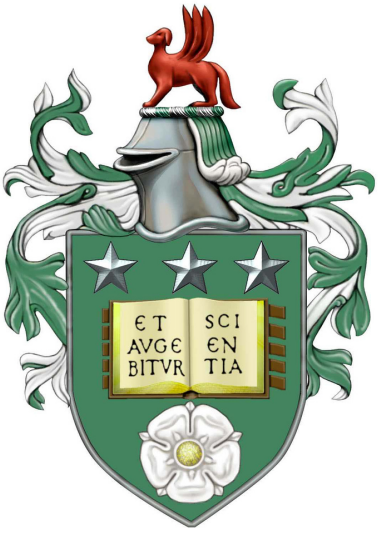
Computational Modelling of the Mechanical Properties of Elastic Fibre Networks

Mark Houghton

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



The University of Leeds
School of Computing
June 2020

The candidate confirms that the work submitted is his own, except where work which has formed part of a jointly authored publication has been included. The contribution of the candidate and the other authors to this work has been explicitly indicated below. The candidate confirms that appropriate credit has been given within the thesis where reference has been made to the work of others.

Some parts of the work presented in Chapters 3, 4, 5 and 6 have been published in the following article [36]:

M. Houghton, D. Head, M. Walkley. A Numerical Model for Random Fibre Networks, *LNCS*, vol. 11189, p. 408, Springer (2019)

The above publication is primarily the work of the candidate.

Acknowledgements

I wish to express my deepest gratitude to both of my supervisors David Head and Mark Walkley for their expertise, advice and support, and for many interesting discussions over the last few years. I'd like to thank the friends I've made throughout this process, who have also provided advice and support, but in particular I am indebted to Jon for his help with my countless queries regarding `make`. I am thankful for the encouragement of my family, in particular for the patience of my parents during the final stages of writing. I am deeply grateful for Giulia's endless support and understanding throughout the entire project. Finally I'd like to thank the examiners for reading my manuscript, and I'd like to acknowledge EPSRC for funding my research.

Abstract

From everyday items such as paper, felt and nappies, to sophisticated biological structures such as mammalian cytoskeletons and the collagen of extracellular matrices, many materials are made up of complex disordered fibre networks with varying microstructures. It is often important to tune the mechanical properties of such networks for their specific application. The macroscopic network response to an applied load can be controlled by modifying the properties of component fibres at the microscopic level. A model relating the properties at these two scales is desirable for the design and fabrication of better materials. We developed a numerical code predicting the mechanical properties of 2D and 3D elastic fibre networks. Specifically, we find an efficient solution of the linear matrix system obtained from a large system of equations, representing a given random fibre network in mechanical equilibrium with an applied linear shear at the network boundary. This global system is assembled by considering the individual contributions of fibres modelled as cross-linked slender elastic rods, and then solved to obtain a prediction of the network displacement and total elastic energy for the applied shear. To study various network architectures, we also developed another code for network generation and visualisation. Using our software, we analysed the numerical performance of preconditioners for the iterative methods used to solve the linear system. This was applied to systems representing established 2D networks, and investigated the mechanical properties of layered 3D and fully 3D disordered systems. Our choices of preconditioners were motivated by exploiting the distinct block structure of our assembled matrix. Drawing from the application of needlepunched nonwoven fabrics, we designed a series of novel networks consisting of random 2D Mikado cross-linked layers. Applying our numerical model, we were able to explore the effects of material anisotropy on shear response, and provide a first analysis of how the macroscopic mechanics are driven by variations in microscopic properties. Evidence was also presented that our software can be used to model the mechanical properties of fully 3D random fibre networks under imposed macroscopic shear. This work can be used to direct future research, and offers the opportunity to model 3D fibrous materials using a geometry generated differently to many related works - in terms of both cross-linking procedure and cross-link type variability. The final outcome of this work is a reusable piece of software for modelling the mechanical properties of elastic fibre networks with various geometries under a macroscopically applied shear. Through use of numerical techniques and integration with the PETSc library [4], we solve the resulting systems of these networks effectively and within reasonable time scales, with the opportunity of additional optimisation if further work were to be carried out.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Material science is a major research discipline that is responsible for technological innovation in many areas of manufacturing. Providing insight into various material properties, we are able to engineer complex new materials, and to manipulate the features of existing ones. A number of interesting mechanical properties can be inferred by looking at the internal structure of a material, and advances in microscopy allow us to see that there are entire classes of fibrous material with little to no visible order in the arrangement of their constituent fibres. These disordered materials are common to the class of materials known as nonwovens, but are also prevalent in many areas of the natural world such as in mammalian cytoskeletons or networks of connective tissue. Through many research contributions in the field of biophysics, we now understand that the disordered internal structure of these materials plays an important role in their remarkable mechanics. By considering nonwoven fabrics and biopolymers concurrently, we can apply established approaches for modelling biopolymers to provide predictions for the mechanical properties of materials such as needlepunched nonwoven fabrics (see Chapter 6).

Beginning with an overview of the nonwoven fabric industry, we follow with a brief look at the raw material that is typically used as the building block of a nonwoven fabric: polymers. From this we give a high-level discussion of some of the primary examples of biopolymers that have sparked increased interest among the biophysics community in the last two decades. We conclude the chapter with an outline of what will follow in the remaining chapters.

1.1 Nonwoven Fabrics

Despite being the most modern branch of the textile industry [87], the use of nonwoven fabrics dates back much further, with felting being used in Asia and Europe in ancient times [54]. In the more recent past, nonwovens were seen as low price substitutes for traditional textiles, leading to the misconception that they are best suited for bulk trading at low cost. Advances in the technology surrounding manufacturing techniques has allowed the industry to begin making

much more sophisticated materials designed with specific properties. The fact that fabrics can be made from the raw materials in one continuous line of production with many steps of traditional textile manufacturing removed, is an attractive aspect of nonwoven production. Improvements in manufacturing combined with a streamlined production process has led to the large variety of nonwovens that are available to date, with applications spread across the automotive industry, civil engineering, clothing, household products, sanitation and medical products, and packaging.

Developing a universal definition for the word *nonwoven* has proven to be a difficult task, with the Handbook of Nonwoven Filter Media [41] listing definitions from 8 different sources. At a high-level, for a material to be considered nonwoven, the yarn-spinning stage should have been omitted from the manufacturing process, and the network of raw fibrous material known as the *web* should have been consolidated through mechanical, chemical or thermal methods, instead of through conventional knitting or weaving. Taken from [74, p. 2], the European Disposables and Nonwovens Association (EDANA) describes a nonwoven as

"a manufactured sheet, web or batt of directionally or randomly orientated fibres, bonded by friction, and/or cohesion and/or adhesion."

They then proceed to exclude a list of materials that would otherwise be captured by this definition, such as paper, and woven, knitted and stitchbonded products. In the following survey of the world of nonwovens, we restrict the content to an overview of the most common manufacturing processes and a brief discussion of the importance of nonwoven material characterisation.

1.1.1 Manufacturing Process

Despite the large breadth of manufacturing techniques that exist for the production of nonwoven materials, the process typically follows the same four stages:

1. Raw material selection.
2. Web formation.
3. Web bonding.
4. Web finishing.

The first stage of raw material selection is when the manufacturer must determine the most suitable fibre type to use, which very much depends on the desired properties of the finished product. Fibres can be natural (e.g. wool or wood pulp), synthetic (e.g. rayon or nylon), or a blend of two or more fibres. The web formation stage consists of assembling a weak web structure of the chosen fibres. There are many techniques for web formation, split into dry-laid, wet-laid and polymer-laid, all of which we cover in more detail below along with a deeper look at web bonding. The web bonding stage is where the web is consolidated using mechanical, thermal or

chemical bonding techniques. It is this stage that gives the material its strength and integrity through the introduction of bonds between fibres. Of particular interest to us is the mechanical bonding technique known as needle punching, whereby fibres are bonded using barbed needles. Nonwoven fabrics are often left unfinished, but when finishing treatment does take place, it can be broadly split into mechanical finishing techniques (e.g. embossing, laminating or brushing), and chemical finishing techniques (e.g. bleaching, dyeing or printing)[87].

Web Formation

Web formation is where a two-dimensional *web* or three-dimensional *batt* (i.e. an assembly of layered webs) is formed from the raw fibrous material. Dry-laid formed webs include carded webs, garnetts and air-laid webs. Carded webs and garnetts are produced using methods that derive from the textile industry such as carding, and generally result in webs with higher fibre alignment than webs formed through the air-laid method. Air-laid web formation is a process taken from the paper industry. Wood pulp is combined with man-made fibres into a blend that is dispersed by air onto a moving conveyor belt to produce a web of randomly oriented fibres.

Wet-laid approaches are where webs are formed through the manipulation of aqueous suspensions of short fibres. Fibres are heavily diluted with water and chemicals to form a *slurry*, before being deposited onto a moving wire mesh that allows excess water to be drained off. After draining, the randomly laid fibres form a uniform web that can be sent for bonding and finishing [see Figure 1.1 *(left)*]. The differences between wet-laid nonwovens and wet-laid pa-

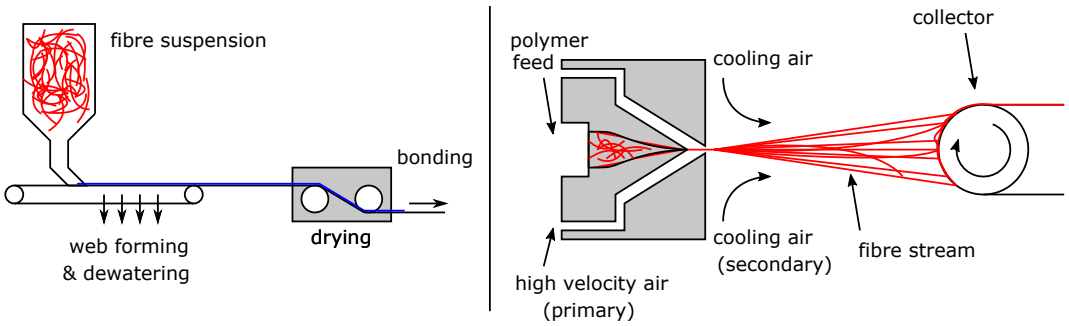


Figure 1.1: A schematic showing wet-laid *(left)* and meltblowing *(right)* web formation processes. Adapted from [87, p. 4] *(left)* and [41, p. 216] *(right)*.

pers are subtle, with the EDANA [74, p. 7] distinguishing the difference in terms of mass, fibre aspect ratio and material density.

Polymer-laid web formation uses manufacturing machinery originating from polymer extrusion, with the two primary processes used commercially being spunbonding and meltblowing. Both of these processes involve extruding fibres from molten polymer through a machine called a spinnerette onto a conveyor to form randomly oriented webs. Figure 1.1 *(right)* demonstrates a typical meltblowing web forming process.

The isotropy of fabric properties is strongly influenced by the arrangement and orientation of fibres in the web. Fibre orientation is typically measured by recording the ratio of machine direction (MD) to cross direction (CD) of the fabric. In practice, most nonwovens are anisotropic and the ratio for a truly isotropic material (MD:CD = 1) is difficult to achieve [74, pp. 8,412].

Bonding Techniques

The choice of bonding process is important, as different processes affect the degree of bonding which has an important influence on the mechanical properties of the final fabric.

Thermal bonding techniques apply heat and pressure to form bonds at the intersection points of thermoplastic fibres contained in the web. To achieve this there are a number of processes available, including *thermal calendaring*, *through-air* bonding, *thermal radiation* bonding and *ultrasonic* bonding.

Chemical bonding techniques deposit a bonding agent on or within the formed web. This is generally done by spraying or printing the adhesive *onto* the web, or by saturating the adhesive *within* the web. Saturated bonding results in the maximum degree of bonding, whereas print bonding gives a more intermediate degree of bonding. Spray bonding leads to the lowest degree of bonding since most of the bonding agent remains near the surface of the fabric.

Mechanical bonding involves entangling fibres together using a method such as stitchbonding, hydroentanglement or needlepunching. Stitchbonding knits needles through cross-lapped fleeces to form interlooped stitches, and is typically used for decorative fabrics and backing fabrics for artificial leather [87, p. 6]. We note that although the EDANA excludes stitchbonded textiles from their classification of nonwovens, it is still a bonding method that is widely discussed in the literature [41, 74, 87]. Hydroentanglement is similar to needlepunching (see below), but instead of needles uses a curtain of high-pressure water jets to entangle fibres. Following the entanglement caused by the jets of water, binders are added to the fabric to reinforce the bonds formed. Hydroentanglement is used for medical fabrics such as gowns and dressing, as well as dust cloths and wipes [74, pp. 288–294].

Needlepunching was originally developed as an alternative for mechanically bonding fabrics that could not be felted like wool, and was used primarily in applications such as mattress padding, carpet underlay and insulation, using fibres such as jute, coir and hair. Since its first use, synthetic fibres have become more widely available, giving rise to the opportunity for production of needlepunched fabrics to expand to a range of additional applications. These include geosynthetics, filter media, household and industrial wipes, synthetic leather and automotive fabrics.

To bond a fabric through needlepunching, the batt is fed through a needlepunching loom, where a board of needles is rapidly punched through the batt. As the needles pass through the batt, tiny barbs on the needle blades hook onto fibres such that the fibres become entangled as the needles are drawn out of the fabric. The fibres unhook from the barbs as the frictional forces generated from the entanglement outcompete the tension in the hooked fibre [87, p. 165].

Any fibres still hooked on the barbs as the needles leave the fabric are stripped from the needle using the stripper plate. Figure 1.2 shows a schematic of the needlepunching process, with a typical loom design *(left)* and an example of needle induced entanglement *(right)*.

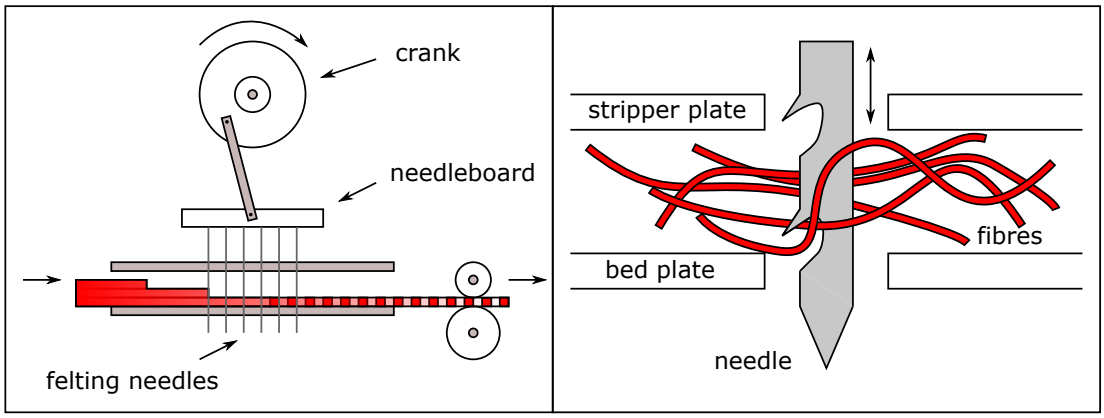


Figure 1.2: A schematic of a fibrous material mechanically bonded via needle punching *(left)*, with a demonstration of fibres entangled via a barb *(right)*. Adapted from [74, pp. 223–224].

In each of the bonding processes discussed above the degree of bonding depends on the tuning of a unique set of parameters. For example, with chemical bonding, the degree of bonding will be influenced by the amount of adhesive binders used, but also by whether the binders are sprayed, printed or saturated. On the other hand, the degree of bonding when using thermal processes depends on the temperature, applied pressure and conveyor speed. Finally with needlepunching, the degree of bonding depends not only on the quantity of needles, conveyor speed and punch (downstroke) speed, but also on the needle design. Figure 1.3 illustrates some of the design choices that are considered in a typical needle selection.

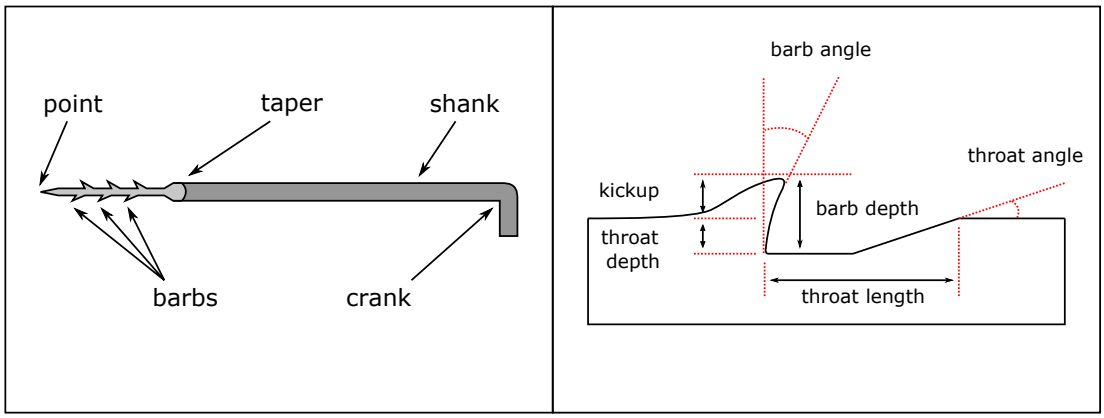


Figure 1.3: The anatomy of a needle *(left)* and the factors to consider when designing or selecting needles for needlepunching *(right)*. Adapted from [74, p. 227].

1.1.2 Characterisation & Modelling

Nonwoven materials can be characterised in a number of ways, including by manufacturing technique (e.g web formation and consolidation), but also by what properties the material has, with the importance of each property depending greatly on the specific application. For example, surgical masks should be designed to meet the appropriate filtration requirements (e.g. creating a barrier for larger liquid droplets while allowing smaller particles such as air to pass through). Hygiene products should be soft, fire fighting jackets should be heat resistant, wipes should hold antibacterial soap, and roofing should be tear resistant.

To meet these demands, manufacturers need to consider what are the most appropriate raw materials and manufacturing techniques to use, and to do this it is necessary to consider the chemical, physical and mechanical properties of the final material resulting from these choices. Mechanical properties include tensile properties such as the Young's modulus, as well as frictional properties, compression recovery, crease and tear resistance, and energy absorption.

The mechanical properties of nonwoven fabrics can be influenced by a variety of parameters including individual fibre and bond properties, but also network level parameters such as fabric dimensions, density and porosity. Some examples of fibre parameters influencing the mechanics are fibre dimensions (e.g. diameter and length), respective alignment across the network, and properties such as Young's modulus, elasticity, and bending rigidity. Bond characterisation includes bond type, size, rigidity and prevalence.

In needlepunched fabrics, bonds are typically flexible and elastic. This contrasts with thermally or chemically bonded fabrics that include solid bonds, creating areas in the fabric with higher rigidity. The size of a bond is dependent on parameters, such as needle diameter, number of needles, and ratio of needle depth to fibre diameter.

As we will see in the next chapter, predicting the properties of a finished material (at the macroscopic level) based on the properties of its constituent components (at the microscopic level) is not trivial. Without the use of computational simulation, property analysis of nonwovens resorts to a cyclic process of measuring the properties of the material *after* it has been manufactured, noting what should be changed and then manufacturing a revised material. This expensive and time consuming trial and error approach presents a clear need for a computational model that can efficiently provide predictions for the (mechanical) properties of nonwoven materials.

1.2 Polymers

The word *polymer* has its origins in Greek, where *poly* means many, and *mer* means unit. Polymers are macromolecules belonging to the soft matter class of materials, alongside colloids, surfactants and liquid crystals. Soft matter materials are characterised by large molecular building blocks, such that their structural units are much larger in scale than that of atoms. The fact that soft matter is composed of large components allows it to deform easily, demonstrating

a large response to weak forces. Furthermore, the structure of soft matter also lends itself to a slow response, such that typical response times are much longer than those of simple liquids.

Polymer science grew dramatically as a field in the latter half of the 20th century [69] due in part to the explosion of global plastic use. However, humans have been exploiting the properties of polymers for much longer than that, with celluloid being used for piano keys and billiard balls from 1856 [69], and natural rubber from rubber-tree plants being used for centuries before that [73]. Outside of human usage, polymers also serve as the basic building blocks of life. The fibrous proteins that connect our tissues, hold together our cells, and the nucleic acids that build up our DNA are all biopolymers. Polymers have a variety of interesting and valuable properties, including the viscoelasticity that allows a material such as chewing gum to behave both like an elastic solid and as a viscous liquid. As we discuss below, polymers are also susceptible to dramatic changes in property when their chemical structure is modified. They can be hard like the plastics used in car parts, or soft like the rubbers used in elastic rubber bands [17].

