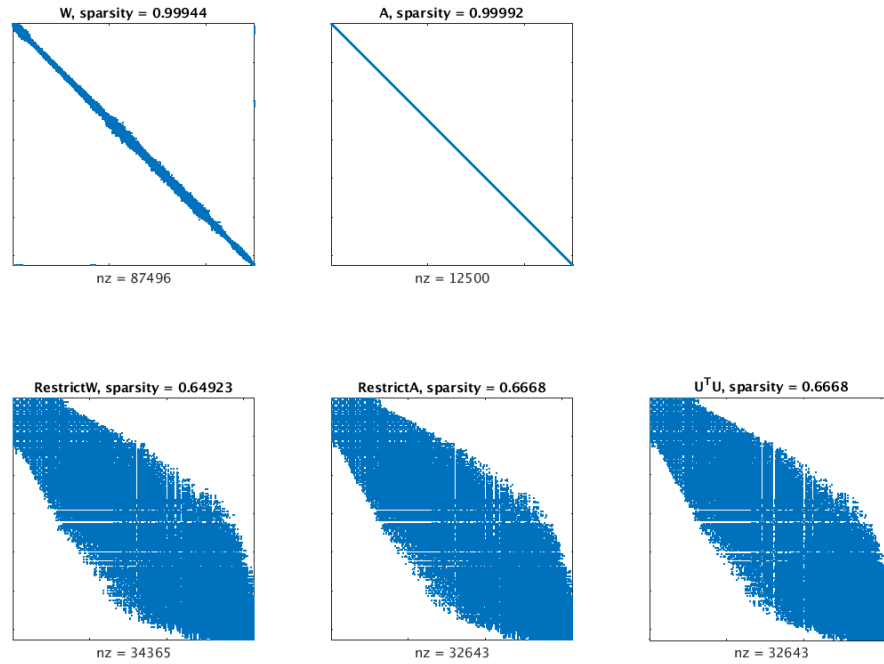


Figure 7.4: The sparsity of the original and restricted area and weight matrices, and $U^T U$, calculated for the ScapeMan001 mesh.

7.3.1 Experimental Results

Compressed manifold modes were calculated for a set of meshes, using both the fast approximation algorithm and ADMM without restriction. CMMs were calculated 10 times then timings averaged, to allow for deviation due to the the random initialisation. Table 7.1 provides details of the sparsity parameter, sample sizes, overall timings and number of iterations required for convergence. The time taken to restrict the problem includes the time taken to sample the mesh, construct the matrix U and calculate the restricted matrices \bar{W} , \bar{A} . It is likely that the time taken to restrict the problem could be improved. The final column calculates an average time per iteration for each method.

These timings show that although the time per iteration of the fast approximation

method is greater than the time per iteration of the unrestricted ADMM, the fast approximation converges much faster. Note the trends that as the number of vertices in the mesh increases the number of iterations required for the unrestricted ADMM to converge increases greatly but the number of iterations required for the fast approximation to converge does not show such a trend, the number of iterations seems to be quite steady.

Table 7.2 compares the timings for the individual steps of the ADMM algorithm. The time given is the average time per iteration where the number of iterations and total times are given in table 7.1. As expected the greatest saving in time taken is in the E step. This is the step where the dimension of the matrix inverse is reduced from an $n \times n$ matrix to a $d \times d$ matrix. The Ψ/X step is significantly slower in the fast approximation, this is likely to be due to the multiplication by $(U^T U)^{-1} U^T$ required to calculate \tilde{S} . Unexpectedly the timings for the S step also increased – probably due to a change in matrix sparsity, a consequence of restricting via U .

Figure 7.5 shows CMMs calculated via the fast approximation and via unrestricted ADMM on the fandisk and ScapeMan001 meshes. The functions are very similar, some change in ordering.

Mesh	Vertices	Modes	μ	ADMM		Sample Size	Restrict time	Fast Approx.			Av. ADMM It. Time	
				time	it.s			time	it.s	total	ADMM	Fast
fandisk	6457	10	64	53.2174	1489	162	0.9057	6.1890	326	7.1082	0.0357	0.0190
chair1	12326	10	120	75.9129	1424	308	3.5430	29.9082	413	33.5672	0.0533	0.0725
ScapeMan001	12500	10	120	31.7624	676	313	3.0367	13.8918	214	17.0057	0.0470	0.0649
octopus	16554	10	160	111.9092	1851	413	6.5580	34.6992	267	41.8057	0.0605	0.1301
horse0	19248	10	190	69.4587	868	481	7.7546	61.1408	405	69.1317	0.0801	0.1509
horse10	19248	10	190	83.5128	1065	481	7.1619	73.8106	509	81.2081	0.0784	0.1451
head1	20490	10	200	335.8071	1134	512	8.2724	47.7540	300	56.1544	0.2963	0.1593
cat6	27894	10	270	124.0095	1070	697	13.7826	86.6193	356	100.4732	0.1159	0.2432
victoria17	45659	10	46	1013.7910	5278	1141	39.0662	229.2959	368	268.6111	0.1921	0.6234
david12	52565	10	53	1126.6470	4296	1312	51.5346	325.2257	391	377.0151	0.2622	0.8326
kid7	59727	10	60	1767.3510	6241	1493	66.9091	390.4531	365	457.7433	0.2832	1.0706

Table 7.1: Table comparing ADMM with Fast Approximation for CMMs on assorted meshes.

Mesh	ADMM			Fast Approx.			% Increase		
	Ψ Step	E Step	S Step	X Step	E Step	S Step	$\Psi \setminus X$ Step	E Step	S Step
fandisk	0.0019	0.0298	0.0005	0.0106	0.0016	0.0016	5.520	0.0525	3.3113
chair1	0.0018	0.0447	0.0007	0.0379	0.0046	0.0075	21.566	0.1021	10.0042
ScapeMan001	0.0019	0.0385	0.0007	0.0392	0.0040	0.0051	20.441	0.1027	7.2747
octopus	0.0032	0.0484	0.0011	0.0703	0.0074	0.0133	21.688	0.1521	12.0973
horse0	0.0034	0.0667	0.0013	0.0931	0.0093	0.0109	27.569	0.1389	8.5095
horse10	0.0035	0.0650	0.0012	0.0894	0.0098	0.0106	25.597	0.1500	8.5426
head1	0.0039	0.2814	0.0013	0.1044	0.0089	0.0095	26.578	0.0317	7.2225
cat6	0.0042	0.0983	0.0017	0.1923	0.0089	0.0074	45.580	0.0907	4.3865
victoria17	0.0074	0.1626	0.0027	0.5082	0.0201	0.0164	68.576	0.1235	6.1293
david12	0.0095	0.2164	0.0039	0.6842	0.0245	0.0205	71.848	0.1132	5.2943
kid7	0.0102	0.2339	0.0042	0.8668	0.0325	0.0254	85.378	0.1390	6.0185

Table 7.2: Table comparing the unrestricted ADMM and fast approximation step times.

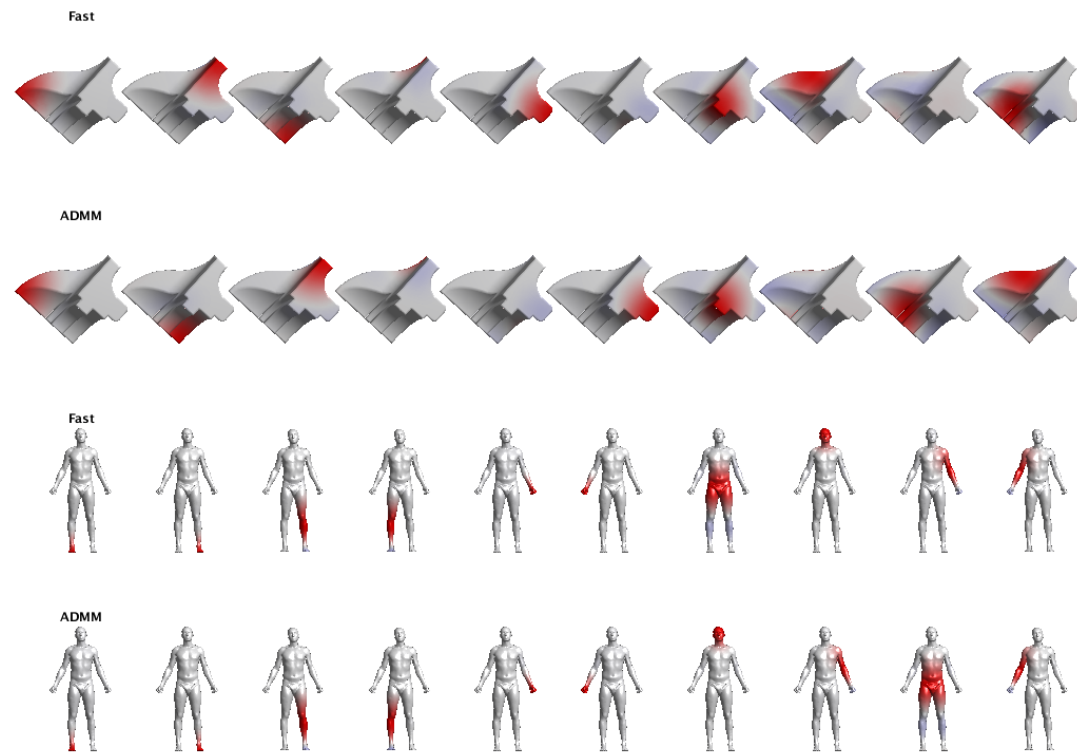


Figure 7.5: CMMs calculated via fast approximation and unrestricted ADMM, on the fan disk and ScapeMan001 meshes.