1.2.1 Structure & Chemistry

A polymer is a large molecule of covalently bonded repeating units known as monomers. The process in which the polymer is formed from several monomers is called polymerisation (see Figure 1.4), and the number of monomers that form the polymer is the *degree of polymerisation*, N. The polymerisation seen in Figure 1.4 suggests a straightforward chain-growth process whereby each monomer is added neatly to the end of the existing chain, and no atoms are lost in the process. While this figure is oversimplified, it still encapsulates the core idea behind *addition* polymerisation. The other main method for polymer preparation is through a condensation

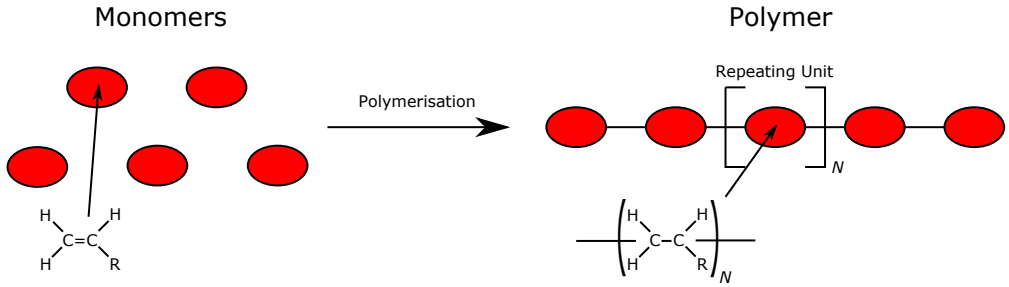


Figure 1.4: The process of polymerisation. Monomers bond together into N repeating units to form a single large molecule known as a polymer. Adapted from [69, p. 128] and [73, p. 3].

reaction where a small molecule such as water is lost in the process (see Figure 1.5). Polymers prepared in this way are referred to as *condensation* polymers.

A variety of factors affect the properties of a given polymer, including which monomers are combined to make the polymer, how atoms are organised along the polymer chain, polymer chain length, and polymer architecture. If a polymer consists of just one type of monomer, we

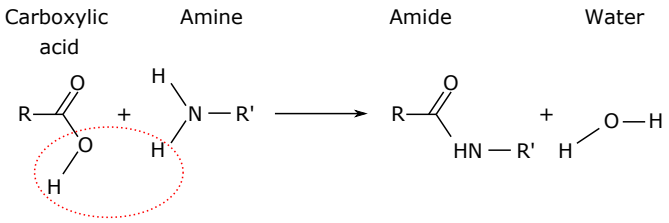


Figure 1.5: An example of condensation polymerisation. Amine reacts with carboxylic acid to form amide. Adapted from [69, p. 129].

refer to it as a *homopolymer*, else if it is constructed from more than one monomer type, it is a *heteropolymer*. If we consider a polymer formed using exactly two types of monomer (known as a *copolymer*), we note that there are multiple possible orderings of the monomers including alternate, block and also graft (see Figure 1.6). Disregarding whether we are considering a

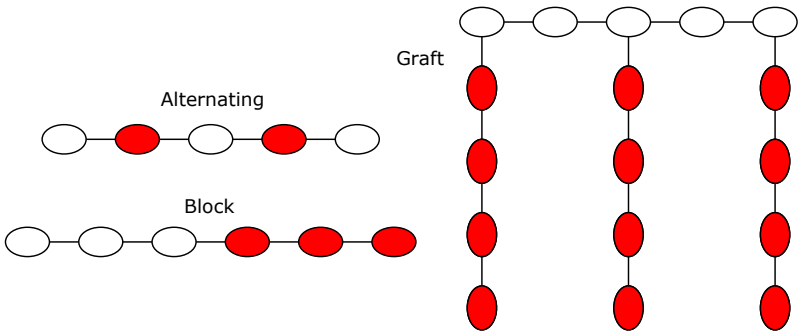


Figure 1.6: A selection of different copolymer types. The alternate and block types have a linear architecture, and the graft type has a comb architecture. Adapted from [69].

homopolymer or heteropolymer, there are a large number of possible architectures, including the linear and comb forms seen already in Figure 1.6, but also ring, star, H, and ladder forms. Given a specific polymer architecture, and assuming the chemical identity of the monomers forming the polymer is determined, further variations are possible within the polymer microstructure. In particular, the atoms along the chain can be organised in multiple ways (depending on factors such as bond angle) before being fixed during polymerisation. Finally, varying the degree of polymerisation will result in polymers with different chain lengths, which can affect properties such as boiling and melting points, resulting in different physical states (see Section 1.2.2) for the same polymer with a differing chain length. From this brief overview of the structure of polymers, the difficulty in achieving a realistic computational polymer model is clear, and it is evident that some simplification may be beneficial. In Section 1.2.3 we introduce a foundational model for an ideal polymer chain, but first we conclude our discussion of polymer characterisation with an overview of the different physical states of polymers.

1.2.2 Physical States

Polymers demonstrate a rich set of behaviours that are directly influenced by the various states that they can be in. Critical temperatures dictate the crossover between different polymer phases. As temperature increases above the glass transition temperature, T_g, polymers cease to be glassy and brittle and become instead viscous and rubbery. As temperature is increased further, there is another critical point known as the melting temperature, $T_m > T_g$, where semicrystalline solids begin a melting transition. We now review the behaviour of polymers in these different phases in more detail, beginning with the types of polymer liquid for temperature $T > T_m$.

Polymer Liquids

Polymer liquids can be grouped into solutions and melts. When a polymer is dissolved in a solvent we say it is a *polymer solution*, and when it forms a bulk liquid state in the absence of a solvent we say it is a *polymer melt*. Polymer solutions can be categorised further depending on the polymer concentration dissolved in the solution. To describe these additional solution regimes, we introduce the volume fraction ϕ, which can be given as the ratio of polymer mass concentration, c, and polymer mass density, ρ,

$$\phi = \frac{c}{\rho}. \tag{1.1}$$

Next we introduce the overlap volume fraction, ϕ^*, which can be given as the ratio of occupied chain volume, Nv_{mon}, and the solution spanned by the polymer chain, V,

$$\phi^* = \frac{Nv_{\text{mon}}}{V}, \tag{1.2}$$

where v_{mon} is the occupied volume of a single chemical monomer, and N is the degree of polymerisation [73, p. 13]. From this, a solution is defined as *dilute* for $\phi < \phi^*$ and *semidilute* for $\phi > \phi^*$. Generally speaking, dilute polymer solutions have very similar properties to the pure solvent, whereas semidilute polymer solutions typically show viscous and physical properties that are dominated by the dissolved polymers due to the overlapping coils of the polymers. In contrast, the mechanical properties of polymer melts are described as viscoelastic, due to a flowing viscous response at long time scales and elastic behaviour at shorter time scales.

Polymer Solids

Semicrystalline solids form from polymer melts when the melt is cooled below melting temperature (i.e. $T < T_m$), and *glasses* form when further cooling causes a drop below the glass transition temperature (i.e. $T < T_g$). Polymer *networks* are formed when polymer melt chains react together to become covalently bonded (or cross-linked). A polymer network above the glass transition point, T_g, is termed a *soft solid*, since the chains between cross-links of the

network are able to move locally but not globally (due to local thermal excitation). Finally, when a polymer network is swollen in a solvent it is called a *gel*, where we note that although softness can be increased by adding more solvent, we still refer to the gel as a solid due to the permanent cross-links that connect individual polymer chains [73, p. 15].

Liquid Crystals

The wide variety of intermediate states between crystalline solids and amorphous liquids can be loosely collected into the category of liquid crystals. Liquid crystal states occur when there is a sufficient number of rigid rod-like monomers, called *mesogens*. Polymers successfully formed from mesogens typically demonstrate notable structural order, leading to physical properties with a higher degree of anisotropy. Amongst the phases that can occur above the melting point but before the liquid becomes isotropic, are the *nematic* phase (with mesogens locally aligned), and the *smectic* phases (with the mesogens forming layered structures) [73, p. 15].

1.2.3 Polymer Modelling

The mechanical properties of conventional polymers are now largely understood, with a wide variety of established models in the literature [18, 73]. Single chain conformation models provide a foundation for some of the more complex models and split broadly into ideal chain models and real chain models. A natural starting point for modelling an ideal polymer chain conformation is the *freely jointed chain model*. If we represent a polymer as a series of n bonds (where $n = N - 1$), each with an orientation described by a bond vector $\mathbf{r}_i$, then we can write the end-to-end vector as

$$\mathbf{R} = \sum_{i=1}^{n} \mathbf{r}_i. \tag{1.3}$$

Then for a freely jointed chain where we assume segments have constant bond length ℓ [see Figure 1.7 *(left)*], we can write the mean square end-to-end distance as

$$\langle \mathbf{R}^2 \rangle = \langle \mathbf{R} \cdot \mathbf{R} \rangle = \langle \sum_{i=1}^{n} \mathbf{r}_i \cdot \sum_{j=1}^{n} \mathbf{r}_j \rangle = \sum_{i=1}^{n} \sum_{j=1}^{n} \langle \mathbf{r}_i \cdot \mathbf{r}_j \rangle = n\ell^2 + \sum_{i \neq j}^{n} \langle \mathbf{r}_i \cdot \mathbf{r}_j \rangle = n\ell^2, \tag{1.4}$$

where we take out the $i = j$ cases from the summation, and drop the $i \neq j$ term since a freely jointed chain implies that different segments i and j have a completely uncorrelated direction. The angular braces $\langle . \rangle$, are used to denote averaging over every possible conformation of the chain considered. From this model, we can arrive at the *freely rotating chain model* which restricts the bond angles in 3D space but allows free rotation about one axis [see Figure 1.7 *(right)*], and the *wormlike chain model* which is discussed in Section 2.1.1. In real polymer chains, the interaction between distant points on a chain should also be considered. This is the motivation behind real chain models that account for excluded volume interactions and self-avoiding walks. From these single chain conformation models it is possible to derive models

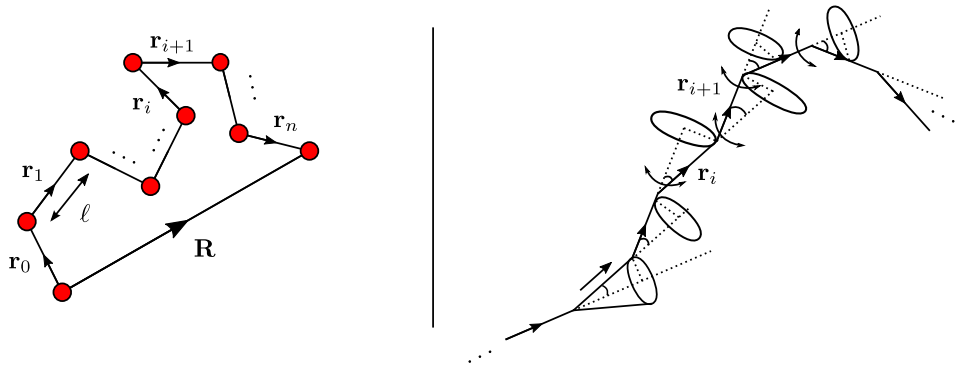


Figure 1.7: A polymer modelled as freely jointed chain *(left)* and freely rotating chain *(right)*. Adapted from [73, p. 51] *(left)* and [18, p. 10] *(right)*.

that account for single chain dynamics or collective network behaviour. To describe dynamics, there is the *Rouse* model that excludes hydrodynamic interactions and the *Zimm* model which accounts for them [73, pp. 311–314]. Network models can treat polymers as unentangled, as in the case of the *affine* and *phantom* models [73, pp. 255–263], or entangled, as in the case of the *Edwards Tube* model [73, p. 264]. The affine model takes the assumption that the network deforms affinely, such that the end points of each network strand are displaced relative to the macroscopic network deformation. The deformation of each network strand is predicted by the macroscopic strain. The phantom model takes this a step further by allowing segments to fluctuate freely around their average positions at their points of intersection with other polymer segments.

1.3 Biopolymers

Many biopolymers that make up much of living matter represent a class of materials that behave similarly to conventional soft matter polymers - in particular, showing softness in the form of a weak resistance to deformation. However, these biopolymers also demonstrate other less typical properties such as a strong stiffening under an applied deformation [45, 56, 80]. Many biopolymers differ from conventional polymers in that they are *semiflexible*. For the purposes of this chapter, we take this to mean that their elasticity is similar to a rigid rod for short fibres and flexible for long fibres, and direct the reader to Chapter 2, for a more formal discussion.

1.3.1 Cytoskeletal Components

We begin this discussion at the length scale of the basic unit of all living organisms - the cell. The two primary groups of cells are prokaryotes and eukaryotes, both of which have many

more subgroups. Mammalian cells are eukaryotes that come in a variety of types, each with their own set of mechanical properties, which are largely dictated by the cytoskeleton [56]. The cytoskeleton is an extremely dynamic and complex network of randomly oriented fibres that demonstrates remarkable plasticity. It is responsible for changing the shape of a cell, as well as organising cell components for cell division. Each of the constituent fibres come from three main groups that have a unique and rich set of properties. These are actin filaments made from actin protein, intermediate filaments made from various related fibrous proteins and microtubules made from tubulin. These three fibre types are often sufficiently percolated in vivo to form distinct and independent networks in their own right (and are in practice often studied as separate entities [45, 50, 56]), but in reality intermediate filaments and microtubules are connected via direct protein links known as plakins, which also connect the microfilaments to the intermediate filaments, leading to a single entangled and highly complex network. It is this collective protein scaffold that is responsible for such an interesting set of mechanical properties of the animal cell. The interactions of each cytoskeletal component are clearly an important consideration when building a complete picture of cell mechanics, with some composite network models in the literature (see Section 2.4) and some attempts at multiscale models [2]. More generally however, a common modelling simplification is to consider the constituent filament networks in isolation. To motivate this, we now provide a summary of each of the main fibre types.

Actin filaments

Actin exists in two forms, as globular G-actin or as filamentary F-actin. The monomeric state of G-actin can be polymerised to give F-actin in its double-stranded helical form as a filament with a diameter ranging between 7nm-9nm and an average contour length of $\sim 15\mu$m (see Figure 1.8 *(left)* and [9, 57, 83]). This polymerisation can be split into a three phase assembly process,

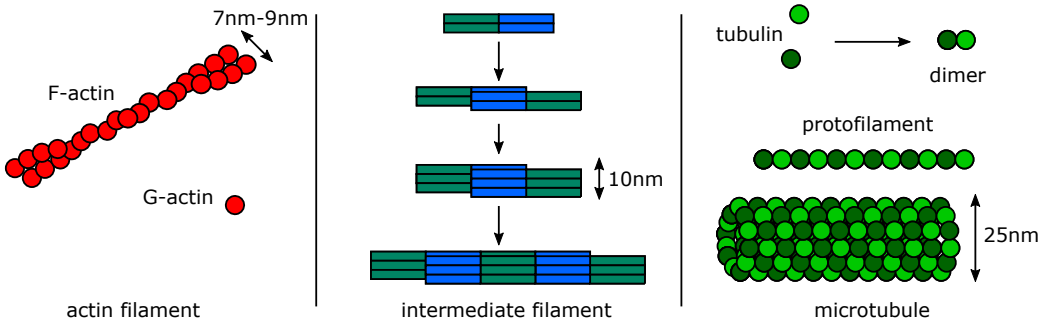


Figure 1.8: A visual comparison of the three major filament types found in the cytoskeleton. Adapted from [23].

consisting of a lag phase where nucleation occurs, a growth phase where the filament elongates,

and a final phase where the length reaches a steady state [57]. F-actin can be organised into bundles or entangled and cross-linked into networks by a variety of actin binding proteins. The actin network that forms near the membrane of the cell is known as the actin cortex. In its filamentary form, actin plays a role in cell locomotion, cell contraction, and is also responsible for specifying and maintaining cell shape [6, 22, 45, 68].

Intermediate filaments

Intermediate filaments are responsible for the mechanical strength of cells and, after being observed in skeletal muscle cells, were named due to their intermediate diameter (approximately 10nm) that lies between actin and the muscle filaments known as myosin [43]. Now they are also commonly considered intermediate between actin and microtubules [28] [see Figure 1.8 *(centre)*]. They are highly ductile but flexible, spanning the entire cell, and even anchoring the membrane junctions known as desmosomes that connect tissue cells. Unlike intermediate filaments found in the nucleus (i.e. lamins), cytoplasmic intermediate filaments vary depending on tissue type and embryonic origin, but all share a basic set of properties such as diameter (after assembly). In humans, there are approximately 70 different gene encodings for the intermediate filament protein family [82], with some examples of filaments being vimentin seen in connective tissues, neurofilaments seen in nerve tissue, and keratin seen in epithelial cells [1, p. 41]. Intermediate filaments also play an important role in the mechanics of extracellular secretions such as finger nails and hair, present in the form of harder keratins.

Microtubules

Historically, microtubules were observed to be key components along with the motor protein dynein in the movement of cilia and flagela, only later to be understood as also an important part of the cytoskeleton, particularly for intracellular transport and cell reorganisation. This is due at least in part to their highly dynamic nature - microtubules can collapse and reform in a matter of seconds. Advances in the ability to inhibit the assembly and disassembly of microtubules led to the understanding that the filaments are critical for intracellular transport, moving matter such as vacuoles across a cell by providing rails for motor proteins such as kinesin to walk along. Structurally, microtubules are essentially hollow tubes with a diameter of 25nm, most commonly assembled from a collection of 13 protofilaments [67] [see Figure 1.8 *(right)*]. Pairs of tubulin molecules form dimers, which in turn bond end-to-end to form the protofilaments. In cultured cells, microtubules have been observed to develop outwards from the centrosome, which is the cellular subunit responsible for the organisation and distribution of microtubules.

1.3.2 Collagen

We have seen above that intermediate filaments can provide mechanical strength beyond the length scale of a cell, playing an important role in the joining of neighbouring cells into tissues and giving structural integrity to excreted extracellular matrices. Another important biopolymer that exists at the scale of tissues is *collagen*. Collagen forms one of the other types of fibres found in the extracellular matrix, alongside *fibrin* and *elastin*, and is one of the primary components in connective tissues. For the connective tissue of mammals, the role of collagen is to provide strength, elasticity and structural stability [70]. In humans, there are 28 known types of collagen, with *type I* being the most prevalent. Collagen fibrils are assembled from single molecules of collagen that stagger with respect to each other and align parallel to the fibril axis. The resulting fibrils have a diameter in the range of 100-500nm, with molecule staggering leading to regular repeating overlaps of length 31nm and gaps of length 36nm. Notably, unlike biopolymers such as microtubules, collagen fibrils are structurally anisotropic.

1.4 Outline of the Thesis

A Review of Elastic Fibre Network Modelling

Having motivated the topic of this thesis with the main applications of network modelling in this chapter, we continue into Chapter 2 with a deeper look at some of the existing research relevant to the modelling of elastic fibre networks. The majority of the research reviewed in this chapter is themed around biopolymers and focuses on semiflexible networks, but the approaches for modelling these biopolymer networks are also applicable to more general elastic fibre networks. The discussion begins with the standard choices of model for single fibres, as well as a brief overview of bundled fibre modelling. Network modelling is then covered with a focus on isotropic networks, but with some later attention also being given to anisotropic and composite networks. The network models discussed here include both lattice based and off-lattice, where the significance of the distinction between these two network types is covered in the most detail in Section 2.2.2 on cross-linked networks. The anisotropic network models discussed here have some relevance to the usual biopolymer applications, but are of additional interest to the modelling of nonwoven fibre networks.

Elastic Fibre Network Methodology

Chapter 2 concludes the background required to begin introducing our numerical model used for generating the mechanical property predictions of 2D and 3D networks covered in Chapter 5 and Chapter 6. The discussion of the numerical model and the related methodology spans two chapters, with Chapter 3 focusing on the model derivation and discretisation, and Chapter 4 centred around the properties of the global matrix and techniques utilised for solving the global matrix system.

The Mechanical Properties of Elastic Fibre Networks

The primary contributions of this work (including the results published in [36]) follow from the use of our numerical model to predict the mechanical properties of elastic fibre networks. To present these contributions, we split the discussion into 2D network modelling in Chapter 5, and 3D network modelling in Chapter 6. In both of these chapters, we begin with the modelling decisions we took to give our discretised representations of the respective networks, in addition to an overview of some the test networks used for validating the model. The results presented in Chapter 5 are centred around a preconditioned solver performance analysis when used with various network types. The results presented in Chapter 6 look at mechanical properties of 3D networks, in particular those inspired by anisotropic layered materials.

Chapter 2

Semiflexible Fibre Network Models

Up to this point, we have provided some background relevant to the thesis and discussed some of the applications appropriate for the work of later chapters. In this chapter we move to a more focused look at the existing approaches to fibre network modelling that motivated development of the model that will be discussed in the chapters that follow. To do this, we begin the chapter with a review of well established single fibre mechanics and follow with an evaluation of what is currently known with respect to collective fibre mechanics. Discussion of collective fibre behaviour will focus on isotropic networks, with a brief review of anisotropic and composite networks.

2.1 Single Fibre & Fibre Bundle Models

Flexible polymer theories are not sufficient for modelling fibrous materials such as those seen in cellular cytoskeletons because they fail to describe some of the properties that distinguish them from conventional polymers such as their large shear moduli, non-linear response and unique dynamics. Alternative models based around the semiflexible characteristics of these fibres are now well established [7, 8, 49, 90]. Although the focus of the chapter is the collective behaviour of fibre networks, we first overview fibres modelled individually.

2.1.1 Wormlike Chain Model

A simple but commonly used model for a single fibre is known as the *wormlike chain* (WLC) model. This models a biopolymer as a slender elastic body that is inextensible and has finite

resistance to bending. This bending energy is given as

$$\mathcal{H}_{\text{bend}} = \frac{\kappa}{2} \int ds \left| \frac{\partial \hat{\mathbf{t}}}{\partial s} \right|^2 , \tag{2.1}$$

for bending modulus, κ, and unit tangent vector, $\hat{\mathbf{t}}$, along the fibre. For an arc-length coordinate s, the position along the fibre, $\mathbf{r}(s)$, can be used to define the tangent vector as

$$\hat{\mathbf{t}} = \frac{\partial \mathbf{r}}{\partial s}. \tag{2.2}$$

For a Young's modulus, E^f, and moment of inertia, I, the bending modulus κ can be written as

$$\kappa = IE^f, \tag{2.3}$$

where E^f defines the relationship between longitudinal tensile stress and strain of the given fibre [73, p. 296]. It is important to note that here we are assuming a single bending modulus, which holds in 2D where fibres are not permitted to deform out of plane, but only holds in 3D if the cross-sectional profile of the respective fibre is rotationally symmetric. More generally there would be two different bending moduli. For example, for a non-rotationally symmetric cross-section, such as a rectangular cross-section, there are two different moments of inertia and therefore two bending moduli [92].

Definition 1. Taking the fibre to be a slender elastic object with rotational symmetry, we can use results from classical elasticity theory [51] to derive the *bending modulus* κ in terms of the Young's modulus and the radius

$$\kappa = \frac{AE^f r^2}{4}, \tag{2.4}$$

for $A = \pi r^2$.

Some typical values of the bending modulus, κ, for different biopolymers can be found at [48, p.473]. For example, κ can range between approximately 10^{-27}nm^2 - 10^{-25}nm^2 for an actin filament, and between 10^{-29}nm^2 - 10^{-23}nm^2 for an amyloid fibril.

Definition 2. The *persistence length*, ℓ_p, characterises the stiffness of a semiflexible polymer undergoing thermal fluctuations, and can be written in terms of κ as [58]

$$\ell_p = \frac{\kappa}{k_B T}, \tag{2.5}$$

for Boltzmann constant k_B and absolute temperature T.

Measurements for various biopolymer persistence lengths can be found at [9, 83]. For example the persistence length of an actin filament ranges between $\sim 2\mu\text{m} - 30\mu\text{m}$. For fibres under thermal equilibrium, we can define the persistence length as the length at which the local

tangent to the polymer backbone becomes uncorrelated with itself. This can be written as [58]

$$\langle \hat{\mathbf{t}}(s) \cdot \hat{\mathbf{t}}(s') \rangle = e^{-|s-s'|/\ell_p}. \tag{2.6}$$

2.1.2 Force Extension & Extensible Fibre Model

While the WLC is a useful base model, semiflexible polymers are capable of a rich elastic response that cannot be completely described by bending alone. In addition to bending, individual fibres can also respond to transverse and longitudinal forces through stretching and compression, or even through twisting in 3D (although torsional responses are usually not considered). To account for stretching and compression, we begin by considering a length scale much shorter than ℓ_p. At this length scale, whereas bending can be expressed in purely mechanical terms, stretching and compressive behaviour can be described as entirely mechanical, entirely entropic, or a combination of both. We focus on a zero-temperature, purely mechanical model, but models that include thermal contributions have also received attention in the literature [30, 39, 58]. A standard example is a model for the force extension behaviour of a fibre, with a Hamiltonian

$$\mathcal{H} = \frac{1}{2} \int dx \left[\kappa \left(\frac{\partial^2 u}{\partial x^2} \right)^2 + \tau \left(\frac{\partial u}{\partial x} \right)^2 \right] \tag{2.7}$$

for chain tension τ and transverse coordinate $u(x,t)$ as a function of time, t, which represents the displacement from the average orientation of the chain (see Figure 2.1). We note this average orientation is simply taken to be the x axis, and that u is one of two independent transverse degrees of freedom.

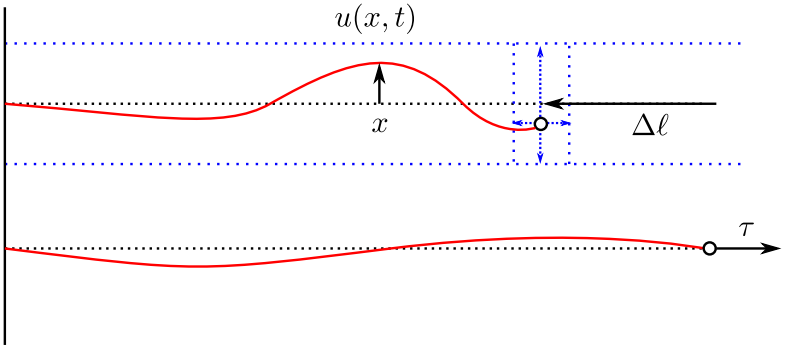


Figure 2.1: Fixing one end of the fibre and leaving the other end free, the transverse displacement field is characterised by $u(x,t)$. Thermal fluctuations contract the end-to-end distance by $\Delta\ell$. Fluctuations are reduced when a tension, τ, is applied *(bottom)*. Adapted from [9].

Now moving to a purely mechanical model, regarding a single fibre as an elastic athermal

rod, an effective model for the energy due to stretching or compression is

$$\mathcal{H}_{\text{stretch}} = \frac{\mu}{2} \int ds \left(\frac{d\ell(s)}{ds} \right)^2, \tag{2.8}$$

where μ is the stretching modulus, and $d\ell/ds$ represents the relative change in the length of the fibre [58].

For a simple Hookean spring, μ can be normalised with the segment length, ℓ, to produce a spring constant k

$$k = \frac{\mu}{\ell}.$$

Definition 3. Treating the fibre as a slender elastic body, we can use results from classical elasticity theory [51] to derive the *stretching constant*

$$k = \frac{AE^f}{\ell}, \tag{2.9}$$

where $A = \pi r^2$ is the cross-sectional area for segment radius r.

Care should be taken in deciding when it is valid to neglect thermal fluctuations, and thus treat fibres as truly rod-like, since it has been raised [58] that in some cases fibres can be dominated by entropic behaviour, even when fibres are much shorter than the persistence length.

2.1.3 Wormlike Bundle Model

Under the influence of certain cross-linking proteins, F-actin fibres can join to form thick hierarchical bundles [22]. These bundles serve as important structural components for live cells in a collection of contexts (filopodia, sensory hair cell bristles, microvilli) [16], and it can be argued that single fibre models such as the WLC model do not account for the additional complexities of such structures. In particular, any internal deformation that a bundle may experience, such as relative sliding or twisting of the individual fibres is not accounted for by the WLC model. Prior to considering the collective behaviour of fibres at the scale of networks, we first give a brief overview of how fibres can be modelled as bundles, along with the additional considerations needed to reflect the change in behaviour of fibres grouped in this way.

To better describe the mechanical properties of bundled fibres such as those seen in various cytoskeletal structures, the *wormlike bundle* (WLB) model was proposed [6, 29, 32]. Here, bundles are described as a series of N parallel fibres of length ℓ with a constant spacing b [see Figure 2.2 *(left)*]. Fibres are permanently cross-linked with an axial spacing of δ, where cross-linking occurs between nearest neighbouring fibres. Each fibre is described as an extensible wormlike polymer with bending and stretching moduli, κ_b, k_s, respectively, in addition to a twist rigidity, κ_t. To model the cross-links a finite shear stiffness, k_x, is used to describe

shearing along the bundle axis, but no stretching modulus is needed since the cross-links are modelled as inextensible transverse to the axis of the bundle.

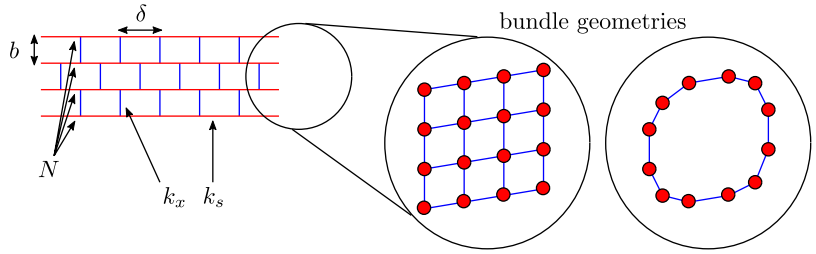


Figure 2.2: A schematic of a wormlike bundle. Bundles of N filaments (red) of length ℓ and stretching stiffness k_s are cross-linked to other nearby filaments, of constant spacing b, by short rod like springs (blue) with axial spacing δ and stiffness k_x *(left)*. Square and cylindrical lattice bundle geometries are illustrated by showing the cross-section of the bundle *(right)*. Adapted from [6, 32]

Depending on the material of interest, Heussinger et al.[32] propose two alternative architectures for the arrangement of the bundle fibres [see Figure 2.2 *(right)*]. The first is motivated by F-actin filaments, and is assembled by assuming that fibres are arranged on a square or triangular lattice, such that the cross-section of the bundle is homogeneous. The second geometry is inspired by microtubules, and is constructed as a cylindrical bundle with fibres arranged such that their cross-sections are evenly spaced around the perimeter of the cylinder's circular cross-section.

Prior to an applied load, cross-links lie parallel to each other and perpendicular to the fibres. After a load is applied to the bundle, cross-link response can be measured using

$$\alpha = \frac{k_x \ell^2}{k_s \delta^2}, \tag{2.10}$$

which describes the competition between the stiffness of the fibres and the stiffness of the cross-links. This value α, determines the degree of coupling of the cross-links, such that $\alpha \ll 1$ is a decoupled regime where fibres can slide past each other, and $\alpha \gg N$ is a tightly coupled regime where fibres are fixed relative to each other, for a bundle of N fibres.

Heussinger et al.[32] found that the pretwist or helicity of a bundle strongly influences the strength of the cross-link induced filament coupling. In particular, bending rigidity decreases as pretwist increases, which has implications for different types of microtubules, where variations in the number of protofilaments, N, lead to differences in the bundle pretwist.

2.2 Isotropic Network Models

Strongly motivated by the biopolymer applications of collagen or cytoskeletal structures, the majority of the existing network models for semiflexible fibres assume the fibrous materials to

be modelled are isotropic and many also assume the materials are homogeneous. It follows that much of the discussed literature can be contained in this section. Before continuing, we elaborate on the terms isotropic and homogeneous.

Definition 4. A network is *isotropic* if there exists no common directionality to the constituent fibres. That is, there is no orientation in which fibre alignment is clearly prevalent. Conversely a network is *anisotropic* if there is a preferred direction or directions after averaging.

Definition 5. A network is *homogeneous* if the density and distribution of the constituent fibres are consistent throughout the domain the network is contained in.

We begin this section with a look at the approaches for modelling fibre components combined into isotropic networks. The natural progression from single and bundled fibre models is to entangled fibre solutions. After giving a short overview of entangled solution models, we move to the main discussion of this chapter: cross-linked network models. This discussion covers both permanent and dynamic cross-linking models, with a further division of permanent cross-linking into 2D and 3D approaches. We then discuss how the mechanics of networks can be characterised in terms of elastic moduli and total energy, before exploring the role of connectivity on network properties through the introduction of some relevant measures. We conclude the section with a review of the mechanical properties of fibre networks obtained using various models.

2.2.1 Entangled Solutions

In the sections that follow, we will focus on models that make the simplification of treating the semiflexible polymer networks as though they are in a vacuum, which is valid for the static response since the fluid only contributes at finite frequencies, but for context and completeness, here we briefly review modelling networks in solution. An approach to modelling semiflexible polymers in concentrated solution is to consider collections of polymers entangled together and then model their interactions. Entanglement models assume thermally fluctuating polymers, and unlike the permanently cross-linked networks discussed in Section 2.2.2, entanglements are assumed to be transient. A consequence of this is that the networks will relax at sufficiently long times, leading to a fluid-like material at zero frequency, rather than a solid.

To better understand entanglement interactions, it is useful to introduce an important length scale, ξ, known as the *mesh size*, which in the context of polymer solutions is the average spacing between the polymers. Recall that the persistence length ℓ_p characterises the stiffness of a polymer, then for $\ell_p \gg \xi$, polymers in solution can be treated as rigid rods. In this limit, the mesh size can be shown to depend on the polymer cross-sectional radius, r, and the volume fraction, ϕ, such that $\xi \sim r/\sqrt{\phi}$. When polymers in solution are close enough to interact, they are said to be entangled, and the length at which this occurs is the *entanglement length*, which is the analogy of the cross-link length, ℓ_c, to be defined later for permanently cross-linked

networks (see Section 2.2.4). For semiflexible polymers the entanglement length can be given as [9, 58, 61]

$$\ell_e \sim (r^4 \ell_p)^{1/5} \phi^{-2/5}. \tag{2.11}$$

The entanglement length is a key quantity taken from flexible polymer physics for predicting the shear modulus of a network, and is especially important in experiments where the network length cannot usually be extracted directly. Existing theories for characterising the plateau modulus G^0 (which is a near constant value of the modulus that exists over a wide frequency known as the rubbery plateau - see Figure 2.3), can be split into two scaling law predictions. Many theories [42, 44, 58] predict

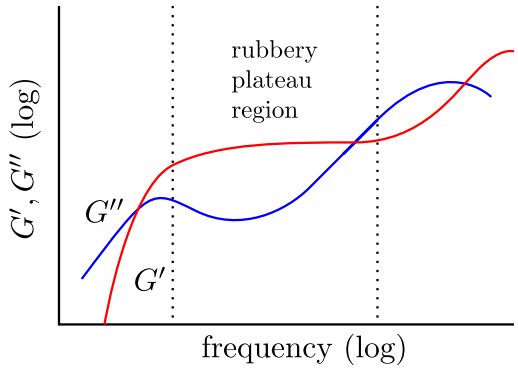


Figure 2.3: A schematic representation of the frequency dependent storage modulus G' and loss modulus G''. The plateau modulus G^0 is defined as the near constant value over the frequency known as the rubbery plateau. Adapted from [73, p. 365].

$$G^0 \propto c^{7/5} \ell_p^{-1/5}, \tag{2.12}$$

for concentration c and persistence length ℓ_p, but competing theories [61] have suggested

$$G^0 \propto c^{4/3} \ell_p^{-1/3}. \tag{2.13}$$

In work on tightly entangled isotropic semiflexible polymer solutions, Tassieri [83] brought together the previous results of Tassieri et al.[84, 85] and new results of Schuldt et al.[77] to identify an inconsistency in theoretical predictions and experimental findings for the scaling law of the plateau modulus G^0. Specifically, Tassieri presents experimental evidence demonstrating that neither $G^0 \propto c^{7/5} \ell_p^{-1/5}$ nor $G^0 \propto c^{4/3} \ell_p^{-1/3}$ fully describe the dynamics of semiflexible polymer solutions in the tightly entangled regime. Instead, the exponent of ℓ_p is observed to be a positive value between 1 and 5 depending on currently unknown factors.

2.2.2 Cross-linked Networks

As we have seen in Section 2.1.3, in many applications fibres bond together forming cross-links. The exact mechanism responsible for fibre bonding is very much dependent on the material, from thermal bonding of nonwoven fabrics to a variety of cross-linking proteins seen in the cytoskeleton. Here we review the different modelling approaches for cross-links, beginning with simple 2D cross-linking before graduating to the additional complexities introduced in 3D, and conclude with an overview of cross-links modelled dynamically.

Permanent Cross-links in 2D

In two dimensions, non-parallel fibre pairs are typically cross-linked at their unique point of intersection. A common modelling assumption for cross-linking is to treat the cross-links as permanent bonds between the respective fibres. This simplified approach removes some of the complexities of dynamic cross-link modelling and is appropriate for zero frequency models where dynamics do not come into play [25, 26].

Assuming fibres are permanently bonded, cross-links can be further categorised by their rotational mechanics. Head et al.[25, 26] defined their cross-links as freely rotating, whereas Onck et al.[66] constrained cross-links to fixed angles. Others have directly compared the difference between fixed angle and freely rotating cross-links [30, 91]. The validity of freely hinged cross-links can be criticised for neglecting the additional rigidity that would result from fixed bond angles, but Head et al.[26] argue that, for the isotropic networks their model applies to, the effect on the mechanics is small when networks are dilute. When Wilhelm and Frey compared fixed angle and freely rotating cross-links for similar networks [91], they found that the rigidity percolation threshold (discussed in Section 2.2.4) occurs at a different fibre density, but beyond this threshold they also found no significant differences to the mechanical properties.

Freely rotating and fixed angle cross-links can be seen as the two limits of a more general approach to modelling the rotational degree of freedom of fibre intersections. In their model of thermally fluctuating semiflexible polymer networks, Heussinger and Frey [30] accounted for this rotational degree of freedom by defining an energy cost $W_{\text{rot}} = m(\phi - \phi_0)^2$ depending on the change in relative cross-link angle ϕ from their initial angles ϕ_0. It follows that by taking $m \to 0$ or $m \to \infty$ they were able to compare the respective limiting cases of freely rotating cross-links with a zero energy cost and cross-links with an infinite stiffness that inhibits a change in angle.

In addition to cross-linking at the intersection point of two fibres, some models have also introduced further nodes at the midpoints of segments connecting two adjacent cross-links [60], which allows for an improved resolution of bending behaviour at the cost of additional computation.

Permanent Cross-links in 3D

In three dimensions, non-parallel fibres are increasingly unlikely to intersect directly for decreasing fibre radius, r. Intervention is therefore required to encourage a cross-link density representative of real-world materials. A number of approaches have been taken in the literature, but we can loosely group these into lattice and off-lattice driven methods.

Lattice based cross-linking is arguably analogous to 2D cross-linking - fibres are arranged over lattice points in such a way that they cross paths. It follows that these fibres are cross-linked directly and a new node is introduced at the point of intersection. In an attempt to reveal the relationship between connectivity and network elasticity in 3D, Broedersz et al.[11, 12] developed a model that arranged fibres on a face-centred cubic (FCC) lattice. To reduce rigidity, fibre segments were diluted with a probability $1 - p$, since an undiluted FCC lattice has six fibres intersecting at each point, placing the undiluted structure in a highly constrained regime. More formally we can give this in terms of the coordination number as $z = 12$, where

Definition 6. The *coordination number*, z, is defined as the number of fibre segments immediately connected to a given node, also sometimes referred to as the node *degree*.

Off-lattice cross-linking invites more creativity, as fibres are typically arranged with a random orientation such that direct intersections are extremely improbable. Huisman et al.[40] took inspiration from molecular dynamics to craft their cross-linking procedure. Fibres were drawn together using an attractive force field $1/r^p$, for distance r between fibres and choice of $p \approx 2$. Each fibre was split into Bernoulli beam elements, such that nodes were inserted at the end points of each element. On the approach of two fibres, node pairs within a sufficiently close proximity were cross-linked permanently and prevented from forming any further cross-links. By varying the force-field properties, Huisman et al. argued they were able to achieve architectures visually similar to real-world actin networks.

Zagar et al.[95] used the dynamics procedure of Huisman et al.[40] to assemble random 3D networks of athermal filaments. They then analysed these networks in an attempt to relate the network shear modulus with network connectivity. After removing dangling ends and isolated fibre clusters, they proposed that all cross-links could be categorised into three types, L, T, and X, characterised by coordination numbers of 2, 3 and 4 respectively.