7.4 Summary and Future Work

A novel algorithm for the fast approximation of compressed manifold modes was given (algorithm 8). This algorithm show significant improvement to the time taken to calculate CMMs.

The algorithm needs to be tested on larger meshes and the effect of the sample size could be investigated.

The algorithm for adapting the fast approximation method to any eigenproblem is simple. As mentioned in section 7.2 the adaptation of the fast approximation method to the calculation of localised manifold harmonics should follow. In practice, however, this requires future work. An attempt was made but the resulting functions were not consistent with the `eigs` eigenfunctions, and hence not explored further or detailed in the chapter. The cause behind the inconsistency is that the localised functions used to lower the dimensional of the eigenvalue problem reduce the ability to construct locally supported functions with a high frequency. A naive solution is to simply increase the number of sampled points, increasing the dimension of the subspace eigenproblem. Instead, we suggest a stratified sample, with a dense sampling of the regions of localisation, and a sparser sampling of the rest of the mesh.

To formulate the restricted ADMM problem the variables U^E and U^S are of different sizes. This has not been done before and requires further work to formally describe the effect on the convergence conditions.

Chapter 8

Key Results and Conclusions

Generalised manifold harmonics have been introduced here as solutions to a general problem with an orthogonality constraint. The problem brings together definitions of various functions used in geometry processing including the Laplacian eigenfunctions, localised manifold harmonics, Hamiltonian eigenfunctions and compressed manifold modes. A new definition for discrete isometry was given, motivated by properties of isometries between Riemannian manifolds. This led to a proof that compressed manifold modes commute with isometry. No such result was previously known. The conditions required for generalised manifold harmonics to commute with isometry were also discussed.

KEY RESULT: Localised manifold harmonics and compressed manifold modes commute with discrete isometry.

Generalised manifold harmonics can be used to span subspaces of a mesh function space. A variety of alternative bases were tested for their ability to reconstruct functions. Localised manifold harmonics are designed to improve the reconstruction of a specific function (or set of functions). It was expected that they would out-perform alternative bases when reconstructing the functions which they were constructed to handle.

KEY RESULT: Function type has a greater impact on reconstruction error than basis type.

KEY RESULT: Localised manifold harmonics did not perform as well as expected.

Functional maps are used as a method of finding point-to-point matches between shapes. They can also be used to transfer functions between meshes. The basis vectors tested for function reconstruction were also used in the construction of functional maps. The functional maps were used to transform functions from one mesh to another. In a first attempt to consider the quality of such transformations on function reconstruction the error between the mapped functions and the known corresponding functions was measured. Results showed no particular trends, except that some bases performed better for some specific meshes than others. This may mean that basis choice can be optimised for specific shape collections, however it may be a side effect of the meshes having the same underlying graph structure. How to test this and the testing of transformation of different functions provide scope for future work.

A specific set of generalised manifold harmonics can be calculated as solutions to an eigenvalue problem, this includes localised manifold harmonics and Hamiltonian eigenfunctions. Compressed manifold modes and other generalised manifold harmonics cannot be found in such a way. Given that they are solutions to an optimisation problem with the constraint $X^T M X = I$ optimisation on generalised Stiefel manifold was studied. Known properties and results for the Stiefel manifold $\mathcal{S} = \{X \in \mathbb{R}^{n \times k} : X^T M X = I, M = I\}$ were generalised for the $M \neq I$ case. A sequential algorithm for optimising on generalised Stiefel manifolds was given. The sequential method requires an objective function $F : \mathbb{R}^{n \times k} \rightarrow \mathbb{R}$ which can be written as a sum of functions acting on the matrix columns; it may be possible to use the same method to optimise alternative functions with the additional of further constraints.

KEY RESULT: Section 6.1 which provides the properties of generalised Stiefel manifolds.

KEY RESULT: A sequential algorithm for optimisation on generalised Stiefel manifolds using the periodic retraction, algorithm 6.

The method of fast approximation of Laplacian eigenfunctions provides a way of approximating Laplacian eigenfunctions by restricting the eigenvalue problem to a subspace. It was proved that the restricted eigenfunctions commute with isometry. Due to the ease of calculating approximated eigenfunctions this could be useful in shape matching problems, but would require corresponding matrices of locally supported functions on each mesh. The method is simply generalised for a specific set of generalised manifold harmonics – those which are solutions to eigenvalue problems. When attempting to use this generalisation to approximate localised manifold harmonics and Hamiltonian eigenfunctions, problems arise due to the mesh sampling. This requires further exploration – a possible solution is to use a mixed-density sample which takes more samples of the local regions of interest. The method was applied to the approximation of compressed manifold modes. Numerical results show a reduction in the number of iterations required to get the ADMM algorithm to converge, and a significant time saving in the E step due to the reduction of the dimension of the matrix inverse required. Further work is required to improve the X step.

KEY RESULT: Restricted eigenfunctions commute with isometry.

KEY RESULT: A novel algorithm for the fast calculation of approximations of compressed manifold modes, algorithm 8.

The sparsity parameter μ in the compressed manifold modes problem remains difficult to choose. It was observed that for some meshes the compressed manifold modes and Laplacian eigenfunctions spanned similar subspaces of mesh function space. If it can be understood why then this could lead to greater understanding of how μ choice affects modes.

Appendices

A Assorted Short Proofs

Lemma A.1: Let M be a full rank matrix. Then the matrix $M^T M$ is invertible.

Proof. Recall that any square matrix with null space equal to zero is invertible. Let x be such that $M^T M x = 0$, then

$$\begin{aligned} x^T M^T M x &= 0 \\ (Mx)^T (Mx) &= 0, \end{aligned}$$

and so Mx must be equal to the zero vector. As M is full rank this means that $x = 0$, and so the null space of $M^T M$ is zero. Hence, $M^T M$ is invertible. \square

Lemma A.2: Let M and N be $n \times n$ symmetric matrices, with M invertible. Then the linear operator represented by $M^{-1}N$ is self-adjoint, with respect to the M inner product.

Proof. Let f, g be vectors, then

$$\begin{aligned} \langle M^{-1}Nf, g \rangle_M &= f^T N^T M^{-T} M g \\ &= f^T N M^{-1} M g, \text{ since } N = N^T, M^{-T} = M^{-1}, \\ &= f^T N g \\ &= f^T M M^{-1} N g \\ &= \langle f, M^{-1}N g \rangle_M \end{aligned}$$

and so $M^{-1}N$ is self-adjoint. \square

Lemma A.3: Let L be self-adjoint with respect to the A inner product, then L has orthogonal eigenvectors and real eigenvalues.