In further work using a Monte Carlo scheme to determine free energy minimising network configurations, Huisman et al.[38, 39] and Huisman and Lubensky [37] were able to circumvent the off-lattice issue of intersection rareness for random fibre configurations. They accomplished this by generating their nodes first and associating fibres with the respective nodes later. Specifically, for a randomly distributed set of m nodes, they identify a triplet of nearby nodes and connect them into a loop with three bonds. From this configuration, they expand connectivity to nearby nodes while maintaining the closed loop, by deleting an old bond and adding two new bonds (see Figure 2.4). The end result is a fully connected network consisting of a single fibre that crosses itself at each node such that cross-links are binary ($z = 4$). The networks resulting

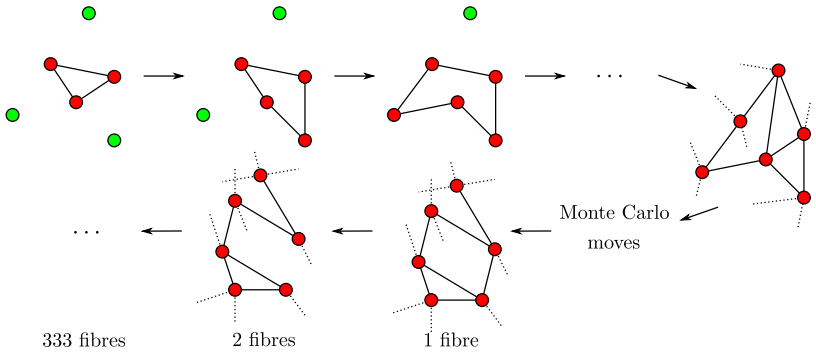


Figure 2.4: A demonstration of how Huisman et al.[38] produce their initial network topology. The closed loop is gradually expanded until the entire network is connected into one self-crossing loop. Next a series of Monte Carlo moves are performed resulting in changes to fibre segment angles and cross-link coordinates. After the Monte Carlo moves have been made the continuous fibre is split into the desired number of fibres.

from this procedure are typically highly stressed, so random Monte Carlo moves are then taken to alter the topology further, and changes are subject to a Metropolis acceptance criterion. At the point that Monte Carlo moves no longer significantly affect the total free energy, the continuous single fibre loop is finally split until the desired number of fibres is reached. This approach to network generation has clear advantages over lattice based methodologies, allowing for binary cross-links while maintaining an architecture of randomly oriented fibres with off-lattice node placement. The disadvantage of such a procedure is the slow computation time required to generate moderate network sizes - a network of 333 fibres, with 1000 cross-links took 1 week of cluster time [39].

Kim et al.[47] compared experimental data with simulations of actin filaments cross-linked with two different cross-linking proteins. In their model [46], they dispersed a uniform arrangement of monomers and a random arrangement of cross-linking proteins. They then employed Brownian dynamics to simulate the motion of the monomers, filaments, and cross-linking proteins, continuing until a polymerisation of 99% was reached. In this molecular scale bead-spring model, each of their monomers represented two G-actin monomers, and therefore had two cross-linker binding sites. To cross-link the F-actin filaments, they used actin cross-linking proteins (ACPs) which cross-linked fibres such that they formed ladder-like bundles, and ACPs which cross-linked fibres orthogonally. Using this fine-grained model, Kim et al. suggested that their resulting network geometries were more representative of reconstituted actin gels than networks produced through random deposition of equal length fibres. While true that their model demonstrates a greater fibre length distribution than some similar approaches, the level of detail used for the polymerisation dynamics could lead to difficulties when generalising to fibrous materials beyond actin.

Dynamic Cross-links

The cytoskeletons of mammalian cells are highly dynamic structures, using communities of cross-linkers to rearrange fibres in such a way that the mechanics of the network can suit the needs of a changing cell shape. While permanent cross-links may be sufficient for modelling a static, linear response to an imposed strain, dynamic cross-link models [10, 62, 94] attempt to address fibre bonding and bond breaking behaviour.

Astrom et al.[94] investigated dynamic cross-linking by comparing networks with three different cross-linking schemes. The first of these was a control group of permanent cross-links inserted between nearby fibres. The second scheme allowed for cross-link breakage when the distance between cross-linked fibres exceeded a given tolerance as a result of the deformation due to a macroscopic load. The final scheme allowed for both cross-link breaking and forming where, similar to cross-link breaking, cross-link forming was determined via comparison of distance to a given tolerance.

To form the cross-links, Astrom et al. discretised their fibres into equally spaced segments prior to connecting fibre pairs. Consecutive segments along a fibre were separated with a node. Then calculating the minimum distance between each fibre, nearby fibres were connected together between the nearest pair of existing nodes. One could argue that the lack of variation in the segment length distribution compared to similar 2D models [25, 30, 66, 91] leads to networks that are less representative of cytoskeletal networks. However, with careful parameter selection, this approach can be easily tuned to avoid badly conditioned systems arising from short segment lengths (see Section 5.2.3).

Other work on dynamic cross-link modelling includes that of Mulla et al.[62], who found that transiently cross-linked actin networks are described well by a simple model combining features of viscoelastic transient gel modelling with nonlinear permanently cross-linked network modelling, and that of Broedersz et al.[10], who developed a cross-link-governed dynamics model to investigate the effect of multiple independent unbinding and rebinding events on the structural relaxation of thermalised networks.

2.2.3 Characterising Network Mechanics

Having discussed the variety of approaches that exist for modelling the cross-link behaviour of semiflexible fibre networks, we move to a brief discussion on how the network mechanics can be characterised through the elastic behaviour of its constituent fibres. Here we focus on the elastic moduli and network energy that are used to describe the Mikado model, which we will derive more formally in Chapter 3.

Elastic Moduli

Both Head et al. and Wilhelm and Frey [25, 91] took an athermal approach to characterising the elastic behaviour of their Mikado networks, using a stretching/compression modulus, μ, and

bending modulus, κ. Wilhelm and Frey absorbed these elastic moduli into the force constants $k_{\text{comp}}(\ell) = \pi r^2 E^f/\ell$ and $k_{\text{bend}}(\ell) = (3/4)\pi r^4 E^f/\ell$. The force constant that describes the longitudinal thermally undulated excitations that was neglected from these models can be given as $k_{||}(\ell) = 6\kappa^2/(k_B T \ell^4)$ [91]. Despite the athermal simplification of the stretching behaviour, Wilhelm and Frey argue that the essential feature of compressional stiffness dominating bending stiffness is still captured for representative network densities. This can be seen by observing the ratio of their stretching/compression and bending force constants

$$\frac{k_{\text{comp}}}{k_{\text{bend}}} = \frac{\pi r^2 E^f/\ell}{(3/4)\pi r^4 E^f/\ell^3} = \frac{4\ell^2}{3r^2} \gg 1. \tag{2.14}$$

In addition to bending and stretching moduli, some models [40, 46, 78] have also included a torsional rigidity ω. Kim et al.[46] justified this inclusion for their actin model through the argument that torsional stiffness is comparable to bending stiffness [86, 93], though Huisman et al.[40] only found a negligible contribution to their total energy ($\leq 2\%$).

Network Energy: The Mikado Model

A simple model to capture the elastic behaviour of athermal disordered whole networks in 2D is the *Mikado* model [25, 26, 91]. In this model networks are generated by depositing the desired number of fibres of length ℓ with random position and orientation on a plane with area $A = W^2$, for a given plane width W. When viewed at the macroscopic length scale, these networks are regarded as *isotropic* and *homogeneous*. The discrete form for the total energy of the Mikado system can be expressed as

$$E = \frac{\mu}{2} \sum_{ij} \frac{\delta \ell_{ij}^2}{\ell_{ij}} + \frac{\kappa}{2} \sum_{\langle ijk \rangle} \frac{\delta \theta_{ijk}^2}{\bar{\ell}_{ijk}}, \tag{2.15}$$

for length ℓ_{ij} of segment ij, average length $\bar{\ell}_{ijk}$ of adjacent segments ij and jk belonging to the same fibre, extension $\delta\ell_{ij}$ of segment ij, and angular deflection $\delta\theta_{ijk}$ between ij and jk (see Figure 2.5), where $\langle ijk \rangle$ are all the possible adjacent pairs of segments ij,jk on a fibre, and both summations span all fibres of the network. This athermal approach models the total energy of the network as the sum of bending energy and the purely mechanical stretching energy. The bending energy is based on a discretised and linearised form of the WLC model (2.1), and the stretching energy comes from a discretised and linearised form of the extensible fibre model described previously for elastic energy in (2.8).

2.2.4 Network Connectivity & Percolation

It is now understood that connectivity plays an important role in determining the mechanical properties of a network [11, 12, 26, 79, 91]. Connectivity characterisation of a given network is often influenced by whether the modelled network is lattice based or off-lattice. Lattice based approaches tend to measure network connectivity in terms of its *coordination number*, whereas

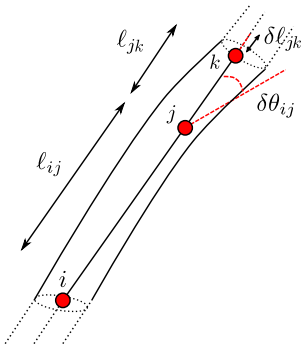


Figure 2.5: A fibre modelled as a slender elastic rod, discretised into segments.

off-lattice approaches may prefer other density based measures such as *cross-link distance* or *mesh size*. It is important to note however, that all of these respective quantities can be defined for *both* lattice and off-lattice geometries. We first define the coordination number and then follow with the off-lattice driven measures for connectivity. We also introduce the concept of rigidity percolation threshold and discuss it in the context of these different connectivity measures.

Coordination Number & Maxwell Counting

The regular structure of a lattice network is often exploited experimentally through bond dilution. This allows for easy manipulation of network rigidity and therefore presents a good opportunity for mechanical property investigation by simply controlling the network connectivity. A convenient measure for the connectivity in this case is the node coordination number, z, introduced in Section 2.2.2.

Maxwell [15] was one of the first to understand the significance of connectivity in the context of stabilising spring networks. In simple spring networks, each spring produces a *central force* that is dependent only on the distance between the two end points of the spring. Through his work, Maxwell found a constraint counting argument that can be used to estimate whether a given network can be considered mechanically rigid. Here we informally define *mechanical* rigidity to be rigidity in the absence of bending, and define *rigidity* to be a state in which no deformation is possible without the cost of energy. Following from Maxwell's argument, a fibre network is considered mechanically rigid if it satisfies

$$N_n d = N_f, \tag{2.16}$$

for network dimension d, large node count N_n, and large fibre count N_f. Noting that the fibre count can be expressed in terms of the node count and coordination number, and dividing by

2 to avoid double counting, we rewrite (2.16) as

$$N_n d = \frac{zN_n}{2}. \tag{2.17}$$

It follows that the point of central force rigidity can be predicted in terms of connectivity as

$$z_{\mathrm{cf}} = 2d. \tag{2.18}$$

In the lattice based experiments of Broedersz et al.[11], undiluted networks initially above this threshold ($z = 6$ for 2D triangular meshes and $z = 12$ for 3D FCC lattices) had bonds diluted with probability $1 - p$, to explore the role of connectivity in network mechanics. As dilution was increased they found critical values at which their networks cross the central force isostatic threshold. These values were $p_{\mathrm{cf}} = 0.651$ for their 2D networks and $p_{\mathrm{cf}} = 0.473$ for their 3D networks.

Network Density Measures

We have previously introduced the mesh size as a measure of network density in the context of entangled solutions (Section 2.2.1).

Definition 7. The *mesh size*, ξ, is defined as the mean separation distance between neighbouring fibres [26].

This measure is also relevant to cross-linked networks, but does not come into shear modulus calculations for athermal systems under linear response. This is because nearby (but not cross-linked) fibres will not interact and therefore have no effect on the mechanical properties. In this case it can be more appropriate to use a measure such as fibre density.

Definition 8. The *fibre density* of a network, ρ, is defined in 2D as

$$\rho = \frac{N\ell}{A} \tag{2.19}$$

for total number of fibres N_f, uniform fibre length ℓ, and network spanning area A (e.g. square box area $A = W \times W$ for edge length W) [91].

This can be extended to 3D by considering a network spanning volume V rather than area A. Alternatively, the cross-link density is another measure of network density that provides insight into overall network connectivity.

Definition 9. The *cross-link density* is defined as ℓ/ℓ_c, where ℓ_c is the mean distance between cross-links along a fibre of length ℓ [25].

Rigidity Percolation Threshold

In addition to the central force isostatic point, z_{cf}, there is another critical value, z_b, that occurs at a lower network connectivity $z_b < z_{\mathrm{cf}}$. This rigidity percolation point z_b marks the true onset for rigidity in fibre networks, where we define rigidity as above (see p. 29). Between this rigidity percolation threshold and the central force isostatic threshold, fibre networks - that could otherwise deform freely when simply modelled as central force springs - are stabilised through bending rigidity. The transition point at which a system becomes rigid can be deduced using counting arguments from Maxwell [52] as seen above for the onset of the isostatic point, but has also been observed using experimental models [11, 27, 91].

In their work on 2D triangular and 3D FCC lattice networks, Broedersz et al.[11] found the rigidity percolation threshold to occur at the respective bond dilutions of $p_b = 0.445$ for regular 2D triangular meshes and $p_b = 0.268$ for 3D FCC lattices. As expected, both of these dilutions correspond to network architectures with a lower coordination number, z_b, than that of the central force isostatic threshold $z_{\mathrm{cf}} = 2d$ for dimension d.

Through observation of a vanishing shear modulus G, Head et al.[25, 26, 27] found that the rigidity percolation threshold occurs at a critical value of $\ell/\ell_c \approx 5.9$, which is consistent with previous work of Latva-Kokko et al.[52] and Latva-Kokko and Timonen [53] who found a critical value of $\ell/\ell_c \approx 5.932$. Below this rigidity percolation threshold, there is a further *conductivity* percolation threshold at which a periodic network first becomes an infinite size, which Head et al. expect to occur at $\ell/\ell_c \approx 5.42$. However in their athermal study with freely hinged cross-links they were not able to test this, since to observe a nonzero G between the conductivity and percolation points one should either account for rotational energy at the cross-link sites or for thermal fluctuation within the fibres.

Wilhelm and Frey [91] found the percolation threshold to occur at a fibre density of $\rho_c = 5.71$ for fixed hinge 2D Mikado networks, and at $\rho_c = 6.7$ for freely hinged networks which also agrees well with the work of Latva-Kokko and Timonen [53] who found $\rho_c = 6.68$.

2.2.5 Mechanical Regimes

The anisotropic properties of individual fibres lead to a complex macroscopic elastic behaviour in the case of irregular network geometries. This macroscopic network elasticity that results from collective fibre interactions has both been described in terms of the ratio of bending-stretching behaviour and in terms of uniformity of localised deformation. We begin with a discussion of bending and stretching network mechanics, and then move to the concept of nonaffine and affine deformation.

Bending vs Stretching

Much work on isotropic networks has demonstrated that semiflexible networks sharply transition from a bending dominated regime to a stiffer fibre stretching dominated regime [25, 31, 91].

Wilhelm and Frey [91] used their 2D Mikado model to investigate the elastic behaviour of random semiflexible polymer networks. They found three distinct scaling regimes that they described in terms of the critical rigidity percolation region, an intermediate bending dominated regime and a stretching dominated regime.

Using a lattice based 3D spring model, Buxton and Clarke [14] found the bending to stretching crossover to be a geometry dependent transition that can be expressed as a function of average connectivity. Further support for this can be seen from Broedersz et al.[11], who found that while their rigidity threshold bond dilution probability p_b marked the onset of rigidity for a nonzero bending modulus κ, their value for the central force isostatic bond dilution probability p_{cf} marked the critical point at which mechanical behaviour crossed from a bending dominated regime to a stretching dominated regime.

Nonaffine vs Affine

Also known as the Cauchy-Born approximation in the context of crystalline solids, the affine assumption assumes that a macroscopic deformation imposed on a network can be translated in a uniform manner to each individual fibre at the microscopic level. Figure 2.6 gives a comparison of a discretised fibre deforming in an affine and nonaffine manner. While the affine assumption

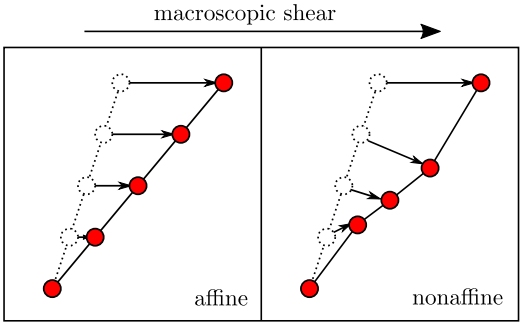


Figure 2.6: An example of a fibre discretised into smaller segments where nodes are deforming affinely *(left)* and nonaffinely *(right)*.

is sufficient for the modelling of ordered solids (such as an undiluted lattice structure), in disordered systems the deformation energy may be reduced by deforming nonaffinely.

For systems of interest, an elastic response to an imposed macroscopic strain that is exclusively stretching based and does not deform through fibre bending may also be characterised as a purely affine deformation. Conversely networks with a purely affine response to an imposed strain can be sufficiently described by deformation through longitudinal fibre elongation alone. The affine assumption breaks down when considering networks with geometries that place them somewhere between the rigidity percolation threshold and this homogeneously elastic regime. In this intermediate regime, deformation through bending is preferred in areas of the network where stretching deformation has a higher energy cost.

In their work on athermal 2D Mikado networks, Head et al.[25] found two distinct mechanical regimes, characterised by affine and nonaffine deformation. They argued that these regimes are primarily a function of cross-link density and filament rigidity, and that the crossover to the affine regime is observed as density or fibre length is increased. Additionally, they observed that the fibre segments between each cross-link act in series. This is distinct from flexible polymers where segments behave independently. Developing this further, nonaffine deformation is more prevalent close to less constrained free ends, and affine deformation is more prevalent far from free ends where they are more heavily constrained. Further work [26] accounting for thermal contributions showed that there are 3 distinct elastic behaviours above the rigidity percolation threshold. Specifically, Head et al. argued that the affine regime can be split into two further regimes of entropically dominated stretching behaviour and mechanically dominated stretching behaviour (see Figure 2.7).

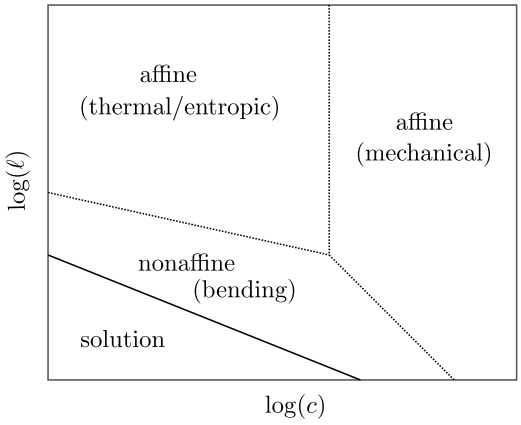


Figure 2.7: Adapted from [26], a schematic for the elastic regimes of semiflexible thermal polymer networks with binary cross-links, where ℓ is molecular weight and $c \sim 1/\ell_c$ is concentration.

2.3 Anisotropic Network Models

While many models take the assumption that the studied networks are isotropic and heterogeneous, real-world fibrous materials can show varying degrees of order. This is especially true of non-woven fabrics, where fibre alignment can occur during the manufacturing process - the extent of which is determined by the web formation and bonding methods as discussed in Section 1.1. To explore anisotropic network modelling, we first look at 2D networks with fibre alignment introduced, and then look at the phenomenon of network wrinkling, observed by applying a nonlinear shear strain to planar networks considered in 3D space.

2.3.1 Nematically Ordered Networks

Missel et al.[60] studied the effects of nematically ordering semiflexible fibres in networks. To set up their networks, they placed fibres with random position in a two dimensional box, and then introduced anisotropy by assigning an orientation from a distribution, $P(\theta)$, and nematic direction, $\hat{n}$. The anisotropy was characterised by a nematic order parameter, S, such that

$$S = \int_{-\pi}^{\pi} d\theta P(\theta) \cos 2\theta, \tag{2.20}$$

where $S = 0$ and $S = 1$ represent isotropic and fully aligned fibre networks respectively.

They found good agreement with isotropic networks for the affine to nonaffine crossover, but observed rich and complex mechanical behaviour within the nonaffine regime that is dependent on the shear direction. Additionally, they found that straining networks with the nematic direction in tension results in network stiffening, and softening with the nematic direction in compression, demonstrating the strong sensitivity of anisotropic networks to the orientation of applied loads. The authors only imposed small strains on their networks but suggested that at larger strains the effects observed should be more prominent and in agreement with experimental data.

2.3.2 Network Wrinkling

Using 2D sheets embedded in a 3D domain, Muller and Kierfeld [63] were able to study the effects of network wrinkling on mechanical properties. Their athermal model placed straight rods in a rectangular 2D simulation cell, either with random orientation or on a regular triangular lattice. Cross-links were formed at the sites of direct intersection between fibres, and were considered permanent but freely rotating. Fixing the lower y boundary of their domain, they applied an increasing amount of shear strain at the upper y boundary. Using the conjugate gradient algorithm, they were able to minimise the total energy of the network resulting from the applied strain. Their nonlinear model allowed for metastable network configurations, whereby the networks deformed in-plane and no wrinkling occured. To combat this and encourage wrinkle formation, the cross-links were randomly perturbed in the z-direction prior to energy minimisation. For comparison, the authors also studied the same networks without wrinkling (by skipping the z-coordinate perturbation step).