Proof. First consider the eigenvalues. Let $\lambda \in \mathbb{C}$ be an eigenvalue for L , with

eigenvector v . Then

$$\begin{aligned}
 \lambda \langle v, v \rangle_A &= \langle \lambda v, v \rangle_A \\
 &= \langle Lv, v \rangle_A, \text{ since } Lv = \lambda v, \\
 &= \langle v, Lv \rangle_A, \text{ since } L \text{ is self-adjoint,} \\
 &= \langle v, \lambda v \rangle_A \\
 &= \overline{\langle \lambda v, v \rangle_A}, \text{ since inner products obey } \langle x, y \rangle = \overline{\langle y, x \rangle}, \\
 &= \bar{\lambda} \overline{\langle v, v \rangle_A} \\
 &= \bar{\lambda} \langle v, v \rangle_A, \text{ since } \overline{\langle v, v \rangle} = \langle v, v \rangle.
 \end{aligned}$$

Therefore, $\lambda = \bar{\lambda}$ which implies $\lambda \in \mathbb{R}$. Now consider the eigenfunctions: Let λ be an eigenvalue for L with eigenvector v , and let μ be an eigenvalue with eigenvector w , $\lambda \neq \mu$. Then

$$\begin{aligned}
 \langle Lv, w \rangle_A &= \langle v, Lw \rangle_A, \text{ since } L \text{ is self-adjoint,} \\
 \langle \lambda v, w \rangle_A &= \langle v, \mu w \rangle_A \\
 \lambda \langle v, w \rangle_A &= \mu \langle v, w \rangle_A \\
 (\lambda - \mu) \langle v, w \rangle_A &= 0
 \end{aligned}$$

but since $\lambda \neq \mu$, $\lambda - \mu \neq 0$ and hence, $\langle v, w \rangle_A = 0$. □

Lemma A.4: Let M be an $n \times n$ symmetric positive definite matrix. Let (pq) be such that $M_{pq} \geq |M_{ij}|$ for all pairs $(i, j) \neq (p, q)$. Then $p = q$. That is, the largest element of M lies on the diagonal.

Proof. Let e_i and e_j be standard basis vectors in \mathbb{R}^n . Define $x := e_i - e_j$ and consider $x^T M x$:

$$\begin{aligned}
 x^T M x &= e_i^T M e_i + e_j^T M e_j - 2e_i^T M e_j \\
 &= M_{ii} + M_{jj} - 2M_{ij}.
 \end{aligned}$$

As M is positive definite and $x \neq 0$ it must be that

$$M_{ii} + M_{jj} > 2M_{ij}$$

and hence at least one of M_{ii}, M_{jj} must be greater than M_{ij} . This holds for all i, j and so it must be that the largest element of M lies on the diagonal. \square

Lemma A.5: An $n \times n$ matrix with all entries equal is not positive definite ($n \geq 2$).

Proof. Let $M \in \mathbb{R}^{n \times n}$ with $M_{ij} = m \in \mathbb{R}$ for all ij and let $x \in \mathbb{R}^n$. Then $x^T M x = m \sum_{ij} x_i x_j$. Consider the cases:

- (i) If $m = 0$ then $x^T M x = 0$ for all x , and hence M is not positive definite.
- (ii) Let x be such that one element is equal to 1, one element is equal to -1 and all other elements are equal to zero. Then $\sum_{ij} x_i x_j = 0$, and so $x^T M x = 0$. As $x \neq 0$ the matrix M is not positive definite.

\square

Lemma A.6: Let $A, B \in \mathbb{R}^{n \times k}$, then $\|A + B\|_F^2 = \|A\|_F^2 + \|B\|_F^2 + 2 \operatorname{tr}(A^T B)$.

Proof.

$$\begin{aligned} \|A + B\|_F^2 &= \operatorname{tr}((A^T + B^T)(A + B)) \\ &= \operatorname{tr}(A^T A + B^T A + A^T B + B^T B) \\ &= \operatorname{tr}(A^T A) + \operatorname{tr}(B^T A) + \operatorname{tr}(A^T B) + \operatorname{tr}(B^T B) \\ &= \|A\|_F^2 + \|B\|_F^2 + 2 \operatorname{tr}(A^T B), \text{ since } \operatorname{tr}(X) = \operatorname{tr}(X^T). \end{aligned}$$

\square

Lemma A.7: Let $A \in \mathbb{R}^{a \times k}$, $B \in \mathbb{R}^{b \times k}$, then $\left\| \begin{bmatrix} A \\ B \end{bmatrix} \right\|_F^2 = \|A\|_F^2 + \|B\|_F^2$.

Proof.

$$\begin{aligned} \left\| \begin{bmatrix} A \\ B \end{bmatrix} \right\|_F^2 &= \operatorname{tr} \left(\begin{bmatrix} A^T & B^T \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} \right) \\ &= \operatorname{tr}(A^T A + B^T B) \\ &= \operatorname{tr}(A^T A) + \operatorname{tr}(B^T B) \\ &= \|A\|_F^2 + \|B\|_F^2. \end{aligned}$$

□

Lemma A.8: $\|X\|_1 = \text{vec}(X)^T \text{vec}(\text{sgn}(X))$.

Proof.

$$\begin{aligned} \text{vec}(X)^T \text{vec}(\text{sgn}(X)) &= \sum_{ij} X_{ij} \text{sgn}(X) \\ &= \sum_{ij} |X_{ij}| \\ &= \|X\|_1. \end{aligned}$$

□

Lemma A.9: Let $\mathcal{L}(X, \Lambda)$ be the Lagrangian multiplier function where Λ is the matrix of λ_{ij} associated to the equations $\phi_i^T M \phi_j - \delta_{ij} = 0$. Then, if M is symmetric so is Λ .

Proof. If M is symmetric then $(\phi_i^T M^T \phi_j)^T = \phi_i^T M \phi_j$, which is equal to $\phi_i^T M^T \phi_j$ since the product is a scalar. Then, since $\delta_{ij} = \delta_{ji}$, the equations $\phi_i^T M \phi_j - \delta_{ij} = 0$ and $\phi_j^T M \phi_i - \delta_{ji} = 0$ are equivalent, and are, therefore, associated to the same Lagrangian variable λ_{ij} . This results in the matrix Λ where $\lambda_{ij} = \lambda_{ji}$; that is, Λ is symmetric. □

Lemma A.10: Let $X \in \mathbb{R}^{n \times k}$ and let $U \in \mathbb{R}^{m \times n}$ such that $U^T U = I$. Then $\|UX\|_F = \|X\|_F$.

Proof.

$$\begin{aligned} \|UX\|_F^2 &= \text{tr}((UX)^T (UX)) \\ &= \text{tr}(X^T U^T UX) \\ &= \text{tr}(X^T X) \\ &= \|X\|_F^2 \end{aligned}$$

□

Lemma A.11: Let $\Psi \in \mathbb{R}^{n \times k}$ be such that $\Psi^T A \Psi = I$ for symmetric positive definite A . Define $f := \Psi a$ and $g := \Psi b$ via $a, b \in \mathbb{R}^k$. Then

$$\|f - g\|_A = \|a - b\|_2.$$

Proof.

$$\begin{aligned} \|f - g\|_A^2 &= (f - g)^T A (f - g) \\ &= f^T A f + g^T A g - 2f^T A g \\ &= a^T a + b^T b - 2a^T b, \text{ since } \Psi^T A \Psi = I, \\ &= (a - b)^T (a - b) \\ &= \|a - b\|_2^2 \end{aligned}$$

Taking the square root of both sides gives the result. \square

Lemma A.12: Let $S \in \text{Sym}(n)$ and let $K \in \text{Skew}(n)$. Then $\text{tr}(SK) = 0$.

Proof. Recall that the trace of any matrix is equal to the trace of the transpose, so

$$\begin{aligned} \text{tr}(SK) &= \text{tr}(K^T S^T) \\ &= \text{tr}(-KS), \text{ by the properties of } S \text{ and } K, \\ &= -\text{tr}(KS). \end{aligned}$$

Also recall that the trace is invariant under cyclic permutation, so

$$\text{tr}(SK) = \text{tr}(KS).$$

Therefore, since $\text{tr}(KS) = -\text{tr}(KS)$ the trace must be zero. \square

Lemma A.13: Let A be an $a \times n$ matrix with $\text{rank}(A) = n$, and let B be an $n \times b$ matrix. Then $\text{Rank}(AB) = \text{Rank}(B)$.

Proof. From the definition of rank

$$\begin{aligned} \text{Rank}(AB) &= \dim(\{y \in \mathbb{R}^a : y = ABx, x \in \mathbb{R}^b\}) \\ &= \dim(\{y \in \mathbb{R}^a : y = Az, z \in \text{Im}(B) \subseteq \mathbb{R}^n\}). \end{aligned}$$

First note that the kernel of AB is contained within the kernel of A , so $\text{Null}(AB) \leq \text{Null}(A)$. The nullity of A is given by

$$\text{Null}(A) = \dim(\{y \in \mathbb{R}^n : Ay = 0\}) = 0,$$

since $\text{Rank } A = n$.

Define a transformation $T : \text{Im}(B) \rightarrow \mathbb{R}^a$ such that

$$T(z) = Az \text{ for all } z \in \text{Im}(B),$$

represented by matrix \bar{A} .