They found while both types of their networks used bending as a means to avoid compressive stress, the 2D networks that allowed wrinkling were able to bend over larger length scales since cross-links could move out of the plane. This results in less bending energy required for wrinkled sheets to avoid the same amount of compressive stress as sheets with no wrinkling permitted. This in turn results in a reduction (of up to 20%) in total elastic energy for wrinkled sheets. Intuitively this makes sense, since allowing nodes additional degrees of freedom to move increases the number of ways the total deformation energy can be reduced.

2.4 Composite Network Models

Much like the observation made in Section 2.3 that many models focus on isotropic networks, composite networks (in particular polydisperse fibre networks) have arguably received less attention than their monodisperse counterparts. Despite this, there are many examples of real-world composite networks where mixed fibre interactions could play an important role. An obvious example is that of the cytoskeleton, where there are typically three interleaved fibre networks (F-actin, intermediate filaments and microtubules), however there are also industrial examples of nonwoven fabrics that are manufactured with a blend of two or more fibre types with the purpose of enhancing certain properties [87, p. 7]. We begin this discussion with the extension of monodisperse fibre network models to polydisperse networks, and follow with a review of composites formed by embedding networks in an elastic medium.

2.4.1 Polydisperse Networks

Bai et al.[3] studied the effects of introducing varied densities of additional filaments into existing heterogeneous filament networks where the added impurity filaments were longer and stiffer than the soft matrix filaments of the original network. In their model, soft matrix filaments were assigned a bending modulus, κ, stretching modulus, μ, and length, ℓ. The stiff impurity filaments were described with the same stretching modulus, μ, but a bending modulus, κ_I, and length, ℓ_I, such that $\kappa < \kappa_I, \ell < \ell_I$. To assemble their networks, the authors dispersed soft matrix filaments with random orientation and position in a $W \times W$ box, cross-linking filaments at points of direct intersection. From this, they added varying densities of long stiff filaments, cross-linking these impurity filaments to existing soft matrix filaments, but not to themselves. To model the energy of their networks, the authors presented a discretised approximation derived from classical linear elastic continuum beam theory. Their discretisation process splits fibres into a collection of segments, introducing nodes at both cross-linking sites and at segment midpoints. The Lees-Edwards method was used to implement periodic boundaries, and then a quasi-Newton minimisation method was used to relax the energy resulting from an imposed shear deformation.

They found that even in cases of low impurity filament density, there was a considerable increase in the collective elastic moduli. The decision to leave impurity filaments uncross-linked with themselves was a deliberate design choice that prevented the additional filaments from producing a load bearing network if considered in isolation from the soft matrix filaments. Further experiments aiming to understand the relationship between impure fibre density and nonaffinity measures revealed that the stiffer filaments suppress nonaffinity locally, redistributing it to areas of the network where fewer added filaments are located.

2.4.2 Embedded Elastic Medium Composites

Another example of network composites existing in literature is that of networks embedded in an elastic medium [13, 88]. Doorn et al.[88] used lattice-based simulations to investigate the effect on mechanical properties of embedding a 2D fibre network in a softer polymer matrix. To do this, both the fibre network and the background matrix were arranged on a triangular lattice with regular spacing ℓ_0. To better understand the role of connectivity on the mechanical properties, the discrete segments of the embedded fibre nework were diluted with probability $1 - p$, such that the average connectivity was $z = 6p$. The fibre network was embedded in the soft matrix by joining the two networks at every vertex of the lattice. Applying a linear shear strain to the boundaries of both the fibre network and embedded matrix, the network was then relaxed by minimising the total mechanical energy, which could then be used to calculate the linear shear modulus G. The total energy is given as the total stretching energy of the elastic medium summed with the total stretching and bending energy of the embedded network. The stretching energy of the fibre network was calculated over each individual segment, with the stretching modulus given by μ_1, and the bending energy was calculated over consecutive colinear segment pairs, with bending modulus κ_1. The elastic polymer background matrix was treated as a homogeneous undiluted network of central force springs with a stretching modulus μ_2.

Using this model, the authors showed that there are a number of distinct mechanical regimes characterised by the competition between deformations of the elastic medium and fibre stretching and bending (see Figure 2.8). Furthermore, they found that the mechanical behaviour observed in these regimes can be very sensitive to the composition of the network.

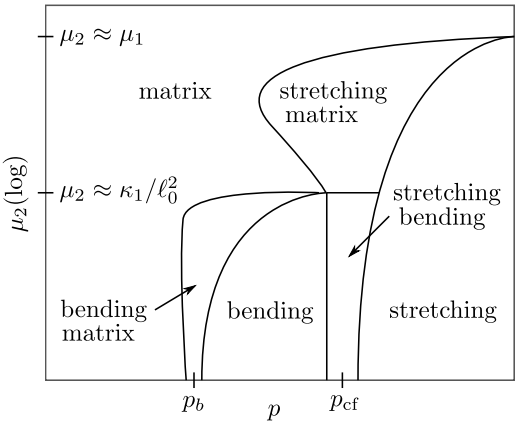


Figure 2.8: Adapted from [88], a schematic for the elastic regimes observed in terms of interaction between the elastic medium and embedded diluted network. Connectivity is given on the x axis where segments are diluted with probability $1 - p$, and p_b, p_{cf} represent the rigidity threshold and central force isostatic points respectively.

Broedersz et al.[13] developed an effective medium theory for stiff polymer networks that are cross-linked with highly flexible connections to an elastic continuum. In effective medium

theories (EMTs), the embedding medium behaviour is approximated by averaging over the properties of the constituent fibres. EMTs assume that macroscopically the system is homogeneous even though at the microscopic level the material is inhomogeneous and varies in structure. To describe network elasticity, Broedersz et al. split the network into two connected parts. The first was a single filament of length ℓ, with n attached flexible cross-links, known as the hairy rod. The second subsystem was the elastic medium modelled from a network that the hairy rod was connected to. The flexible cross-linkers that connected the hairy rod to the medium were modelled as short and highly flexible links, whereas the hairy rod was modelled as a stiff rod. In terms of network response to an applied force, the medium was assumed to deform affinely, and the local strain response of cross-linkers was modelled as dependent on cross-link position along the hairy rod.

Using their model they found a linear to nonlinear elastic regime crossover that they argued could be shown to be consistent with experiments. Furthermore, as a result of modelling cross-links as highly flexible (in contrast to simple rigid cross-links), they observed that the gradual onset of the nonlinear network behaviour originated from the nonlinear elasticity of the cross-links.

Summary

Here we have reviewed a variety of modelling considerations covered in the literature, giving additional attention where possible to concepts and topics core to the main body of the thesis. As we continue into the chapters that follow, we will draw from the content in this chapter, revisiting the key ideas. In particular, we will focus on the ideas that allow us to construct our linear computational approach for modelling mechanical properties of various static and athermal elastic fibre networks.

Chapter 3

Numerical Discretisation

In the previous chapter we provided an overview of the theory of fibres individually and collectively that is relevant to the derivation of the model that will be presented here. In the following pages we derive discretisations of the mechanical behaviour that we use to provide predictions. We discuss how the discretised mechanical behaviour of each individual fibre can be assembled into a large global system comprised of a sparse Hessian matrix and right hand side (RHS) vector of partial derivatives. To do this, we begin by introducing some preliminary notation and definitions, followed by a Taylor series expansion of the underlying system. From there, we move on to the discretisation of the stretching and bending behaviour of fibres at the level of isolated segments, before giving a description of how these definitions can be used locally to assemble the system globally.

3.1 Preliminaries

3.1.1 Notation

For the purposes of describing the numerical discretisation of our model, it is sufficient to begin with a high level description of a network and leave the details until chapters 5 and 6. For some yet to be formally defined network of uniform and rod-like fibres, confined within a 2D plane or a 3D box, we can discretise each fibre into a series of segments each joined by a finite number of points. We refer to each unique point i as a node, and denote it n_i. As each node represents a unique point, we therefore provide it with a unique ID which, unless stated explicitly, will be the same as the node number (i.e. n_i has ID i). A fibre can contain any number of nodes and any number of segments of variable length, but is denoted using the IDs of the nodes that represent the end points. Therefore a fibre with end points n_i and n_j is denoted f_{ij}. We note that f_{ij} and f_{ji} are described by opposite direction vectors and reversed node lists and therefore should not generally be considered equivalent.

3.1.2 Fundamental Definitions

We define the *network* to be the bounding box or plane in which the fibres lie.

Definition 10. Let n_i be a node. Then the *initial position vector* of this node in three dimensional space is written as

$$\mathbf{s}_i = [s_{i_x}, s_{i_y}, s_{i_z}]^T. \tag{3.1}$$

Definition 11. The unit *tangent vector* between n_i and some other node n_j is defined as

$$\hat{\mathbf{t}}_{ij} = \frac{\mathbf{s}_j - \mathbf{s}_i}{\ell_{ij}}, \tag{3.2}$$

where $\ell_{ij} = |\mathbf{s}_j - \mathbf{s}_i|$ is the *segment length*.

Definition 12. If some external force is applied to the network in which the fibres are contained, then the *displacement* of node n_i after this event is denoted

$$\mathbf{u}_i = [u_{i_x}, u_{i_y}, u_{i_z}]^T. \tag{3.3}$$

In addition to writing $\mathbf{u}_i$ for the displacement of an individual node, one can construct a global displacement vector $\mathbf{U}$ using two possible logical orderings. One such ordering is

$$\mathbf{U} = [U_1, U_2, U_3, U_4, \dots]^T = [u_{1_x}, u_{1_y}, u_{1_z}, \dots, u_{2_x}, u_{2_y}, u_{2_z}, \dots]^T, \tag{3.4}$$

which represents a vector of all the x, y, z components of all the *unknown* displacements of the system. Alternatively, we can order the vector as

$$\mathbf{U} = [u_{1_x}, u_{2_x}, \dots, u_{1_y}, u_{2_y}, \dots, u_{1_z}, u_{2_z}, \dots]^T. \tag{3.5}$$

The orderings presented here will be relevant to solution strategies in the sections and chapters hereafter. Unless explicitly indicated, it should be assumed that there is a $x, x, \dots, y, y, \dots, z, z, \dots$ ordering as in (3.5) rather than a $x, y, z, \dots, x, y, z, \dots$ ordering as in (3.4).

3.2 Model Derivation

For the total energy, E, of a system, expressed as a function of displacements $E(U_1, U_2, ..., U_N)$, where we write $E(\mathbf{U})$ for displacement vector $\mathbf{U}$, we begin with a Taylor series expansion of the energy of the displaced system about the unperturbed system $\mathbf{U}_0 = \mathbf{0}$;

$$E(\mathbf{U}) = E(\mathbf{U}_0) + \sum_{i=1}^{N} \frac{\partial E(\mathbf{U}_0)}{\partial U_i} U_i + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{\partial^2 E(\mathbf{U}_0)}{\partial U_i \partial U_j} U_i U_j + \dots \tag{3.6}$$

We wish to minimise the energy, such that $\frac{\partial E}{\partial U_k} = 0$ for all k, giving

$$\begin{aligned}
\frac{\partial E}{\partial U_k} &= \frac{\partial}{\partial U_k}\left(E_0 + \sum_{i=1}^{N} \frac{\partial E}{\partial U_i}\bigg|_{\mathbf{U}_0} U_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \frac{\partial^2 E}{\partial U_i \partial U_j}\bigg|_{\mathbf{U}_0} U_i U_j + \dots\right) \\
&= 0 + \sum_{i=1}^{N} \frac{\partial E}{\partial U_i}\bigg|_{\mathbf{U}_0} \frac{\partial}{\partial U_k}(U_i) + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \frac{\partial^2 E}{\partial U_i \partial U_j}\bigg|_{\mathbf{U}_0} \frac{\partial}{\partial U_k}(U_i U_j) + \dots \\
&= \sum_{i=1}^{N} \frac{\partial E}{\partial U_i}\bigg|_{\mathbf{U}_0} \frac{\partial U_i}{\partial U_k} + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \frac{\partial^2 E}{\partial U_i \partial U_j}\bigg|_{\mathbf{U}_0} \left(\frac{\partial U_i}{\partial U_k} U_j + U_i \frac{\partial U_j}{\partial U_k}\right) + \dots
\end{aligned}$$

where $|_{\mathbf{U}_0}$ denotes terms fixed as constants. Then substituting in the Kronecker delta

$$\delta_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j, \end{cases} \tag{3.7}$$

this is

$$\begin{aligned}
\frac{\partial E}{\partial U_k} &= \sum_{i=1}^{N} \frac{\partial E}{\partial U_i}\bigg|_{\mathbf{U}_0} \delta_{ik} + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \frac{\partial^2 E}{\partial U_i \partial U_j}\bigg|_{\mathbf{U}_0} (\delta_{ik} U_j + U_i \delta_{jk}) + \ \dots \\
&= \frac{\partial E}{\partial U_k}\bigg|_{\mathbf{U}_0} + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \left(\delta_{ik} U_j \frac{\partial^2 E}{\partial U_i \partial U_j}\bigg|_{\mathbf{U}_0} + \delta_{jk} U_i \frac{\partial^2 E}{\partial U_i \partial U_j}\bigg|_{\mathbf{U}_0}\right) + \ \dots \\
&= \frac{\partial E}{\partial U_k}\bigg|_{\mathbf{U}_0} + \frac{1}{2}\sum_{j=1}^{N} U_j \frac{\partial^2 E}{\partial U_k \partial U_j}\bigg|_{\mathbf{U}_0} + \frac{1}{2}\sum_{i=1}^{N} U_i \frac{\partial^2 E}{\partial U_i \partial U_k}\bigg|_{\mathbf{U}_0} + \ \dots \\
&= \frac{\partial E}{\partial U_k}\bigg|_{\mathbf{U}_0} + \sum_{j=1}^{N} U_j \frac{\partial^2 E}{\partial U_k \partial U_j}\bigg|_{\mathbf{U}_0} + \ \dots
\end{aligned}$$

For a *linear* model, higher order terms may be neglected, and thus for each U_k we can write a unique equation when $\frac{\partial E}{\partial U_k} = 0$ as

$$\sum_{j=1}^{N} U_j \frac{\partial^2 E}{\partial U_k \partial U_j}\bigg|_{\mathbf{U}_0} = -\frac{\partial E}{\partial U_k}\bigg|_{\mathbf{U}_0} = -B_k. \tag{3.8}$$

Combining these results in the following system of equations

$$\begin{aligned} U_1\frac{\partial^2 E}{\partial U_1\partial U_1}+U_2\frac{\partial^2 E}{\partial U_1\partial U_2}+\cdots+U_N\frac{\partial^2 E}{\partial U_1\partial U_N}&=-B_1,\\ U_1\frac{\partial^2 E}{\partial U_2\partial U_1}+U_2\frac{\partial^2 E}{\partial U_2\partial U_2}+\cdots+U_N\frac{\partial^2 E}{\partial U_2\partial U_N}&=-B_2,\\ &\vdots\\ U_1\frac{\partial^2 E}{\partial U_N\partial U_1}+U_2\frac{\partial^2 E}{\partial U_N\partial U_2}+\cdots+U_N\frac{\partial^2 E}{\partial U_N\partial U_N}&=-B_N, \end{aligned} \tag{3.9}$$

this can then be rewritten as the matrix system

$$\begin{pmatrix} \frac{\partial^2 E}{\partial U_1^2} & \frac{\partial^2 E}{\partial U_1\partial U_2} & \cdots & \frac{\partial^2 E}{\partial U_1\partial U_N}\\ \frac{\partial^2 E}{\partial U_2\partial U_1} & \frac{\partial^2 E}{\partial U_2^2} & \cdots & \frac{\partial^2 E}{\partial U_2\partial U_N}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{\partial^2 E}{\partial U_N\partial U_1} & \frac{\partial^2 E}{\partial U_N\partial U_2} & \cdots & \frac{\partial^2 E}{\partial U_N^2} \end{pmatrix}\begin{pmatrix} U_1\\ U_2\\ \vdots\\ U_N \end{pmatrix}=-\begin{pmatrix} B_1\\ B_2\\ \vdots\\ B_N \end{pmatrix}, \tag{3.10}$$

or in the more compact notation

$$H\mathbf{U}=-\mathbf{B}, \tag{3.11}$$

which defines the Hessian matrix H.

3.3 Discretising Fibre Behaviour

From this general problem we can specialise to fibre networks. For modelling the internal displacements of a network after applying a *linear* external force, thermal fluctuations of individual fibres can be disregarded and thus *thermal* contributions to the total energy are ignored. As such we consider the total energy of a given network to be the sum of the total *stretching* energy for the network and the total *bending* energy of the network

$$E=E^{\text{ stretch}}+E^{\text{ bend}}, \tag{3.12}$$

where $E^{\text{ stretch}}$ and $E^{\text{ bend}}$ are discretised and defined in the following subsections.

To provide discretised forms of individual fibres, as described in Chapter 2, it is necessary to decompose the continuous fibre length into finite segments. While some models partition each fibre into segments of equal lengths [94], we are striving for networks with randomised architectures, thus we allow fibres to be partitioned into irregular intervals. The partitioning of the fibres is determined by how fibres in the network are connected, which will be left for detailed discussion in chapters 5 and 6.

3.3.1 Discretised Bending Energy

Definition 13. Consider a segment ijk of a fibre comprised of an adjacent pair of segments ij and jk respectively. The discretised *bending energy* is given as

$$E_{ijk}^{\text{bend}} = \frac{\kappa_{ijk}\theta_{ijk}^2}{2\bar{\ell}_{ijk}} = \frac{\kappa_{ijk}\left[(\Delta\mathbf{s}_{ij} \times \Delta\mathbf{u}_{jk}) + (\Delta\mathbf{u}_{ij} \times \Delta\mathbf{s}_{jk})\right]^2}{\ell_{ij}^2\ell_{jk}^2(\ell_{ij} + \ell_{jk})}, \tag{3.13}$$

for a bending constant $\kappa_{ijk} = \frac{AE^f r^2}{4}$, with $A = \pi r^2$ for fibre segment radius r, and Young's modulus E^f, where

$$\Delta\mathbf{s}_{ij} = \mathbf{s}_j - \mathbf{s}_i, \quad \Delta\mathbf{s}_{jk} = \mathbf{s}_k - \mathbf{s}_j, \tag{3.14}$$

$$\Delta\mathbf{u}_{ij} = \mathbf{u}_j - \mathbf{u}_i, \quad \Delta\mathbf{u}_{jk} = \mathbf{u}_k - \mathbf{u}_j. \tag{3.15}$$

Derivation. Using what has been discussed in Chapter 2, we can express bending energy continuously as

$$E^{\text{bend}} = \frac{1}{2}\kappa \int ds \frac{1}{R^2}, \tag{3.16}$$

where R is the radius of the circle that fits to the local curvature, such that $R\theta = \ell$, where θ is the subtended angle, and ℓ is the arc length for the segment δs, as seen in Figure 3.1. Assuming

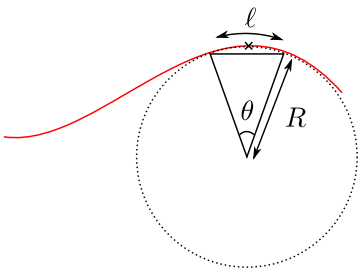


Figure 3.1: The local curvature of a fibre segment δs can be fit to a circle of radius R, such that the arc length of segment is ℓ and the bending angle is θ.

constant curvature, we integrate over the contour length, ℓ, of the fibre, and write (3.16) as

$$E^{\text{bend}} = \frac{1}{2}\kappa \int_{s=0}^{s=\ell} ds \left(\frac{\theta}{\ell}\right)^2 = \frac{1}{2}\kappa\left[\ell - 0\right]\left(\frac{\theta}{\ell}\right)^2. \tag{3.17}$$

In the discretised form, bending is only permitted between consecutive segments ij and jk of a common fibre. Thus for respective nodes n_i, n_j, n_k we must calculate the bending energy associated with central node n_j. Let $\mathbf{s}_i, \mathbf{s}_j, \mathbf{s}_k$ be the initial position vectors of n_i, n_j, n_k respectively. Then ℓ_{ij} and ℓ_{jk} are the lengths between n_i, n_j and n_j, n_k, and $\mathbf{u}_i, \mathbf{u}_j, \mathbf{u}_k$ are the displacements. Adopting the common convention [25], we consider curvature as constant be-

tween the two midpoints, $\bar{\ell}_{ijk} = \frac{1}{2}(\ell_{ij} + \ell_{jk})$. In this way we will be calculating the bending energy associated with the central node. We can visualise this discretised problem as in Figure 3.2, where θ_{ijk} is the angle between segments ij and jk through bending. Thus we write the