By the rank-nullity theorem $\text{Rank}(\bar{A}) + \text{Null}(\bar{A}) = \text{Rank}(B)$. Note that $\text{Rank}(AB) = \text{Rank}(\bar{A})$ and that $\text{Null}(AB) = \text{Null}(\bar{A})$. Therefore

$$\begin{aligned} \text{Rank}(AB) &= \text{Rank}(\bar{A}) \\ &= \text{Rank}(B) - \text{Null}(\bar{A}) \\ &= \text{Rank}(B) - \text{Null}(AB) \\ &= \text{Rank}(B). \end{aligned}$$

□

Lemma A.14: Let $\rho \in \mathbb{R}$, $\rho > 0$ and let $\bar{W} = U^T W U$, with U full rank and W positive semi-definite. Then $\bar{W} + \rho U^T U$ is invertible.












Proof. Let x be a non-zero vector, then














$$\begin{aligned} x^T (\bar{W} + \rho U^T U) x &= x^T U^T (W + \rho I) U x \\ &= y^T (W + \rho I) y, \text{ where } y = U x \\ &= y^T W y + \rho y^T y. \end{aligned}$$






Since U is full rank and $x \neq 0$, $y \neq 0$. Therefore, since W is positive semi-definite and $y^T y > 0$, the matrix $\bar{W} + \rho U^T U$ is positive definite. Hence it is invertible. □

B Table of Mesh Details

Table B.1: Mesh details.

Mesh name	Vertices	Source/Comments
 screwdriver 1 †	2502	aim@shape [133]
 screwdriver 2 †	2502	Deformation of screwdriver 1
 screwdriver 2 †	2502	Deformation of screwdriver 1
 homer †	5103	SHREC '12 [134]
 fandisk †	6457	yobi3d.com [135]
 table 1 †	10182	SHREC '12
 chair 1 †	12326	SHREC '12
 ScapeMan001	12500	SCAPE [136]
 chair 2 †	13462	SHREC '12
 table 2 †	13579	SHREC '12
 octopus †	16554	SHREC '12

Mesh name	Vertices	Source/Comments
 horse 0 †	19248	TOSCA [111]
 horse 6 †	19248	TOSCA
 horse 10 †	19248	TOSCA
 head 2 †	20359	Headspace [137], remeshed
 head 1 †	20490	Headspace, remeshed
 fish 1 †	24830	Bristol [138], remeshed
 fish 2 †	24873	Bristol, remeshed
 bunny †	26002	SHREC '12
 armadillo †	26002	SHREC '12
 hand 1 †	26000	SHREC '12
 hand 2 †	26000	SHREC '12
 cat 0 †	27894	TOSCA
 cat 6 †	27894	TOSCA

Mesh name	Vertices	Source/Comments
 cat 10 †	27894	TOSCA
 victoria 17 †	45659	TOSCA
 victoria 21 †	45659	TOSCA
 david 12	52565	TOSCA
 kid 7	59727	KIDS [112]

Meshes marked by † are used in the experiments in chapters 4 and 5.

C List of pairs used in matching problem

screwdriver / screwdriver x
screwdriver / screwdriver y
screwdriver x / screwdriver y
table 1 / table 2 (ground truth between meshes unknown)
chair 1 / chair 2 (ground truth between meshes unknown)
horse 0 / horse 6
horse 0 / horse 10
horse 6 / horse 10
head 1 / head 2 (ground truth between meshes unknown)
fish 1 / fish 2 (ground truth between meshes unknown)
hand 1 / hand 2 (ground truth between meshes unknown)
cat 0 / cat 6
cat 0 / cat 10
cat 6 / cat 10
victoria 17 / victoria 21

The table, chair, head and fish meshes are pairs of distinct meshes, not obtained through deformation. They are “less isometric” than the meshes where the ground truth match is known, and do not have the same number of vertices.

D Reliability of Compressed Manifold Modes

Figure D.1 shows a set of comparison plots from the calculation of compressed manifold modes. The modes were calculated sequentially via ADMM, with a cap on mode calculation after 150 minutes. A cap was also set on the number of iterations required to get convergence. Modes were calculated twice, once allowing a maximum of 10K iterations (blue points) and once allowing a maximum of 100K iterations (red points). The parameter controlling sparsity was set to $\mu = 50$. Further details of the graphs are listed below.

TOP LEFT: Markers plot (number of vertices, number of CMMs calculated in time) for each mesh. As expected, fewer modes are calculated in a 2.5 hour window if the maximum number of iterations is increased (red points are lower than blue points).

TOP RIGHT: Markers plot (number of vertices, index of first mode for which the maximum number of iteration was reached) for each mesh. Some meshes fail at the same mode (where blue and red points overlap); some meshes do not fail (columns where there are blue points but no red points), however this may just be because there was not enough time to reach the mode of first failure; in general, the mode of first failure occurs later when the maximum number of iterations is increased (blue points are lower than red points).

BOTTOM LEFT: Markers plot (number of vertices, number of modes for which the maximum number of iterations was reached) for each mesh. Fewer modes fail when the maximum number of iteration is increased (red points are lower than blue points).

BOTTOM RIGHT: Markers plot (number of vertices, number of modes for which the maximum number of iterations was reached as a percentage of number of modes calculated in time) for each mesh.

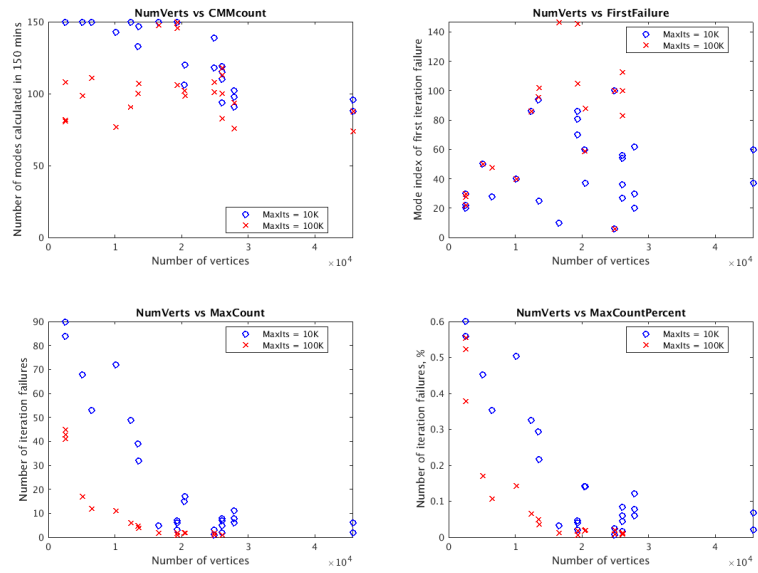


Figure D.1: CMM calculation details

E Additional figures

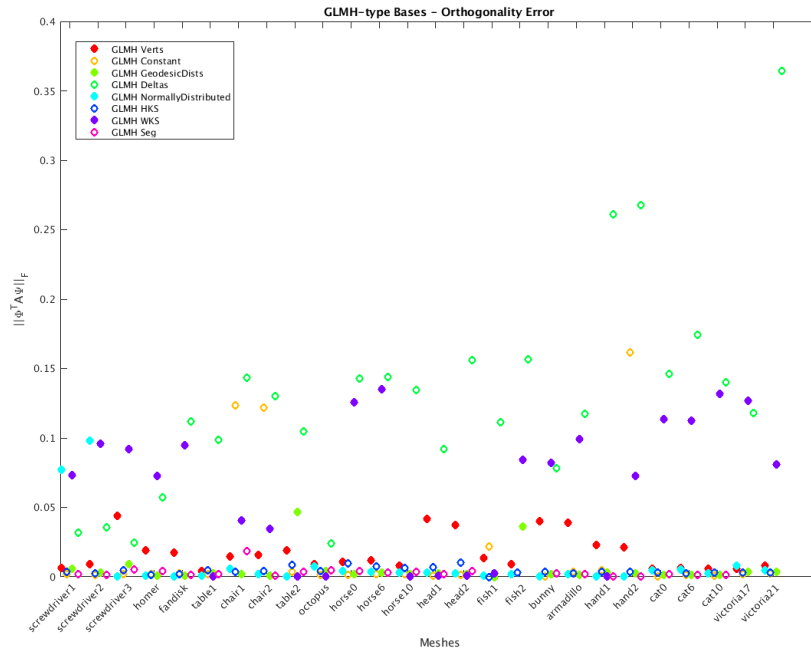


Figure E.2: The failure to meet the $\Phi^T A \Psi = I$ orthogonality condition, with large GLMH Delta errors included.

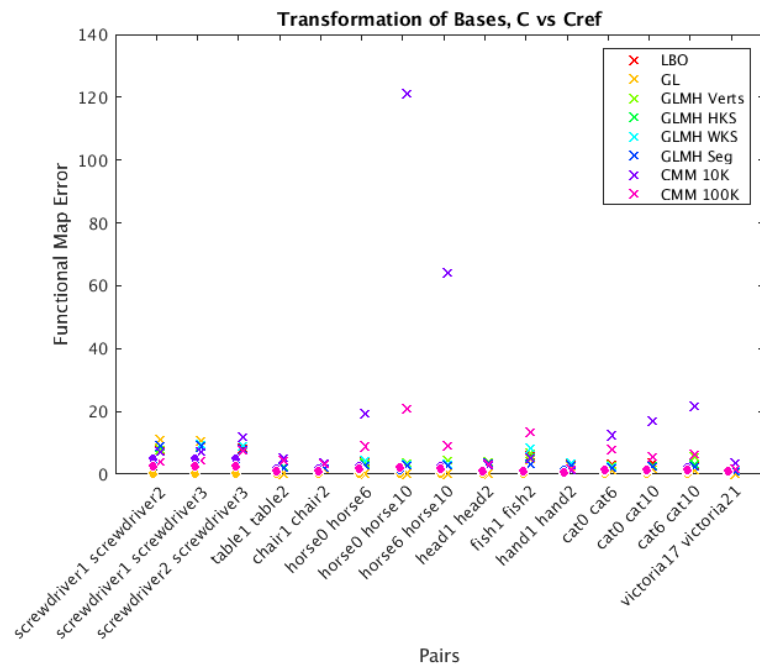


Figure E.3: Function transformation error between C and refined C , with basis type CMM 10K included.

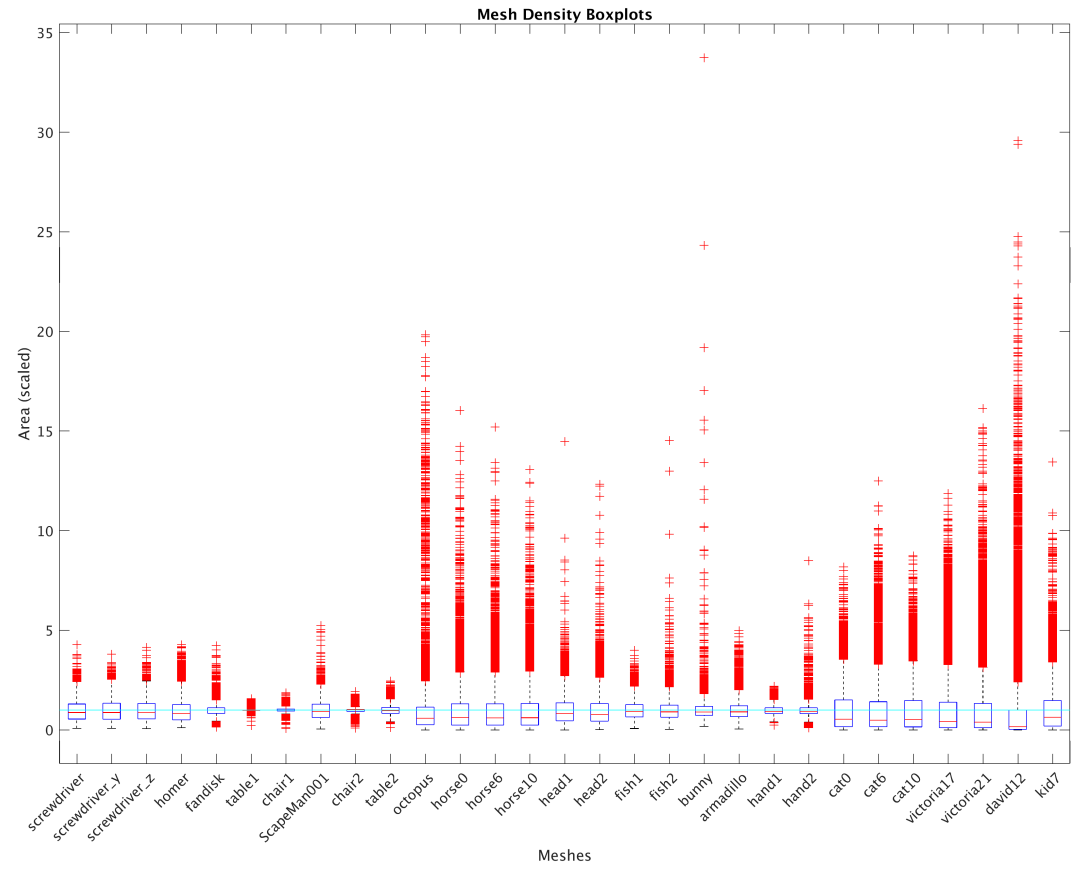


Figure E.4: Mesh area comparison boxplots – complete mesh set.

F Properties of the projection P

In section 6.4 the projection of any element $y \in \mathbb{R}^n$ into the tangent space $T_x(V_\Phi \cap \mathbb{E}_M)$ is given by Py where $P := I - \Phi\Phi^T M - xx^T M$. The following equalities may be useful in calculations, particularly in evaluation of $\|Pg\|$.

First, recall that $x^T M x = 1$, $\Phi^T M \Phi = I$ and $\Phi^T M x = 0$, then

$$\begin{aligned} P^T M P &= (I - M\Phi\Phi^T - Mxx^T)(M - M\Phi\Phi^T M - Mxx^T M) \\ &= M - M\Phi\Phi^T M + Mxx^T M \\ &= MP. \end{aligned}$$

Also,

$$\begin{aligned} P^T Mxx^T M P &= P(Mxx^T M - Mxx^T M\Phi\Phi^T M - Mxx^T xx^T M) \\ &= P(Mxx^T M - Mxx^T M) \\ &= 0. \end{aligned}$$

It follows that $\|Pg\|^2 = g^T M P g$ for either the Euclidean or the canonical metric.

G Proof of the Woodbury Matrix Identity (theorem 6.1.17)

Proof. Let $A \in \mathbb{R}^{n \times n}$, $U \in \mathbb{R}^{n \times k}$ and $V \in \mathbb{R}^{k \times n}$ and consider the system of equations given by

$$\begin{bmatrix} A & U \\ V & -I_k \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} I_n \\ 0 \end{bmatrix}.$$

Expanding out gives the equations

$$AX + UY = I_n \tag{G.1}$$

$$VX - Y = 0. \tag{G.2}$$

From (G.1)

$$X = A^{-1}(I_n - UY) \tag{G.3}$$

and from (G.2)

$$Y = VX. \tag{G.4}$$

Substitution of (G.3) into (G.2) gives

$$VA^{-1}(I_n - UY) - Y = 0$$

and, assuming that $(I_n + VA^{-1}U)$ is invertible, gives

$$Y = (I_n + VA^{-1}U)^{-1}VA^{-1}. \tag{G.5}$$

Substitution of (G.5) into (G.3) gives

$$X = A^{-1} - A^{-1}U(I_n + VA^{-1}U)^{-1}VA^{-1}. \tag{G.6}$$

Substitution of (G.4) into (G.1) gives

$$(A + UV)X = I_n$$

and hence (G.6) gives an expression for the inverse of $(A + UV)$. □

Bibliography

- [1] D. Thompson, *On Growth and Form*. Cambridge University Press, 1917.
- [2] M. Reuter, *Laplace Spectra for Shape Recognition*. PhD thesis, Universität, Hannover, 2005.
- [3] B. Vallet and B. Lévy, “Spectral Geometry Processing with Manifold Harmonics,” *Computer Graphics Forum*, vol. 27, no. 2, pp. 251–260, 2008.
- [4] M. Zhang, J. Ma, X. Liu, and L. Kobbelt, “Spectral Quadrangulation with Orientation and Alignment Control,” *ACM transactions on graphics*, vol. 27, no. 5, 2008.
- [5] M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas, “Functional Maps: A Flexible Representation of Maps Between Shapes,” *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2012*, vol. 4, no. 31, pp. 1–11, 2012.
- [6] S. Melzi, E. Rodola, U. Castellani, and M. Bronstein, “Localized Manifold Harmonics for Spectral Shape Analysis,” in *Eurographics Symposium on Geometry Processing*, (University College London), 3–5 July 2017. [Poster].
- [7] Y. Choukroun, G. Pai, and R. Kimmel, “Schrödinger Operator for Sparse Approximation of 3D Meshes,” in *Eurographics Symposium on Geometry Processing*, (University College London), 3–5 July 2017. [Poster].