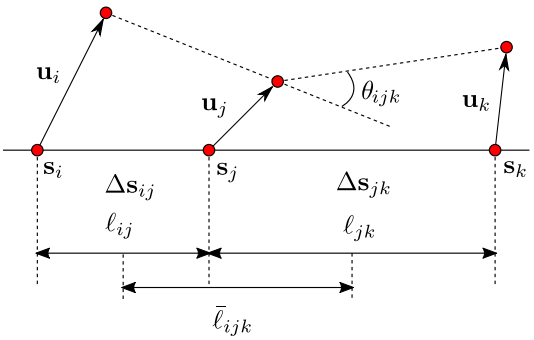


Figure 3.2: A fibre discretised into consecutive segments ij and jk. The dashed line at the top of the figure demonstrates bending over the segments for a given displacement.

discretised form of (3.17) as

$$E_{ijk}^{\text{bend}} = \frac{1}{2}\kappa_{ijk}\bar{\ell}_{ijk}\left[\frac{\theta_{ijk}}{\bar{\ell}_{ijk}}\right]^2, \tag{3.18}$$

$$= \frac{1}{4}\kappa_{ijk}(\ell_{ij} + \ell_{jk})\left[\frac{\theta_{ijk}}{\frac{1}{2}(\ell_{ij} + \ell_{jk})}\right]^2, \tag{3.19}$$

$$= \frac{\kappa_{ijk}\theta_{ijk}^2}{\ell_{ij} + \ell_{jk}}. \tag{3.20}$$

To calculate an estimate for θ_{ijk} we begin by considering the definition of the unit tangent vector of segment ij, after each node n_i, n_j has been displaced

$$\hat{\mathbf{t}}_{ij} = \frac{\mathbf{s}_j + \mathbf{u}_j - \mathbf{s}_i - \mathbf{u}_i}{|\mathbf{s}_j + \mathbf{u}_j - \mathbf{s}_i - \mathbf{u}_i|}. \tag{3.21}$$

Given a standard definition of the vector product

$$\sin\theta\hat{\mathbf{n}} = \hat{\mathbf{t}}_{ij} \times \hat{\mathbf{t}}_{jk}, \tag{3.22}$$

we can substitute (3.21) to give

$$\sin\theta\hat{\mathbf{n}} = \left(\frac{\mathbf{s}_j + \mathbf{u}_j - \mathbf{s}_i - \mathbf{u}_i}{|\mathbf{s}_j + \mathbf{u}_j - \mathbf{s}_i - \mathbf{u}_i|}\right) \times \left(\frac{\mathbf{s}_k + \mathbf{u}_k - \mathbf{s}_j - \mathbf{u}_j}{|\mathbf{s}_k + \mathbf{u}_k - \mathbf{s}_j - \mathbf{u}_j|}\right). \tag{3.23}$$

Using the notation defined in (3.14) and (3.15), equation (3.23) can be written more succinctly

as

$$\sin\theta\hat{\mathbf{n}} = \left(\frac{\Delta\mathbf{s}_{ij} + \Delta\mathbf{u}_{ij}}{|\Delta\mathbf{s}_{ij} + \Delta\mathbf{u}_{ij}|}\right) \times \left(\frac{\Delta\mathbf{s}_{jk} + \Delta\mathbf{u}_{jk}}{|\Delta\mathbf{s}_{jk} + \Delta\mathbf{u}_{jk}|}\right). \tag{3.24}$$

Rearranging (3.24)

$$(\Delta\mathbf{s}_{ij} + \Delta\mathbf{u}_{ij}) \times (\Delta\mathbf{s}_{jk} + \Delta\mathbf{u}_{jk}) = |\Delta\mathbf{s}_{ij} + \Delta\mathbf{u}_{ij}||\Delta\mathbf{s}_{jk} + \Delta\mathbf{u}_{jk}| \sin\theta\hat{\mathbf{n}}, \tag{3.25}$$

and then expanding the LHS

$$\Delta\mathbf{s}_{ij} \times \Delta\mathbf{s}_{jk} + \Delta\mathbf{s}_{ij} \times \Delta\mathbf{u}_{jk} + \Delta\mathbf{u}_{ij} \times \Delta\mathbf{s}_{jk} + \Delta\mathbf{u}_{ij} \times \Delta\mathbf{u}_{jk} = \ldots \tag{3.26}$$

$$\Delta\mathbf{s}_{ij} \times \Delta\mathbf{u}_{jk} + \Delta\mathbf{u}_{ij} \times \Delta\mathbf{s}_{jk} + \mathcal{O}(\Delta\mathbf{u}^2) = \ldots \tag{3.27}$$

we can see that it is linear to leading order, since $\Delta\mathbf{s}_{ij}$ and $\Delta\mathbf{s}_{jk}$ lie parallel, i.e. $(\Delta\mathbf{s}_{ij} \times \Delta\mathbf{s}_{jk}) = 0$. Now expanding the RHS

$$\ldots = |\Delta\mathbf{s}_{ij} + \Delta\mathbf{u}_{ij}||\Delta\mathbf{s}_{jk} + \Delta\mathbf{u}_{jk}| \sin\theta\hat{\mathbf{n}} \tag{3.28}$$

$$= |\Delta\mathbf{s}_{ij} \cdot \Delta\mathbf{s}_{jk} + \Delta\mathbf{s}_{ij} \cdot \Delta\mathbf{u}_{jk} + \Delta\mathbf{u}_{ij} \cdot \Delta\mathbf{s}_{jk} + \Delta\mathbf{u}_{ij} \cdot \Delta\mathbf{u}_{jk}| \sin\theta\hat{\mathbf{n}} \tag{3.29}$$

$$= |\Delta\mathbf{s}_{ij} \cdot \Delta\mathbf{s}_{jk} + \Delta\mathbf{s}_{ij} \cdot \Delta\mathbf{u}_{jk} + \Delta\mathbf{u}_{ij} \cdot \Delta\mathbf{s}_{jk} + \mathcal{O}(\Delta\mathbf{u}^2)| \sin\theta\hat{\mathbf{n}}. \tag{3.30}$$

For small displacements we expect a sufficiently small angle, such that $\sin\theta$ can be approximated as θ. Furthermore, since the LHS is linear to leading order and the θ term on the RHS is linear order, we may drop the RHS $\Delta\mathbf{u}$ terms. Thus the linearised expression is written

$$\Delta\mathbf{s}_{ij} \times \Delta\mathbf{u}_{jk} + \Delta\mathbf{u}_{ij} \times \Delta\mathbf{s}_{jk} = |\Delta\mathbf{s}_{ij}||\Delta\mathbf{s}_{jk}|\theta\hat{\mathbf{n_0}} \tag{3.31}$$

$$= \ell_{ij}\ell_{jk}\theta\hat{\mathbf{n_0}}. \tag{3.32}$$

Rearranging for θ and squaring gives

$$\theta^2 = \frac{|\Delta\mathbf{s}_{ij} \times \Delta\mathbf{u}_{jk} + \Delta\mathbf{u}_{ij} \times \Delta\mathbf{s}_{jk}|^2}{\ell_{ij}^2\ell_{jk}^2}, \tag{3.33}$$

which can be substituted back into (3.20) to give (3.13). □

3.3.2 Discretised Stretching Energy

Definition 14. Consider a segment ij comprising of an adjacent node pair n_i, n_j respectively, treated as a simple Hookean spring. The discretised *stretching energy* for this segment is

$$E_{ij}^{\text{stretch}} = \frac{k_{ij}}{2}[\Delta\mathbf{u}_{ij} \cdot \hat{\mathbf{t}}_{ij}]^2, \tag{3.34}$$

with a stretching constant $k_{ij} = \frac{\mu}{\ell_{ij}} = \frac{AE^f}{\ell_{ij}}$, for $A = \pi r^2$, radius r and Young's modulus E^f.

Derivation. We begin with the expression

$$E^{\text{stretch}} = \frac{\mu}{2} \int ds \left(\frac{d\epsilon(s)}{ds} \right)^2, \tag{3.35}$$

taken from (2.8), where $\epsilon(s)$ is the relative extension at arc coordinate s. Defining this relative extension as $\epsilon(s) = s\frac{\delta\ell}{\ell}$, it follows that

$$\frac{d\epsilon(s)}{ds} = \frac{\delta\ell}{\ell}, \tag{3.36}$$

thus integrating over the contour length of the segment, this is

$$E^{\text{stretch}} = \frac{\mu}{2} \int_{s=0}^{s=\ell} ds \left(\frac{\delta\ell}{\ell} \right)^2 = \frac{\mu}{2} \left[\ell - 0\right] \frac{\delta\ell^2}{\ell^2} = \frac{1}{2}\frac{\mu}{\ell}\delta\ell^2. \tag{3.37}$$

Discretising this as a single rod-like segment with end coordinates described by position vectors $\mathbf{s}_i, \mathbf{s}_j$, (3.37) is

$$E_{ij}^{\text{stretch}} = \frac{1}{2}\frac{\mu}{\ell_{ij}}[\delta\ell_{ij}]^2 = \frac{1}{2}\frac{\mu}{\ell_{ij}}[\Delta\mathbf{u}_{ij} \cdot \hat{\mathbf{t}}_{ij}]^2, \tag{3.38}$$

where $\delta\ell_{ij}$ has been rewritten using the linear estimation of extension and compression given in Appendix B.2, which gives (3.34) for $k_{ij} = \frac{\mu}{\ell_{ij}}$. □

3.4 Global Assembly from Local Behaviour

For a given network, the procedure for assembling a system such as (3.11) comprises of two stages:

1. Define the *local* behaviour of isolated components of the network.
2. Assemble the contributions of this defined behaviour into the appropriate part of the *global* system.

I will now elaborate on each of these in turn, beginning by considering for simplicity a *single* segment in isolation.

3.4.1 Defining Local Behaviour

Isolated Stretching Behaviour

We begin by taking a single segment from the global system and considering it in isolation. Our goal here is to accurately describe the stretching and compressive energy contribution of an isolated segment to the total stretching/compression energy of the global system.

As an individual segment, we can define the stretching/compression energy as in (3.34). It follows that in a similar manner we may also define the 1st and 2nd order partial derivatives of the energy with respect to each node, n_i, n_j, of the segment and the three degrees of freedom (x, y, z). For example

$$\frac{\partial E_{ij}}{\partial u_{i_x}} = -\frac{k_{ij}s_{ij_x}\left(s_{ij_x}u_{ij_x} + s_{ij_y}u_{ij_y} + s_{ij_z}u_{ij_z}\right)}{s_{ij_x}^2 + s_{ij_y}^2 + s_{ij_z}^2}, \tag{3.39}$$

$$\begin{aligned}
\frac{\partial^2 E_{ij}}{\partial u_{i_x}^2} &= \frac{k_{ij}s_{ij_x}^2}{s_{ij_x}^2 + s_{ij_y}^2 + s_{ij_z}^2}, \\
\frac{\partial^2 E_{ij}}{\partial u_{i_x}\partial u_{i_y}} &= \frac{k_{ij}s_{ij_x}s_{ij_y}}{s_{ij_x}^2 + s_{ij_y}^2 + s_{ij_z}^2}, \\
\frac{\partial^2 E_{ij}}{\partial u_{i_x}\partial u_{i_z}} &= \frac{k_{ij}s_{ij_x}s_{ij_z}}{s_{ij_x}^2 + s_{ij_y}^2 + s_{ij_z}^2}.
\end{aligned} \tag{3.40}$$

Proceeding in this way, we can calculate the remaining 1st and 2nd order partial derivatives. For convenience we choose to store the values of the 1st and 2nd order partial derivatives in a local system, which we can write as

$$(H\mathbf{U})_{ij}^{\text{stretch}} = -\mathbf{B}_{ij}^{\text{stretch}}, \tag{3.41}$$

where the 6×6 H_{ij}^{stretch} and $\mathbf{B}_{ij}^{\text{stretch}}$ are defined as

$$H_{ij}^{\text{stretch}} = \frac{k_{ij}}{s_{ij_x}^2 + s_{ij_y}^2 + s_{ij_z}^2}\begin{bmatrix}
s_{ij_x}^2 & -s_{ij_x}^2 & s_{ij_x}s_{ij_y} & -s_{ij_x}s_{ij_y} & s_{ij_x}s_{ij_z} & -s_{ij_x}s_{ij_z} \\
-s_{ij_x}^2 & s_{ij_x}^2 & -s_{ij_x}s_{ij_y} & s_{ij_x}s_{ij_y} & -s_{ij_x}s_{ij_z} & s_{ij_x}s_{ij_z} \\
s_{ij_x}s_{ij_y} & -s_{ij_x}s_{ij_y} & s_{ij_y}^2 & -s_{ij_y}^2 & s_{ij_y}s_{ij_z} & -s_{ij_y}s_{ij_z} \\
-s_{ij_x}s_{ij_y} & s_{ij_x}s_{ij_y} & -s_{ij_y}^2 & s_{ij_y}^2 & -s_{ij_y}s_{ij_z} & s_{ij_y}s_{ij_z} \\
s_{ij_x}s_{ij_z} & -s_{ij_x}s_{ij_z} & s_{ij_y}s_{ij_z} & -s_{ij_y}s_{ij_z} & s_{ij_z}^2 & -s_{ij_z}^2 \\
-s_{ij_x}s_{ij_z} & s_{ij_x}s_{ij_z} & -s_{ij_y}s_{ij_z} & s_{ij_y}s_{ij_z} & -s_{ij_z}^2 & s_{ij_z}^2
\end{bmatrix}, \tag{3.42}$$

$$\mathbf{B}_{ij}^{\text{stretch}} = \frac{k_{ij}\left(s_{ij_x}u_{ij_x} + s_{ij_y}u_{ij_y} + s_{ij_z}u_{ij_z}\right)}{s_{ij_x}^2 + s_{ij_y}^2 + s_{ij_z}^2}\begin{bmatrix}
-s_{ij_x} \\ s_{ij_x} \\ -s_{ij_y} \\ s_{ij_y} \\ -s_{ij_z} \\ s_{ij_z}
\end{bmatrix}, \tag{3.43}$$

These local systems do not correspond to any physical network. They are used as an aid for assembling the correct contributions into the global system that will be solved. Indeed, there are often values in these local systems that provide no contribution and are therefore unused.

Further details of this will be discussed in Section 3.4.2.

Isolated Bending Behaviour

Next we move on to consider a pair of consecutive segments (equivalently a triplet of adjacent nodes on one fibre) in isolation from the global system. In this section the goal is to describe the bending energy contribution of an isolated segment pair to the total bending energy of the global system.

As above, we can use the definition of the bending energy (3.13) to derive the 1st and 2nd order partial derivatives of the energy with respect to each node, n_i, n_j, n_k, of the segment and the three degrees of freedom (x, y, z). For example

$$\frac{\partial E_{ijk}}{\partial u_{i_x}} = \frac{2\kappa_{ijk}}{\ell_{ij}^2 \ell_{jk}^2 (\ell_{ij} + \ell_{jk})} \Big[s_{jk_y} \left(-s_{ij_x} u_{jk_y} + s_{ij_y} u_{jk_x} + s_{jk_x} u_{ij_y} - s_{jk_y} u_{ij_x} \right) + s_{jk_z} \left(-s_{ij_x} u_{jk_z} + s_{ij_z} u_{jk_x} + s_{jk_x} u_{ij_z} - s_{jk_z} u_{ij_x} \right) \Big], \quad (3.44)$$

$$\begin{aligned} \frac{\partial^2 E_{ijk}}{\partial u_{i_x}^2} &= \frac{2\kappa_{ijk}}{\ell_{ij}^2 \ell_{jk}^2 (\ell_{ij} + \ell_{jk})} \left(s_{jk_y}^2 + s_{jk_z}^2 \right), \\ \frac{\partial^2 E_{ijk}}{\partial u_{i_x} \partial u_{j_x}} &= -\frac{2\kappa_{ijk}}{\ell_{ij}^2 \ell_{jk}^2 (\ell_{ij} + \ell_{jk})} \left(s_{ik_y} s_{jk_y} + s_{ik_z} s_{jk_z} \right), \\ \frac{\partial^2 E_{ijk}}{\partial u_{i_x} \partial u_{k_x}} &= \frac{2\kappa_{ijk}}{\ell_{ij}^2 \ell_{jk}^2 (\ell_{ij} + \ell_{jk})} \left(s_{ij_y} s_{jk_y} + s_{ij_z} s_{jk_z} \right). \end{aligned} \quad (3.45)$$

Proceeding in this way, we can calculate the remaining 1st and 2nd order partial derivatives. Once again we choose to store the values of the 1st and 2nd order partial derivatives in a local system, which we can write as

$$(H\mathbf{U})_{ijk}^{\text{bend}} = -\mathbf{B}_{ijk}^{\text{bend}}, \quad (3.46)$$

where the 9×9 H_{ijk}^{bend} is defined as

$$H_{ijk}^{\text{bend}} = \frac{2\kappa_{ijk}}{\ell_{ij}^2 \ell_{jk}^2 (\ell_{ij} + \ell_{jk})} \begin{bmatrix} H_{ijk_{xx}}^{\text{bend}} & H_{ijk_{xy}}^{\text{bend}} & H_{ijk_{xz}}^{\text{bend}} \\ H_{ijk_{yx}}^{\text{bend}} & H_{ijk_{yy}}^{\text{bend}} & H_{ijk_{yz}}^{\text{bend}} \\ H_{ijk_{zx}}^{\text{bend}} & H_{ijk_{zy}}^{\text{bend}} & H_{ijk_{zz}}^{\text{bend}} \end{bmatrix}, \quad (3.47)$$

where

$$H_{ijk_{xx}}^{\text{bend}} = \begin{bmatrix} s_{jk_y}^2 + s_{jk_z}^2 & -s_{ik_y} s_{jk_y} - s_{ik_z} s_{jk_z} & s_{ij_y} s_{jk_y} + s_{ij_z} s_{jk_z} \\ -s_{ik_y} s_{jk_y} - s_{ik_z} s_{jk_z} & s_{ik_y}^2 + s_{ik_z}^2 & -s_{ik_y} s_{ij_y} - s_{ik_z} s_{ij_z} \\ s_{ij_y} s_{jk_y} + s_{ij_z} s_{jk_z} & -s_{ik_y} s_{ij_y} - s_{ik_z} s_{ij_z} & s_{ij_y}^2 + s_{ij_z}^2 \end{bmatrix}, \quad (3.48)$$

and $H^{\text{bend}}_{ijk_{xy}}, \ldots, H^{\text{bend}}_{ijk_{zz}}$ are given in Appendix A.1. We define $\mathbf{B}^{\text{bend}}_{ijk}$ as

$$\mathbf{B}^{\text{bend}}_{ijk} = \frac{2\kappa_{ijk}}{\ell_{ij}^2 \ell_{jk}^2 \left(\ell_{ij} + \ell_{jk}\right)} \begin{bmatrix} s_{jk_y}\beta_1^{i_x} + s_{jk_z}\beta_2^{i_x} \\ -s_{ik_y}\beta_1^{j_x} - s_{ik_z}\beta_2^{j_x} \\ s_{ij_y}\beta_1^{k_x} + s_{ij_z}\beta_2^{k_x} \\ -s_{jk_x}\beta_1^{i_y} + s_{jk_z}\beta_2^{i_y} \\ s_{ik_x}\beta_1^{j_y} - s_{ik_z}\beta_2^{j_y} \\ -s_{ij_x}\beta_1^{k_y} + s_{ij_z}\beta_2^{k_y} \\ -s_{jk_x}\beta_1^{i_z} - s_{jk_y}\beta_2^{i_z} \\ s_{ik_x}\beta_1^{j_z} + s_{ik_y}\beta_2^{j_z} \\ -s_{ij_x}\beta_1^{k_z} - s_{ij_y}\beta_2^{k_z} \end{bmatrix}, \tag{3.49}$$

where the shorthand forms $\beta_1^{i_x}, \ldots, \beta_2^{k_z}$ are defined in Appendix A.2.

3.4.2 Global Assembly

Having defined the local behaviour of network components considered in isolation, we can now demonstrate how each entry of the global system is the sum of many individual contributions. We begin this discussion by considering a network of fibres with internal nodes $n_a, n_b, n_c, n_d, \ldots$. We note that *internal* is yet to be formally defined, but here we are referring to a node lying within our system that has an unknown displacement to be calculated. For each of these nodes, we can consider their respective relationships with each other node in the system. Node pairs with a common segment each contribute to the stretching of the common segment, and node triplets with a common pair of adjacent segments each contribute to the bending between the adjacent segments. By accounting for the interactions of nearby nodes in this way, we are indirectly accounting for how the nodes interact across the entire system.