- [8] V. Ozolinš, R. Lai, R. Caffisch, and S. Osher, “Compressed modes for variational problems in mathematics and physics,” *Proceedings of the National Academy of Sciences*, vol. 110, no. 46, pp. 18368–18373, 2013.
- [9] T. Neumann, K. Varanasi, C. Theobalt, M. Magnor, and M. Wacker, “Compressed Manifold Modes for Mesh Processing.,” *Eurographics Symposium on Geometry Processing 2014.*, vol. 33, no. 5, pp. 35–44, 2014.
- [10] J. Pokrass, A. Bronstein, M. Bronstein, P. Sprechmann, and G. Sapiro, “Sparse Modeling of Intrinsic Correspondences,” *Computer Graphics Forum*, vol. 32, no. 2pt4, pp. 459–468, 2013.
- [11] C. Zhang, W. Smith, A. Dessein, N. Pears, and H. Dai, “Functional Faces: Groupwise Dense Correspondence Using Functional Maps,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5033–5041, IEEE, June 2016.
- [12] O. Litany, E. Rodolà, A. Bronstein, M. Bronstein, and D. Cremers, “NonRigid Puzzles,” *Computer Graphics Forum*, vol. 35, no. 5, pp. 135–143, 2016.
- [13] Kovnatsky, A. and Bronstein, M. and Bresson, X. and Vandergheynst, P., “Functional correspondence by matrix completion,” in *IEEE Conference on Computer Vision and Pattern Recognition. Proceedings*, pp. 905–914, 2015.
- [14] A. Nasikun, C. Brandt, and K. Hildebrandt, “Fast Approximation of Laplace-Beltrami Eigenproblems,” *Computer Graphics Forum*, vol. 37, no. 5, pp. 121–134, 2018.
- [15] R. Horn and C. Johnson, *Matrix Analysis*. Cambridge, UK: Cambridge University Press, 1985.
- [16] J. Solomon, *Numerical Algorithms: Methods for Computer Vision, Machine Learning, and Graphics*. Natick, MA, USA: A. K. Peters, Ltd., 2015.
- [17] L. Hogben, ed., *Handbook of linear algebra*. Discrete mathematics and its applications, Boca Raton: CRC Press, second edition. ed., 2014.

- [18] “Cholesky Decomposition. [Online].” Wikipedia. [Accessed Dec 2018].
- [19] K. Petersen and M. Pedersen, “The Matrix Cookbook,” tech. rep., Technical University of Denmark, 2012. [Online], Version 20121115.
- [20] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstien, “Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers,” *Foundations and Trends in Machine Learning*, vol. 3, pp. 1–122, 2010.
- [21] R. Nishihara, L. Lessard, B. Recht, A. Packard, and M. Jordan, “A General Analysis of the Convergence of ADMM,” 2015. [Available from: <https://arxiv.org/abs/1502.02009>].
- [22] Y. Wang, W. Yin, and J. Zeng, “Global Convergence of ADMM in Nonconvex Nonsmooth Optimization,” *Journal of Scientific Computing*, vol. 78, no. 1, pp. 29–63, 2019.
- [23] B. Lévy and H. Zhang, “Elements of Geometry Processing,” in *SIGGRAPH Asia 2011 Courses*, SA ’11, (New York, NY, USA), pp. 1–48, ACM, 2011.
- [24] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy, *Polygon Mesh Processing*. Natick, MA: A.K. Peters, 2010.
- [25] M. Campen, “Quad Mesh Generation,” 2017. IGS Summerschool 2016, Berlin. [Slides available from: <http://www.geometrysummit.org/summerschool/presentations.html>].
- [26] G. Taubin, “A Signal Processing Approach to Fair Surface Design,” in *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’95, pp. 351–358, 1995.
- [27] M. Desbrun, M. Meyer, P. Schröder, and A. Barr, “Implicit Fairing of Irregular Meshes Using Diffusion and Curvature Flow,” in *Proceedings of the 26th annual conference on computer graphics and interactive techniques*, SIGGRAPH ’99, pp. 317–324, 1999.

- [28] A. Belyaev and Y. Ohtake, “A Comparison Of Mesh Smoothing Methods,” in *Israel-Korea Bi-National Conference on Geometric Modeling and Computer Graphics*, pp. 83–87, 2003.
- [29] V. Krishnamurthy and M. Levoy, “Fitting Smooth Surfaces to Dense Polygon Meshes,” in *Proceedings of the 23rd annual conference on computer graphics and interactive techniques*, SIGGRAPH '96, pp. 313–324, ACM, 1996.
- [30] B. Levy, S. Petitjean, N. Ray, and J. Maillot, “Least Squares Conformal Maps for Automatic Texture Atlas Generation,” *ACM Transactions On Graphics*, vol. 21, no. 3, pp. 362–371, 2002.
- [31] M. Floater and K. Hormann, “Surface parameterization: a tutorial and survey,” in *Advances in Multiresolution for Geometric Modelling* (N. Dodgson, M. Floater, and M. Sabin, eds.), (Berlin, Heidelberg), pp. 157–186, Springer Berlin Heidelberg, 2005.
- [32] M. Marinov and L. Kobbelt, “A Robust Two-Step Procedure for Quad-Dominant Remeshing,” *Computer Graphics Forum*, vol. 25, no. 3, pp. 537–546, 2006.
- [33] V. Surazhsky and C. Gotsman, “Explicit Surface Remeshing,” in *ACM International Conference Proceeding Series; Vol. 43: Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing : Aachen, Germany; 23-25 June 2003*, vol. 43, pp. 20–30, 2003.
- [34] P. Alliez, G. Ucelli, C. Gotsman, and M. Attene, *Recent Advances in Remeshing of Surfaces*, pp. 53–82. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.
- [35] M.-E. Algorri and F. Schmitt, “Mesh Simplification,” *Computer Graphics Forum*, vol. 15, no. 3, pp. 77–86, 1996.
- [36] D. Cohen-Steiner, P. Alliez, and M. Desbrun, “Variational Shape Approximation,” *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 905–914, 2004.

- [37] P. Cignoni, C. Montani, and R. Scopigno, “A Comparison of Mesh Simplification Algorithms,” *Computers & Graphics*, vol. 22, no. 1, pp. 37–54, 1998.
- [38] G. Barequet and S. Kumar, “Repairing CAD Models,” in *Proceedings. Visualization '97*, pp. 363–370, IEEE, 1997.
- [39] A. Gueziec, G. Taubin, F. Lazarus, and B. Hom, “Cutting and Stitching: Converting Sets of Polygons to Manifold Surfaces,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 7, no. 2, pp. 136–151, 2001.
- [40] T. Ju, “Fixing Geometric Errors on Polygonal Models: A Survey,” *Journal of Computer Science and Technology*, vol. 24, no. 1, pp. 19–29, 2009.
- [41] O. Sorkine and M. Alexa, “As-Rigid-As-Possible Surface Modeling,” in *Geometry Processing* (A. Belyaev and M. Garland, eds.), The Eurographics Association, 2007.
- [42] M. Botsch, M. Pauly, M. Gross, and L. Kobbelt, “Primo: Coupled prisms for intuitive surface modeling,” in *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, pp. 11–20, 2006.
- [43] M. Botsch and O. Sorkine, “On Linear Variational Surface Deformation Methods,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 1, pp. 213–230, 2008.
- [44] Y. Ma, Z. Chen, W. Hu, and W. Wang, “Packing Irregular Objects in 3D Space via Hybrid Optimization,” *Computer Graphics Forum*, vol. 37, no. 5, pp. 49–59, 2018.
- [45] M. Kazhdan, M. Bolitho, and H. Hoppe, “Poisson surface reconstruction,” in *Proceedings of the fourth Eurographics symposium on geometry processing*, SGP '06, pp. 61–70, Eurographics Association, 2006.
- [46] M. Wardetzky, S. Mathur, F. Kaelberer, and E. Grinspun, “Discrete Laplace operators: No free lunch,” in *Eurographics Symposium on Geometry Processing. Proceedings.*, pp. 33–37, 2007.

- [47] D. Cvetković, P. Rowlinson, and S. Simić, *An Introduction to the Theory of Graph Spectra*. Cambridge: Cambridge University Press, 2010.
- [48] U. Pinkall and K. Polthier, “Computing Discrete Minimal Surfaces and Their Conjugates,” *Experimental Mathematics*, vol. 2, no. 1, pp. 15–36, 1993.
- [49] C. Gotsman, X. Gu, and A. Sheffer, “Fundamentals of spherical parameterization for 3D meshes,” in *ACM SIGGRAPH 2003 Papers*, vol. 22 of *SIGGRAPH '03*, pp. 358–363, ACM, 2003.
- [50] H. Zhang, “Discrete combinatorial Laplacian operators for digital geometry processing,” in *SIAM Conference on Geometric Design.*, p. 5755923, 2004.
- [51] *Laplace-Beltrami: The Swiss Army Knife of Geometry Processing*, 2014. [Slides available from: <http://ddg.cs.columbia.edu/SGP2014/LaplaceBeltrami.pdf>].
- [52] T. Sakai, *Riemannian Geometry*. Providence, RI: American Mathematical Society, 1996.
- [53] S. Melzi, E. Rodolà, U. Castellani, and M. Bronstein, “Localized Manifold Harmonics for Spectral Shape Analysis,” 2017. [Available from: <http://arxiv.org/abs/1707.02596>].
- [54] Y. Choukroun, G. Pai, and R. Kimmel, “Sparse Approximation of 3D Meshes Using the Spectral Geometry of the Hamiltonian Operator,” *Journal of Mathematical Imaging and Vision*, vol. 60, no. 6, pp. 941–952, 2018.
- [55] J. . Tropp, A. Gilbert, and M. Strauss, “Algorithms for simultaneous sparse approximation. Part I: Greedy pursuit.,” *Signal Processing*, vol. 86, pp. 572–588, 2006.
- [56] Y. Dodge, “An Introduction to L_1 Norm Based Statistical Data Analysis,” *Computational Statistics and Data Analysis*, vol. 5, no. 4, pp. 23–,253, 1987.
- [57] D. Vidaurre, C. Bielza, and P. Larranaga, “A Survey of L1 Regression,” *International Statistical Review*, vol. 81, no. 3, 2013.

- [58] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.
- [59] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Adaptive computation and machine learning series, Cambridge, MA: MIT Press, 2017.
- [60] J. Lavery, “Univariate cubic L_p splines and shape-preserving, multiscale interpolation by univariate cubic L_1 splines,” *Computer Aided Geometric Design*, vol. 17, no. 4, pp. 319–336, 2000.
- [61] J. Lavery, “Shape-preserving, multiscale interpolation by univariate curvature-based cubic L_1 splines in Cartesian and polar coordinates,” *Computer Aided Geometric Design*, vol. 19, no. 4, pp. 257–273, 2002.
- [62] H. Avron, A. Sharf, C. Greif, and D. Cohen-Or, “ l_1 -Sparse Reconstruction of Sharp Point Set Surfaces,” *ACM Transactions On Graphics*, vol. 29, no. 5, pp. 1–12, 2010.
- [63] R. M. Rustomov, “Multiscale Biharmonic Kernels,” *Computer Graphics Forum*, vol. 30, no. 5, pp. 1521–1531, 2011.
- [64] MathWorks, “Computational Advantages of Sparse Matrices.” Webpage, 2019. [Accessed May 2019].
- [65] A. Bronstein, Y. Choukroun, R. Kimmel, and M. Sela, “Consistent Discretization and Minimization of the L_1 Norm on Manifolds,” *2016 Fourth International Conference on 3D Vision*, pp. 435–440, 2016.
- [66] K. Houston, “Compressed manifold modes: Fast calculation and natural ordering,” 2015. [Available from: <http://arxiv.org/abs/1507.00644>].
- [67] F. Barekat, R. Cflisch, and S. Osher, “On the Support of Compressed Modes,” *SIAM Journal on Mathematical Analysis*, vol. 49, no. 4, pp. 2573–2590, 2017.
- [68] R. Lai and S. Osher, “A Splitting Method for Orthogonality Constrained Problems,” *Journal of Scientific Computing*, vol. 58, no. 2, pp. 431–449, 2014.

- [69] M. Huska, D. Lazzaro, and S. Morigi, “Shape Partitioning via Lp Compressed Modes,” *Journal of Mathematical Imaging and Vision*, vol. 60, no. 7, pp. 1111–1131, 2018.
- [70] R.-C. Li, “Relative Perturbation Theory: I. Eigenvalue and Singular Value Variations,” *SIAM Journal on Matrix Analysis and Applications*, vol. 19, no. 4, pp. 956–982, 1998.
- [71] N. Parikh and S. Boyd, “Proximal Algorithms,” *Foundations and Trends in Optimization*, vol. 3, no. 1, pp. 123–231, 2013.
- [72] K. Houston, “Sequentially-Defined Compressed Modes via ADMM,” in *Symposium on Geometry Processing 2017 - Posters* (J. Brentzen and K. Hildebrandt, eds.), The Eurographics Association, 2017.
- [73] M. Ovsjanikov, Q. Mrigot, F. Mmoli, and L. Guibas, “One Point Isometric Matching with the Heat Kernel,” *Computer Graphics Forum*, vol. 29, no. 5, pp. 1555–1564, 2010-07.
- [74] J. Sun, M. Ovsjanikov, and L. Guibas, “A Concise and Provably Informative Multi-Scale Signature Based on Heat Diffusion. [Online],” *Eurographics*, vol. 5, no. 28, pp. 1383–1392, 2009.
- [75] E. Rodolà, M. Möller, and D. Cremers, “Point-wise Map Recovery and Refinement from Functional Correspondence,” 2015. [Available from: <http://arxiv.org/abs/1506.05603>].
- [76] A. Elad and R. Kimmel, “On Bending Invariant Signatures for Surfaces,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1285–1295, 2003.
- [77] F. Mémoli and G. Sapiro, “A theoretical and computational framework for isometry invariant recognition of point cloud data,” *Foundations of computational mathematics.*, vol. 5, no. 3, pp. 313–347, 2005.

- [78] S. Lee and M. Kazhdan, “Dense Point-to-Point Correspondences Between Genus-Zero Shapes,” *Computer Graphics Forum*, vol. 38, no. 5, pp. 27–37, 2019.
- [79] D. Ezuz and M. Ben-Chen, “Deblurring and Denoising of Maps between Shapes,” *Computer Graphics Forum*, vol. 36, no. 5, pp. 165–174, 2017.
- [80] M. Shoham, A. Vaxman, and M. BenChen, “Hierarchical Functional Maps between Subdivision Surfaces,” *Computer Graphics Forum*, vol. 38, no. 5, pp. 55–73, 2019.
- [81] V. Kim, Y. Lipman, and T. Funkhouser, “Blended intrinsic maps,” *ACM Transactions on Graphics (TOG)*, vol. 30, no. 4, pp. 1–12, 2011.
- [82] Y. Lipman, R. Rostamov, and T. Funkhouser, “Biharmonic Distance,” *ACM Transactions on Graphics (TOG)*, vol. 29, no. 3, pp. 1–11, 2010.
- [83] V. Surazhsky, T. Surazhsky, D. Kirsanov, S. Gortler, and H. Hoppe, “Fast exact and approximate geodesics on meshes,” *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3, pp. 553–560, 2005.
- [84] D. Martnez, L. Velho, and P. Carvalho, “Computing Geodesics on Triangular Meshes,” *Computers & Graphics*, vol. 29, no. 5, pp. 667–675, 2005.
- [85] K. Crane, C. Weischedel, and M. Wardetzky, “The heat method for distance computation,” *Communications of the ACM*, vol. 60, no. 11, pp. 90–99, 2017.
- [86] J. Gross, J. Yellen, and P. Zhang, *Handbook of Graph Theory*. CRC Press, 2 ed., 2013.
- [87] R. Rostamov, M. Ovsjanikov, O. Azencot, M. Ben-Chen, F. Chazal, and L. Guibas, “Map-based exploration of intrinsic shape differences and variability,” *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, pp. 1–12, 2013.
- [88] D. Mateus, R. Horaud, D. Knossow, F. Cuzzolin, and E. Boyer, “Articulated Shape Matching Using Laplacian Eigenfunctions and Unsupervised Point

- Registration,” in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, IEEE, 2008.
- [89] S. Umeyama, “An Eigendecomposition Approach to Weighted Graph Matching Problems,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 5, pp. 695–703, 1988.
- [90] W. Graustein, *Differential Geometry*. New York: Dover Publications, 1966.
- [91] Y. Canzani, “Analysis on Manifolds via the Laplacian.” Lecture notes, 2013. [Available from: <http://www.math.harvard.edu/~canzani/math253.html>].
- [92] M. Ovsjanikov, M. Ben-Chen, F. Chazal, and L. Guibas, “Analysis and Visualization of Maps Between Shapes,” *Computer Graphics Forum*, vol. 32, no. 6, pp. 135–145, 2013.
- [93] B. Allen, B. Curless, and Z. Popovi, “The space of human body shapes: reconstruction and parameterization from range scans,” in *ACM SIGGRAPH 2003 Papers*, vol. 22 of *SIGGRAPH '03*, pp. 587–594, ACM, 2003.
- [94] M. Ovsjanikov, Q. Mèrigot, F. Mémoli, and L. Guibas, “One Point Isometric Matching with the Heat Kernel,” *Computer Graphics Forum*, vol. 29, no. 5, pp. 1555–1564, 2010.
- [95] “Dijkstra’s algorithm. [Online].” Wikipedia. [Accessed April 2017].
- [96] M. Pauly, R. Keiser, and M. Gross, “Multi-scale feature extraction on point-sampled surfaces,” *Computer Graphics Forum*, vol. 22, no. 3, pp. 281,289., 2003.
- [97] X. Li and I. Guskov, “Multi-scale features for approximate alignment of point-based surfaces,” in *ACM International Conference Proceeding Series; Vol. 255: Proceedings of the third Eurographics symposium on Geometry processing : Vienna, Austria; 04-06 July 2005*, vol. 255, 2005.
- [98] M. Aubry, U. Schlickewei, and D. Cremers, “The wave kernel signature: A quantum mechanical approach to shape analysis,” in *2011 IEEE*