By taking the $x, x, \ldots, y, y, \ldots, z, z, \ldots$ ordering as in (3.5), we can represent the global Hessian matrix (3.11) as a set of sub-blocks

$$H = \begin{bmatrix} H_{xx} & H_{xy} & H_{xz} \\ H_{yx} & H_{yy} & H_{yz} \\ H_{zx} & H_{zy} & H_{zz} \end{bmatrix}, \tag{3.50}$$

where each sub-block can be written for a hypothetical system with internal nodes $n_a, n_b, n_c, n_d, \ldots$ as

$$H_{pq} = \begin{bmatrix} E_{u_{a_p}u_{a_q}} & E_{u_{a_p}u_{b_q}} & E_{u_{a_p}u_{c_q}} & E_{u_{a_p}u_{d_q}} & \cdots \\ E_{u_{b_p}u_{a_q}} & E_{u_{b_p}u_{b_q}} & E_{u_{b_p}u_{c_q}} & E_{u_{b_p}u_{d_q}} & \cdots \\ E_{u_{c_p}u_{a_q}} & E_{u_{c_p}u_{b_q}} & E_{u_{c_p}u_{c_q}} & E_{u_{c_p}u_{d_q}} & \cdots \\ E_{u_{d_p}u_{a_q}} & E_{u_{d_p}u_{b_q}} & E_{u_{d_p}u_{c_q}} & E_{u_{d_p}u_{d_q}} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \tag{3.51}$$

such that each element represents a second order partial derivative of the total network energy

$$E_{u_{a_p} u_{b_q}} = \frac{\partial^2 E}{\partial u_{a_p} \partial u_{b_q}}. \tag{3.52}$$

More specifically, each element can be assembled from the sum of the stretching and bending contributions relating to the interaction of the two nodes of the respective element

$$\frac{\partial^2 E}{\partial u_{a_p} \partial u_{b_q}} = \begin{cases} \displaystyle\sum_i \frac{\partial^2 (E^{ai}_{\text{stretch}} + E^{ia}_{\text{stretch}})}{\partial u_{a_p} \partial u_{a_q}} + \sum_{i,j} \frac{\partial^2 (E^{aij}_{\text{bend}} + E^{iaj}_{\text{bend}} + E^{ija}_{\text{bend}})}{\partial u_{a_p} \partial u_{a_q}}, & \text{if } a = b \\ \displaystyle\frac{\partial^2 E^{ab}_{\text{stretch}}}{\partial u_{a_p} \partial u_{b_q}} + \sum_i \frac{\partial^2 (E^{abi}_{\text{bend}} + E^{aib}_{\text{bend}} + E^{iab}_{\text{bend}})}{\partial u_{a_p} \partial u_{b_q}}, & \text{if } a \neq b \end{cases} \tag{3.53}$$

Thus for any ordered pairing of internal nodes, there are 9 partial derivatives within the global matrix (one for each sub-block) that need to be calculated. In the case of nodes n_a and n_b, these are

$$\begin{gathered} \frac{\partial^2 E}{\partial u_{a_x} \partial u_{b_x}}, \frac{\partial^2 E}{\partial u_{a_x} \partial u_{b_y}}, \frac{\partial^2 E}{\partial u_{a_x} \partial u_{b_z}}, \\ \frac{\partial^2 E}{\partial u_{a_y} \partial u_{b_x}}, \frac{\partial^2 E}{\partial u_{a_y} \partial u_{b_y}}, \frac{\partial^2 E}{\partial u_{a_y} \partial u_{b_z}}, \\ \frac{\partial^2 E}{\partial u_{a_z} \partial u_{b_x}}, \frac{\partial^2 E}{\partial u_{a_z} \partial u_{b_y}}, \frac{\partial^2 E}{\partial u_{a_z} \partial u_{b_z}}, \end{gathered} \tag{3.54}$$

where each 2nd order partial derivative is a sum of relevant local stretching and bending energy contributions. Therefore, to obtain the values of (3.54) we assemble the relevant local systems (3.42), (3.43), (3.47), and (3.49). From each of these systems, we select the entries that contribute to a given partial derivative of (3.54) and add the value stored there to the global system. To visualise which entries will contribute to a given derivative in (3.54) we note that the $x, x, \ldots, y, y, \ldots, z, z, \ldots$ ordering also exposes a block structure in the local stretching matrix (3.42)

$$H^{\text{stretch}}_{ij} = \begin{bmatrix} H^{\text{stretch}}_{ij_{xx}} & H^{\text{stretch}}_{ij_{xy}} & H^{\text{stretch}}_{ij_{xz}} \\ H^{\text{stretch}}_{ij_{yx}} & H^{\text{stretch}}_{ij_{yy}} & H^{\text{stretch}}_{ij_{yz}} \\ H^{\text{stretch}}_{ij_{zx}} & H^{\text{stretch}}_{ij_{zy}} & H^{\text{stretch}}_{ij_{zz}} \end{bmatrix}, \tag{3.55}$$

such that each sub-block can be written in the form

$$H^{\text{stretch}}_{ij_{pq}} = \begin{bmatrix} E^{\text{stretch}}_{u_{i_p} u_{i_q}} & E^{\text{stretch}}_{u_{i_p} u_{j_q}} \\ E^{\text{stretch}}_{u_{j_p} u_{i_q}} & E^{\text{stretch}}_{u_{j_p} u_{j_q}} \end{bmatrix}, \tag{3.56}$$

along with the local stretching RHS vector (3.43) as

$$\mathbf{B}_{ij}^{\text{stretch}} = \begin{bmatrix} E_{u_{i_x}}^{\text{stretch}} \\ E_{u_{j_x}}^{\text{stretch}} \\ E_{u_{i_y}}^{\text{stretch}} \\ E_{u_{j_y}}^{\text{stretch}} \\ E_{u_{i_z}}^{\text{stretch}} \\ E_{u_{j_z}}^{\text{stretch}} \end{bmatrix}. \tag{3.57}$$

Using this notation as a guide, contributing sub-block entries of a given local stretching system can be identified and added to the relevant location of the global system. It follows that a similar procedure can be carried out with the local bending system (3.47) and (3.49).

To elaborate, we study the case of the pair of nodes n_a, n_b and begin with the stretching contributions. If the nodes are not adjacent along a common fibre, there will be no local contribution, so we do not assemble (3.55). Thus the 9 values in the global matrix (corresponding to derivatives given by (3.54)) will remain unchanged. Notably for a stretching only system, the global matrix entries corresponding with mixed contributions of any non-adjacent nodes will remain zero. Assuming that n_a and n_b share a common fibre and are adjacent, then we may assemble the local stretching matrix (3.55), which we refer to as H_{ab}^{stretch}. We have already stated that both n_a and n_b are internal, and so from this local system, the value of $E_{u_{a_x}u_{b_x}}^{\text{stretch}}$ will be added to $E_{u_{a_x}u_{b_x}}$ of the global system, $E_{u_{a_x}u_{b_y}}^{\text{stretch}}$ to $E_{u_{a_x}u_{b_y}}$ etc. from each sub-block until all 9 contributions have been added. We do the same for all $E_{u_{a_p}u_{a_q}}$, $E_{u_{b_p}u_{a_q}}$, $E_{u_{b_p}u_{b_q}}$, noting that the derivatives $E_{u_{b_p}u_{a_q}}$ have locations in the global system that are distinct from $E_{u_{a_q}u_{b_p}}$. In contrast, the fact that n_a and n_b are internal means that we add *no* contributions to the global RHS vector for this node pair.

Now let us consider adjacent nodes n_a and n_w, where n_w is *not* internal but lies instead on the boundary of our domain. In this case, there are still contributions to be added to the global system, and so we must still assemble the local system, H_{aw}^{stretch}. Since n_a is internal, we add the $E_{u_{a_p}u_{a_q}}^{\text{stretch}}$ contributions to the global matrix. These contributions are added to the *same* locations in the global matrix as the $E_{u_{a_p}u_{a_q}}^{\text{stretch}}$ contributions added from H_{ab}^{stretch}. However, since n_w lies on the boundary, we do not add any of the $E_{u_{a_p}u_{w_q}}^{\text{stretch}}$, $E_{u_{w_p}u_{a_q}}^{\text{stretch}}$ or $E_{u_{w_p}u_{w_q}}^{\text{stretch}}$ contributions to the global matrix. The fact that n_a is adjacent to boundary n_w, means that indirectly n_a is contributing to the global RHS vector, and so the 3 values $E_{u_{a_x}}^{\text{stretch}}$, $E_{u_{a_y}}^{\text{stretch}}$ and $E_{u_{a_z}}^{\text{stretch}}$ of the local RHS vector are added to their counterpart locations in the global RHS vector.

Moving on to consider the bending contributions, the process is largely the same. For the hypothetical consecutive node triplet n_c, n_a, n_b on the same fibre, we assemble the relevant local bending system, H_{cab}^{bend}, and add the relevant contributions at their respective locations for each of

$$E_{u_{c_p}u_{c_q}}^{\text{bend}}, E_{u_{c_p}u_{a_q}}^{\text{bend}}, E_{u_{c_p}u_{b_q}}^{\text{bend}}, E_{u_{a_p}u_{c_q}}^{\text{bend}}, E_{u_{a_p}u_{a_q}}^{\text{bend}}, E_{u_{a_p}u_{b_q}}^{\text{bend}}, E_{u_{b_p}u_{c_q}}^{\text{bend}}, E_{u_{b_p}u_{a_q}}^{\text{bend}}, E_{u_{b_p}u_{b_q}}^{\text{bend}}, \tag{3.58}$$

Since all three nodes are internal, they do not contribute to the global RHS vector. In the case of a triplet n_a, n_b, n_w where one node, n_w, lies on the boundary, the contributions relating to each of $E^{\text{bend}}_{u_{a_p}u_{a_q}}$, $E^{\text{bend}}_{u_{a_p}u_{b_q}}$, $E^{\text{bend}}_{u_{b_p}u_{a_q}}$, and $E^{\text{bend}}_{u_{b_p}u_{b_q}}$ would be added to the global matrix, but all contributions involving the boundary node n_w would not. Since n_w is a boundary node, n_a and n_b indirectly contribute to the global RHS vector, and therefore the relevant entries of the local RHS vector should be added to the global RHS vector. In the unusual case that a node triplet n_v, n_a, n_w, spans the entire network such that both n_v and n_w lie on the boundary, only the $E^{\text{bend}}_{u_{a_p}u_{a_q}}$ contributions would be added to the global matrix and the $E^{\text{bend}}_{u_{a_p}}$ contributions to the global RHS.

Summary

Using the notation and fundamental definitions introduced at the start of this chapter, we have a foundation on which we can build a more detailed description of what constitutes a network, random or otherwise, and how such a network is structured from individual elements.

By following the description of the global system that we have provided, in the reverse direction, we can obtain an understanding of how the macroscopic mechanical behaviour of real world fibres networks can be decomposed into smaller contributing components. To characterise the mechanics of these individual components in isolation, we have given discretisations that we can use to approximate the continuous mechanical behaviour of individual real world fibres.

From the derivation of the model and the global system that we obtain from it, we can now elaborate further on the matrix properties and the methodology surrounding the numerical techniques used to reach a meaningful solution of the global system that can then be translated into mechanical properties.

Chapter 4

Numerical Solution Techniques

In the previous chapter we outlined the discretised model that is used to provide predictions of the mechanical properties of a given random fibre network. In this chapter we will give more details of how the discretised model can be used to calculate these predictions. Specifically, we will discuss the different approaches that can be taken to solve the large global equation system presented in the previous chapter. Beginning with a short discussion of how the Hessian matrix can be represented and stored within our model, we will then outline some of the simple properties that can be inferred from this, and use it to motivate approaches for solving the system. These approaches can be broadly divided into methods that solve the system as a time-independent static problem, and methods that move towards the solution in a pseudo-time-stepping approach. We will focus first on matrix-based methods used to solve the system, beginning with a description of the direct methods used, which are largely variations of the direct LU decomposition method, before moving on to discuss some of the iterative methods considered. We will conclude the chapter with a brief look at pseudo time stepping and elastic medium embedding.

4.1 Matrix Representation

Having discussed the fine-grained details of how the global system can be assembled from the local contributions of individual fibres, we will now discuss the matrix at a higher level, with a particular focus on the properties that can be inferred from its representation and structure.

4.1.1 Matrix Properties

As we have discussed in Section 3.4, grouping the unknown displacements in $\mathbf{U}$ by coordinate axis reveals the block structure of H which can be written as

$$H = \begin{bmatrix} H_{xx} & H_{xy} & H_{xz} \\ H_{yx} & H_{yy} & H_{yz} \\ H_{zx} & H_{zy} & H_{zz} \end{bmatrix}, \quad \text{where } H_{pq} = \begin{bmatrix} \dfrac{\partial^2 E}{\partial u_{i_p}\partial u_{i_q}} & \dfrac{\partial^2 E}{\partial u_{i_p}\partial u_{j_q}} & \cdots & \dfrac{\partial^2 E}{\partial u_{i_p}\partial u_{k_q}} \\ \dfrac{\partial^2 E}{\partial u_{j_p}\partial u_{i_q}} & \dfrac{\partial^2 E}{\partial u_{j_p}\partial u_{j_q}} & \cdots & \dfrac{\partial^2 E}{\partial u_{j_p}\partial u_{k_q}} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 E}{\partial u_{k_p}\partial u_{i_q}} & \dfrac{\partial^2 E}{\partial u_{k_p}\partial u_{j_q}} & \cdots & \dfrac{\partial^2 E}{\partial u_{k_p}\partial u_{k_q}} \end{bmatrix} \tag{4.1}$$

for $i, j = 1, ..., N$ and $p, q \in \{x, y, z\}$. It follows from above that the sub-matrix H_{pq} is not necessarily symmetric, but the global symmetry of H can be exposed by taking the transpose of H_{pq}

$$H_{pq}^T = \begin{bmatrix} \dfrac{\partial^2 E}{\partial u_{i_p}\partial u_{i_q}} & \dfrac{\partial^2 E}{\partial u_{j_p}\partial u_{i_q}} & \cdots & \dfrac{\partial^2 E}{\partial u_{k_p}\partial u_{i_q}} \\ \dfrac{\partial^2 E}{\partial u_{i_p}\partial u_{j_q}} & \dfrac{\partial^2 E}{\partial u_{j_p}\partial u_{j_q}} & \cdots & \dfrac{\partial^2 E}{\partial u_{k_p}\partial u_{j_q}} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 E}{\partial u_{i_p}\partial u_{k_q}} & \dfrac{\partial^2 E}{\partial u_{j_p}\partial u_{k_q}} & \cdots & \dfrac{\partial^2 E}{\partial u_{k_p}\partial u_{k_q}} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial^2 E}{\partial u_{i_q}\partial u_{i_p}} & \dfrac{\partial^2 E}{\partial u_{i_q}\partial u_{j_p}} & \cdots & \dfrac{\partial^2 E}{\partial u_{i_q}\partial u_{k_p}} \\ \dfrac{\partial^2 E}{\partial u_{j_q}\partial u_{i_p}} & \dfrac{\partial^2 E}{\partial u_{j_q}\partial u_{j_p}} & \cdots & \dfrac{\partial^2 E}{\partial u_{j_q}\partial u_{k_p}} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 E}{\partial u_{k_q}\partial u_{i_p}} & \dfrac{\partial^2 E}{\partial u_{k_q}\partial u_{j_p}} & \cdots & \dfrac{\partial^2 E}{\partial u_{k_q}\partial u_{k_p}} \end{bmatrix} = H_{qp}, \tag{4.2}$$

provided that the order of differentiation does not matter, which will be true for any sufficiently smooth E. From (4.2) we can rewrite H as

$$H = \begin{bmatrix} H_{xx} & H_{xy} & H_{xz} \\ H_{xy}^T & H_{yy} & H_{yz} \\ H_{xz}^T & H_{yz}^T & H_{zz} \end{bmatrix}, \tag{4.3}$$

thus we deduce H is symmetric. Furthermore for the diagonal blocks, $p = q$, so it follows from (4.1) that H_{xx}, H_{yy} and H_{zz} are also symmetric.

Calculating the global mechanical behaviour as a sum of individual local contributions results in a sparse matrix. This is because the sparsity of H is largely determined from the local connectivity of each fibre with the nearby fibres of the network. It follows that an estimate of the sparsity pattern can be obtained from an adjacency matrix of the internal nodes of a network, where we take an adjacency matrix to be a matrix, A, representing the connectivity

between each pair of internal nodes, n_i, n_j, such that

$$A_{ij} = \begin{cases} 2, & \text{if } i = j, \\ 1, & \text{if there is a segment between nodes } n_i, n_j, \\ 0, & \text{otherwise.} \end{cases} \tag{4.4}$$

For a system with N_{int} internal nodes the adjacency matrix will be $N_{int} \times N_{int}$, which is the same size as a single sub-block of the $3N_{int} \times 3N_{int}$ global Hessian. Thus a first estimate for the sparsity of H is the sparsity pattern of the adjacency matrix, A, repeated for each sub-block H_{pq} of the global matrix H (as seen in Section 3.4.2). For stretching only systems with bending behaviour excluded, we can therefore deduce $nz(H) \leq 9nz(A)$, for total nonzero entries, nz, of the Hessian matrix H and adjacency matrix A respectively, where A has the same dimensions as the sub-matrix H_{pq}. However when bending behaviour is accounted for, any node pair n_a, n_c, such that n_a is only indirectly connected to n_c along the same fibre through common node n_b, will have a zero value in the adjacency matrix, but will likely have a nonzero value in the 9 relevant entries of H. The result of this is an underestimate of the number of nonzero values which is less preferable due to the cost of additional copying of matrix entries and reallocation of memory at a later stage, rather than preallocating excess memory at an earlier stage. The relevance of a good estimate for the matrix preallocation should be evident in Section 4.1.2 that follows.

Tests using the `chol` function in MATLAB reveal that the global Hessian matrix is not necessarily symmetric *positive definite* for random fibre networks, despite being so for some lattice based networks. The `chol` function is typically used for performing a Cholesky factorisation on a given matrix, however it can also be used for testing if a matrix is symmetric positive definite by requesting the return of an integer flag.

4.1.2 Matrix Storage

There are a variety of possible storage schemes for efficient representation of large sparse matrices, including coordinate (COO), compressed sparse row (CSR), compressed sparse column (CSC), and modified sparse row (MSR) formats [76, p. 84]. The COO format is the simplest of these schemes, exploiting the fact that any element A_{ij} of a sparse matrix A can be identified by row and column indexes i, j respectively. Thus the COO format stores the nonzero elements of a sparse matrix A in three arrays, corresponding to an array for row indexes, i, column indexes, j, and matrix values, A_{ij}. Since every element can be identified using a pair of coordinates, ordering is not required for COO storage.

To derive the CSR format, we can begin with a matrix stored in COO format and order the arrays by the row indexes, as seen in Figure 4.1 (a) and (b). From this *row major* ordering, we note that the repeated indices in the row array can be compressed to form a new array of pointers that identify where in the column and value arrays a new row begins [see Figure 4.1

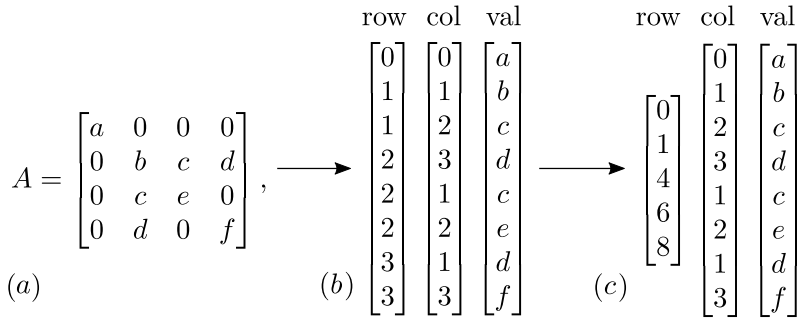


Figure 4.1: A sparse matrix (a) stored in COO (b) and CSR (c) formats respectively.

(c)]. The CSC format can derived in a similar manner, beginning with a *column major* ordering of the coordinates. The MSR format builds on the CSR format, compressing the row indexes into an array of pointers, but rather than taking a standard row major ordering, it places the diagonal entries of the matrix before the other nonzero entries (see [76, p. 85]).

In the case of the Hessian matrix presented above, and hereafter, the sparsity of H is exploited by storing the nonzero values in the CSR format. The CSR scheme was implemented early on in the project, during some of the first iterations of the model development. At the later integration stage with PETSc, this scheme was maintained since it is the default storage scheme for PETSc matrices, however the need to provide an exact number of nonzero values per row becomes optional. For optimum performance it is recommended that the user provides the PETSc `MatSetSizes` function with an estimate of the number of nonzeros per row, `nz`. In the case of random fibre networks where the number of nonzero values will fluctuate per row, the user should provide an array, `nnz`, of length `m` for `m` rows, where `nnz[i]` has a value representing the number of nonzeros in row `i` of the sparse matrix.