- International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 1626,1633, IEEE, 2011.
- [99] P. Skraba, M. Ovsjanikov, F. Chazal, and L. Guibas, “Persistence-based Segmentation of Deformable Shapes,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, (San Francisco, CA), pp. 45–52, 2010. [Accessed October 2015].
- [100] MathWorks, “mldivide, \.” Webpage, 2019. [Accessed May 2019].
- [101] K. Ye and L.-H. Lim, “Schubert Varieties and Distances between Subspaces of Different Dimensions,” *SIAM Journal on Matrix Analysis and Applications*, vol. 37, no. 3, pp. 1176–1197, 2016.
- [102] H. Wang and J. Oliensis, “Rigid Shape Matching by Segmentation Averaging,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 619–635, 2010.
- [103] H. Zhang, A. Sheffer, D. Cohen-Or, Q. Zhou, O. Van Kaick, and A. Tagliasacchi, “Deformation-Driven Shape Correspondence,” *Computer Graphics Forum*, vol. 27, no. 5, pp. 1431–1439, 2008.
- [104] A. Bronstein and M. Bronstein, “Not only size matters: Regularized partial matching of nonrigid shapes,” in *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. pp.1–6, June 2008.
- [105] O. Van Kaick, H. Zhang, G. Hamarneh, and D. CohenOr, “A Survey on Shape Correspondence,” *Computer Graphics Forum*, vol. 30, no. 6, pp. 1681–1707, 2011.
- [106] C. Lawson and H. R. J., *Solving Least Squares Problems*. New Jersey: Prentice-Hall, Inc., 1974.
- [107] A. Kovnatsky, M. Bronstein, A. Bronstein, K. Glashoff, and R. Kimmel, “Coupled quasi-harmonic bases,” *Computer Graphics Forum*, vol. 32, no. 2, pp. 439–448, 2013.

- [108] D. Eynard, A. Kovnatsky, M. Bronstein, K. Glashoff, and A. Bronstein, “Multimodal Manifold Analysis by Simultaneous Diagonalization of Laplacians,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 12, pp. 2505–2517, 2015.
- [109] D. Eynard, E. Rodola, K. Glashoff, and M. Bronstein, “Coupled Functional Maps,” in *2016 Fourth International Conference on 3D Vision (3DV)*, pp. 399–407, 2016.
- [110] farnboroughmaths, “Hungarian Algorithm [Online].” Youtube video, 2013. [Accessed May 2016] Available from: <https://www.youtube.com/watch?v=dQDZNHwuuOY>.
- [111] Bronstein, A.M. and Bronstein, M.M. and Kimmel, R., “Numerical geometry of non-rigid shapes.” Dataset, online, 2008. [Accessed Nov 2017] http://tosca.cs.technion.ac.il/book/resources_data.html.
- [112] E. Rodola, S. Bulo, T. Windheuser, M. Vestner, and D. Cremers, “Dense Non-rigid Shape Correspondence Using Random Forests,” in *IEEE Conference on Computer Vision and Pattern Recognition. Proceedings*, pp. 4177–4184, 2014.
- [113] P. Besl and N. McKay, “A method for registration of 3-D shapes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.
- [114] S. Biasotti, A. Cerri, A. Bronstein, and M. Bronstein, “Recent Trends, Applications and Perspectives in 3D Shape Similarity Assessment,” *Computer Graphics Forum*, 2015.
- [115] M. Ovsjanikov, E. Corman, M. Bronstein, E. Rodola, M. Ben-Chen, L. Guibas, F. Chazal, and A. Bronstein, “Computing and Processing Correspondences with Functional Maps,” in *ACM SIGGRAPH 2017 Courses*, (University College London), pp. 1–62, 2017.

- [116] E. Rodolà, L. Cosmo, M. Bronstein, A. Torsello, and D. Cremers, “Partial Functional Correspondence,” *Computer Graphics Forum*, vol. 36, no. 1, pp. 222–236, 2017.
- [117] P.-A. Absil, R. Mahoney, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*. Princeton, New Jersey: Princeton University Press, 2008.
- [118] A. Edelman, T. Arias, and S. Smith, “The Geometry of Algorithms with Orthogonality Constraints,” *SIAM Journal on Matrix Analysis and Applications*, vol. 20, no. 2, pp. 303–353, 1998.
- [119] J. Siegel, *Accelerated First-Order Optimization with Orthogonality Constraints*. PhD thesis, UCLA, 2018.
- [120] Z. Wen and W. Yin, “A Feasible Method for Optimization with Orthogonality Constraints,” *Mathematical Programming*, vol. 142, no. 1-2, pp. 397–434, 2013.
- [121] H. Oviedo and H. Lara, “Riemannian Conjugate Gradient Algorithm with Implicit Vector Transport for Optimization on the Stiefel Manifold.” [Online] Available from: www.researchgate.net, 2018.
- [122] H. Sato and K. Aihara, “Cholesky QR-based retraction on the generalized Stiefel manifold,” *Computational Optimization and Applications*, vol. 72, no. 2, pp. 293–308, 2019.
- [123] H. Tagare, “Notes on Optimization on Stiefel Manifolds,” tech. rep., Yale University, 2011.
- [124] J. Nocedal and S. Wright, *Numerical Optimization*. New York: Springer-Verlag, second edition ed., 2006.
- [125] Y. Nesterov, “A Method of Solving a Convex Programming Problem with Convergence Rate $0(1/k^2)$,” *Soviet Mathematics Doklady*, vol. 27, no. 2, pp. 372–376, 1983.

- [126] B. O’Donoghue and E. Candès, “Adaptive Restart for Accelerated Gradient Schemes,” *Foundations of Computational Mathematics*, vol. 15, no. 3, pp. 715–732, 2015.
- [127] J. Barzilai and J. M. Borwein, “Two-Point Step Size Gradient Methods,” *IMA Journal of Numerical Analysis*, vol. 8, no. 1, pp. 141–148, 1988.
- [128] W. Yin, “The Barzilai-Borwein method.” Lecture notes, Math273a: Optimization. UCLA. [Available online], 2015.
- [129] J. Lee, *Introduction to Smooth Manifolds*. Graduate texts in mathematics ; 218, New York: Springer, 2003.
- [130] K. Lange, “Convergence of EM Image Reconstruction Algorithms with Gibbs Smoothing,” *IEEE Transactions on Medical Imaging*, vol. 9, no. 4, pp. 439–446, 1990.
- [131] R. Cook, “Stochastic Sampling in Computer Graphics,” *ACM Transactions on Graphics (TOG)*, vol. 5, no. 1, pp. 51–72, 1986.
- [132] C. Yuksel, “Sample Elimination for Generating Poisson Disk Sample Sets,” *Computer Graphics Forum (Proceedings of EUROGRAPHICS 2015)*, vol. 34, no. 2, pp. 25–32, 2015.
- [133] <http://visionair.ge.imati.cnr.it/ontologies/shapes/view.jsp?id=40-Screwdriver#>, “Screwdriver.” Online shape repository. [Accessed Nov 2017].
- [134] <http://www-rech.telecom-lille1.eu/shrec2012-segmentation/#dataset>, “SHREC ’12.” Shape Retrieval Contest Dataset, online. [Accessed Nov 2017].
- [135] <https://www.yobi3d.com/q/fandisk>, “Fandisk.” Online shape repository. [Accessed Nov 2017].
- [136] <https://ai.stanford.edu/~drago/Projects/scape/scape.html>, “SCAPE.” Dataset, online. [Accessed Nov 2017].

- [137] <https://www-users.cs.york.ac.uk/~nep/research/Headspace/>, “The Headspace dataset.” Dataset, online. [Accessed Nov 2017].
- [138] Hammond, C., “Fish jawbone meshes.” Personal correspondence. University of Bristol.