For early random fibre network experimentation, the bottleneck was not system assembly but solving the system (discussed in Section 4.2). In such a situation, the user can request that PETSc estimates the number of nonzero entries instead. It should be noted that when the user provides either a value `nz` or array of values `nnz`, it does not have to be the *correct* value(s) of nonzeros, but rather a sensible estimate. When the nonzeros are underestimated, PETSc will perform additional allocations as needed. Referring back to Section 4.1.1, it can be seen why the adjacency matrix may provide a reasonable estimate for the number of nonzero entries required for each block of the Hessian in a stretching only system, but why it could be an underestimate when bending behaviour is also included.

Ideally for optimal assembly time, the user should provide PETSc with the exact number of nonzeros. Moreover, early C (pre-PETSc) model development that fed to a MATLAB sparse matrix linear algebra package required an exact sparsity pattern. For this reason we will now present a naive algorithm for calculating the sparsity of the global Hessian of a given network.

For a stretching only system described as a set of fibres, with each fibre described as a list

of nodes:

1. Count unique nodes connected to each node in the network.

2. Allocate storage for lists of unique nodes corresponding to each node.

3. Create lists of unique nodes connected to each node.

To extend this algorithm to account for bending, the search span along a fibre must be extended beyond immediately adjacent nodes to nodes that share an adjacent node. In this way we consider pairs of segments (i.e. triplets of nodes) along the same fibre, over which bending may occur.

Depending on how the network data is structured, to count the unique nodes connected to each node, the parent fibres of each node may need to be known. If this information is not provided in the network data, an additional routine can be performed beforehand:

1. Count the fibres associated with each node in the network.

2. Allocate storage for lists of associated fibres at each node.

3. Create lists of fibres associated with each node.

Using the lists of unique nodes connected to each node, it is possible to allocate the number of nonzero elements in a single block of the Hessian. This sparsity pattern can then be copied for each of the 9 sub-blocks to generate the sparsity of the entire Hessian matrix. For compatibility with CSR format, the associated fibre and node IDs should each be stored in one long contiguous array, rather than an array of nested lists. We omit the details of our CSR implementation of this algorithm here, but refer the interested reader to Appendix A.3 for the full sparsity calculations of an example network.

4.2 Static Solver

With the intention of producing a reuseable model to predict network properties within reasonable time frames, matrix-based solvers were generally preferred over time stepping approaches. At a higher level, prior to obtaining (3.11), the system to be solved can be written as

$$\pm\,\mathbf{F} = H\mathbf{U} - \mathbf{B}^*, \tag{4.5}$$

for a force vector $\mathbf{F}$, Hessian matrix H, displacement vector $\mathbf{U}$, and RHS vector $\mathbf{B}^* = -\mathbf{B}$ (where $\mathbf{B}$ is taken from (3.11)). Unless stated otherwise, it should be assumed that the work

presented in this chapter and hereafter takes the sign convention

$$\begin{aligned} -\mathbf{F} &= H\mathbf{U} - \mathbf{B}^*, \\ \mathbf{F} &= -(H\mathbf{U} - \mathbf{B}^*), \\ \mathbf{F} &= \mathbf{B}^* - H\mathbf{U}. \end{aligned} \tag{4.6}$$

To reach (3.11) it is assumed that there exists a state of equilibrium

$$\mathbf{F} = \mathbf{0} = \mathbf{B}^* - H\mathbf{U}, \tag{4.7}$$

which can be rearranged to obtain

$$H\mathbf{U} = \mathbf{B}^*. \tag{4.8}$$

Then to solve for $\mathbf{U}$, we can write formally

$$\mathbf{U} = H^{-1}\mathbf{B}^*, \tag{4.9}$$

provided H has an inverse.

4.2.1 Direct Solution

Early model prototypes predicting the properties of smaller networks used direct methods since they offer exact solutions (subject to numerical precision) and can be run within reasonable time frames. The majority of the direct methods employed during the project derive from direct LU decomposition. This involves the factorisation of a given matrix A into lower and upper triangular matrices L and U respectively, such that $A = LU$. Using this factorisation, the system can be solved efficiently using substitution. We can write this symbolically as

$$A\mathbf{x} = (LU)\mathbf{x} = L(U\mathbf{x}) = L\mathbf{y} = \mathbf{b}. \tag{4.10}$$

From (4.10) we see that the original system is solved by solving $U\mathbf{x} = \mathbf{y}$ and $L\mathbf{y} = \mathbf{b}$, each of which is trivially solved using substitution.

LU Decomposition

The first solver that was used during the development of the model was the `linalg.solve` routine from the Python package `numpy`. This uses a LAPACK routine called `_gesv`, which is based on a standard LU factorisation. In later iterations of the model, the first prototype that was written in C called routines directly from within MATLAB. Unlike the `numpy` routine, the default direct solve routine in MATLAB (called with the backslash operator as `A\b`) varies based on the structure of the matrix it is passed. Since the appearance of the global Hessian matrix is influenced by network geometry, the solver that is called is not necessarily LU decomposition.

Prior to 3D model development in PETSc, some initial experimentation with iterative methods was carried out in MATLAB. This preliminary work was undertaken to provide an insight into potential performance gains when using iterative methods and simple block preconditioners to solve larger random networks in 2D. The methods considered are discussed in Section 4.2.2 that follows, and the results that were obtained through this investigation are discussed in Section 5.4.

Sparse LU

During the implementation changes that were required for PETSc compatibility, it was found that the standard (sparse) LU factorisation algorithm implemented in PETSc performed competitively when compared with LU based routines in both `numpy` and MATLAB (see Section 5.4.2). As the density of the random networks were increased and solve time began to noticeably increase, we investigated some of the available externally packaged sparse direct solvers that PETSc offers, namely SuperLU and MUMPS.

Super LU

In a simplified form, the SuperLU solver can be described as an algorithm of three parts. For a given system $A\mathbf{x} = \mathbf{b}$:

1. Preprocess matrix A.
2. Perform an LU factorisation of the preprocessed matrix A_p.
3. Solve the original system using the factorisation calculated in step 2.

To preprocess the matrix A, a permutation matrix P_c is chosen to reorder the columns of A with the aim of maximising the sparsity of lower and upper triangular factors L and U. Thus A_p above is defined as $A_p = AP_c$. After preprocessing A, the SuperLU algorithm uses pivoting to produce an LU factorisation defined as

$$P_r A_p = LU, \tag{4.11}$$

where P_r is a permutation matrix which reorders the rows of A.

This simplified factorisation is based on a full triangular factorisation defined as

$$P_r D_r A D_c P_c = LU, \tag{4.12}$$

with diagonal matrices D_r, D_c that are used to condition the matrix A. Using this full factorisation, the system is solved symbolically as

$$\mathbf{x} = A^{-1}\mathbf{b} = (D_r^{-1}P_r^{-1}LUP_c^{-1}D_c^{-1})^{-1}\mathbf{b} = D_c(P_c(U^{-1}(L^{-1}(P_r(D_r\mathbf{b}))))). \tag{4.13}$$

The L and U inverses use efficient linear solution for triangular sparse matrices, the permutation matrices P_r and P_c are self-inverse and reduce to sparse matrix multiplication, and the diagonal matrices D_r and D_c are trivial to invert [55, p. 5].

In the simplified algorithm $D_c = I$ and $D_r = I$, and so it follows that

$$\mathbf{x} = A^{-1}\mathbf{b} = (P_r^{-1}LUP_c^{-1})^{-1}\mathbf{b} = P_c(U^{-1}(L^{-1}(P_r\mathbf{b}))). \tag{4.14}$$

Improvements to the simplified SuperLU algorithm can be achieved by passing additional option arguments to the PETSc interface for the SuperLU solver. This allows the user to specify additional details such as the acceptable pivot threshold or whether iterative refinement (a method for iteratively improving the accuracy of the solution using the residual - see for example [21, p. 228]) should be performed. At this time of early experimentation, it was observed that MUMPS outperformed SuperLU when both solvers were given the default options, with solve time up to an order of magnitude faster for higher density networks (see Figure 5.17 in Chapter 5). For this reason, and the fact that MUMPS was able to solve higher density networks in a short time frame, no additional experimentation was undertaken with respect to the option arguments of SuperLU.

MUMPS LU

Much like the SuperLU algorithm, the MUMPS (MUltifrontal Massively Parallel Solver) can be broadly split into the three stages of preprocessing, factorising, and solving. In the MUMPS preprocessing stage, the matrix A is analysed and then *symbolically factorised*. This symbolic factorisation is distinct from the main factorisation step and can be leveraged for predictive data. Specifically an *elimination tree* is generated, which is a mapping that can be used to estimate the required memory and number of operations for the factorisation and solve step. The result of the analysis step is a preprocessed matrix A_p which is to be factorised as

$$A_p = LU \quad (\text{or } A_p = LDL^T). \tag{4.15}$$

MUMPS factorisation consists of a sequence of dense factorisations that utilise standard threshold pivoting and two by two pivoting, as well as static pivoting if specified by option flag to the PETSc interface. Following the factorisation step, MUMPS solves

$$LU\mathbf{x}_p = \mathbf{b}_p, \tag{4.16}$$

using the standard approach of forwards elimination ($L\mathbf{y} = \mathbf{b}_p$) followed by backwards elimination ($U\mathbf{x}_p = \mathbf{y}$), where $\mathbf{x}_p, \mathbf{b}_p$ have been preprocessed to be compatible with the preprocessed matrix A_p. Finally, $\mathbf{x}_p$ undergoes postprocessing to transform it back to $\mathbf{x}$. Further option flags can be provided to PETSc to perform additional steps during this postprocessing, such as iterative refinement and backwards error analysis. At the time of writing MUMPS performed

adequately "out of the box", and was utilised with all the default options.

4.2.2 Iterative Solution

Iterative solvers do not generally produce a solution after a fixed number of operations. Instead, these solvers use multiple iterations of an algorithm to move towards a desired solution, terminating after a given tolerance has been reached. More precisely, we begin with an initial guess, $\mathbf{x}_0$, of the exact solution vector, $\mathbf{x}$, and then we improve the solution using our chosen solver, generating a series of estimates $\mathbf{x}_k$ that are successively closer to the solution. Ideally, we would set a tolerance, ϵ, based on the error $\mathbf{e}_k = \mathbf{x} - \mathbf{x}_k$. Since it is not possible to calculate this without knowing the actual solution, we can choose to fix ϵ using the residual $\mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k$, so that $|\mathbf{r}_k| < \epsilon$. We say that our sequence of vectors $\mathbf{x}_k$ converge if the limit

$$\mathbf{x} = \lim_{k \to \infty} \mathbf{x}_k, \tag{4.17}$$

exists [71], and note that the convergence rate depends greatly on the choice of iterative method.

Jacobi

An example of a basic iterative solver is the *Jacobi* method. To derive this method we start by decomposing the matrix A into $D - E - F$, where D is the diagonal of A, $-E$ is the strict lower part, and $-F$ is the strict upper part. Then we can substitute this into $A\mathbf{x} = \mathbf{b}$ to give

$$\begin{aligned} (D - E - F)\mathbf{x} &= \mathbf{b}, \\ D\mathbf{x} &= \mathbf{b} + (E + F)\mathbf{x}, \\ \mathbf{x} &= D^{-1}(\mathbf{b} + (E + F)\mathbf{x}). \end{aligned}$$

Since D is the diagonal of A, it is inexpensive to invert. Thus the Jacobi method can be written as

$$\mathbf{x}_{k+1} = D^{-1}(E + F)\mathbf{x}_k + D^{-1}\mathbf{b}. \tag{4.18}$$

Krylov Subspace Methods

While the Jacobi method demonstrates the general idea behind the iterative approach to solving a linear system, in practice more sophisticated algorithms are used for large linear systems. Popular choices include the *Conjugate Gradient* (CG), MINRES, and GMRES methods, which all belong in a class of methods known as *Krylov subspace methods*. Methods in this class search for an approximate solution of $A\mathbf{x} = \mathbf{b}$ from a generated Krylov subspace. The general idea of these methods is to avoid the expensive matrix inversion A^{-1} by instead performing a series of matrix-vector multiplications. Starting from an initial guess $\mathbf{x}_0$, the k^{th} approximation $\mathbf{x}_k$ is

obtained from vectors belonging to the k^{th} *Krylov subspace* $\mathcal{K}_k$, which is generally given as

$$\mathcal{K}_k(A, \mathbf{v}) = \text{span}\{\mathbf{v}, A\mathbf{v}, A^2\mathbf{v}, \ldots, A^{k-1}\mathbf{v}\}, \tag{4.19}$$

for a square matrix A and nonzero vector $\mathbf{v}$. In this context, $\mathbf{v}$ is taken to be the residual vector for the initial guess, $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$. The suitability of each Krylov method depends on the properties of the matrix A. CG requires a symmetric positive definite matrix, whereas MINRES requires the matrix to be symmetric but not necessarily positive definite, and GMRES can be performed on asymmetric matrices.

Our system is symmetric but may not be positive definite (see Section 4.1.1), hence MINRES is a suitable candidate for an iterative method. The Krylov subspace family of methods perform best when the system is preconditioned to cluster the matrix eigenvalues. A useful measure to introduce here is the *condition number*, κ, of our matrix H, which reflects the distribution of the eigenvalues and is defined as

$$\kappa(H) = \frac{\max|\lambda_i|}{\min|\lambda_i|}, \tag{4.20}$$

where λ_i are the eigenvalues of the matrix H. To precondition our system, the goal is to select a *preconditioner*, P, which is a nonsingular matrix that can be used to transform the system into one with a more narrowly clustered distribution of eigenvalues. To demonstrate the trade off between different preconditioners, it is useful to note that $H\mathbf{U} = \mathbf{B}$ is equivalent to

$$P^{-1}H\mathbf{U} = P^{-1}\mathbf{B}. \tag{4.21}$$

Then we observe that a choice of $P = I$ is trivial to invert but results in no improvement to distribution of eigenvalues (since $\kappa(P^{-1}H) = \kappa(H)$). Alternatively, a choice of $P = H$ dramatically improves the eigenvalue distribution, giving a condition number of $\kappa(P^{-1}H) = \kappa(I) = 1$, but would require an expensive matrix inversion that would allow us to solve the original system.

Exploiting the block structure seen in Equation (4.3), we can consider the following initial choices of preconditioner when it later comes to solving systems in chapters 5 and 6. Since preconditioning was first investigated after completion of the 2D model, we will give the preconditioner descriptions in two dimensions, but the extension to their three dimensional counterparts will be evident and immediate.

Using a block reduction of the original matrix system we can define

$$P_{AS} = \begin{bmatrix} H_{xx} & H_{xy} \\ O & S \end{bmatrix}, \tag{4.22}$$

where the Schur complement matrix $S = H_{yy} - H_{xy}^T H_{xx}^{-1} H_{xy}$. An alternative iterative method to MINRES, such as GMRES, is needed if P_{AS} was used since this preconditioner does not preserve the symmetry of H.

In practice the Schur complement is too expensive to compute and this preconditioner, although effective in reducing iteration count, would increase the expense of the overall solution algorithm. This methodology has proved effective in other fields, notably incompressible flow [20], where a cheaper approximation to the Schur complement matrix can be made.

We note that the leading term of the Schur complement is H_{yy} so that we can use this as an approximation in the following two forms:

$$P_{Ab} = \begin{bmatrix} H_{xx} & H_{xy} \\ O & H_{yy} \end{bmatrix}, \quad P_{Db} = \begin{bmatrix} H_{xx} & O \\ O & H_{yy} \end{bmatrix}. \tag{4.23}$$

The non-symmetric form P_{Ab} would require the use of the GMRES iterative method. The symmetric form P_{Db} which drops the H_{xy}, off-diagonal block, would allow the cheaper MINRES method to be used. We will explore the behaviour of these preconditioners in Chapter 5.

4.3 Time Stepping Approaches

An alternative approach for solving the system (4.5), is to approximate the entries of global solution vector $\mathbf{U}$ via time stepping. This eliminates the need to calculate a matrix inverse, and instead iteratively steps towards the solution by introducing a second equation

$$\mathbf{U} \rightarrow \mathbf{U} + \alpha \mathbf{F}, \tag{4.24}$$

for some positive, nonzero, scalar α that acts as a step size. At every step, $\mathbf{F}$ and $\mathbf{U}$ are updated and this is repeated until $|\mathbf{F}| < \epsilon$, for some tolerance ϵ. The stability of the system depends heavily on α, with the system converging extremely slowly for small values and diverging for large values. This approach is effectively a *forward Euler* scheme for the (pseudo-)time dependent equation,

$$\frac{d\mathbf{U}}{dt} \;=\; \mathbf{F} \tag{4.25}$$

and can be rewritten as

$$\begin{aligned} \mathbf{F}(t_k, \mathbf{U}_k) &= \mathbf{B}^* - H\mathbf{U}_k, \\ \mathbf{U}_{k+1} &\rightarrow \mathbf{U}_k + \alpha \mathbf{F}(t_k, \mathbf{U}_k), \end{aligned} \tag{4.26}$$

where each $\mathbf{U}_{k+1}$ is calculated explicitly in terms of $\mathbf{U}_k$.

During this investigation for a time stepping based approach, α was generally kept constant, but further work could employ more complex schemes such as dynamic assignment of α, where its value changes based on the convergence rate.

4.3.1 Elastic Medium Embedding

We develop this argument for the time stepping method just described, in the understanding that the static solvers discussed in Section 4.2 are equally applicable. In the event of a network having insufficient rigidity, direct methods may fail due to there being no unique solution. One approach that may be used to resolve this is singular value decomposition [21, p. 299], however this is computationally expensive and was not explored over the duration of the project due to the difficulty of extending to large networks. An alternative approach is to embed the network in an elastic medium. To do this, one can assume that all nodes associated with the network have an additional spring connected to their initial positions. If the spring constant is λ, the restoring force vector for all internal nodes would be

$$\mathbf{F} = -\lambda \mathbf{U}. \tag{4.27}$$

Based on the sign convention used when developing the model, recall from (4.6) that

$$\mathbf{F} = \mathbf{B}^* - H\mathbf{U}. \tag{4.28}$$

Now rather than assuming a state of equilibrium, the restoring force vector must be taken into account. In order to keep the system in equilibrium, this force vector is added to both sides of (4.28) to find

$$-\lambda \mathbf{U} = \mathbf{B}^* - H\mathbf{U} - \lambda \mathbf{U}, \tag{4.29}$$

$$-\lambda \mathbf{U} = \mathbf{B}^* - (H + \lambda I)\mathbf{U}, \tag{4.30}$$

thus the correction needed is

$$H \to H + \lambda I, \tag{4.31}$$

and the scheme seen in (4.26) is updated to

$$\begin{aligned} \mathbf{F}_k &= \mathbf{B}^* - (H + \lambda I)\mathbf{U}_k, \\ \mathbf{U}_{k+1} &\to \mathbf{U}_k + \alpha \mathbf{F}_k, \end{aligned} \tag{4.32}$$

where H is only corrected once prior to time stepping and used at each step without the need for recalculation. We note that (4.31) would be $H \to H - \lambda I$ if we instead had $\mathbf{F} = H\mathbf{U} - \mathbf{B}^*$ from an alternative sign convention.

A potential issue with associating every internal node of the network with a spring is that there is a risk of increased tension and thus unusual deformations near the boundaries where shear strain has been applied to the original network. To ameliorate this, the approach can be developed further by imposing an affine shear on the weak elastic medium.

Assuming that the weak elastic medium deforms in an affine manner, the restoring force

vector can be updated to

$$\mathbf{F} = -\lambda(\mathbf{U} - \mathbf{U}^{\text{aff}}). \tag{4.33}$$

As before the system must be kept in equilibrium, and so

$$-\lambda(\mathbf{U} - \mathbf{U}^{\text{aff}}) = \mathbf{B}^* - H\mathbf{U} - \lambda(\mathbf{U} - \mathbf{U}^{\text{aff}}), \tag{4.34}$$

$$-\lambda(\mathbf{U} - \mathbf{U}^{\text{aff}}) = \mathbf{B}^* + \lambda\mathbf{U}^{\text{aff}} - (H + \lambda I)\mathbf{U}, \tag{4.35}$$

thus since $\mathbf{U}^{\text{aff}}$ can be calculated prior to solving the system, H is corrected as before and the RHS vector can be corrected to

$$\mathbf{B}^* \to \mathbf{B}^* + \lambda\mathbf{U}^{\text{aff}}. \tag{4.36}$$

Thus the improved scheme that attempts to reduce tension at boundaries is

$$\begin{aligned} \mathbf{F}_k &= \mathbf{B}^* + \lambda\mathbf{U}^{\text{aff}} - (H + \lambda I)\mathbf{U}_k, \\ \mathbf{U}_{k+1} &\to \mathbf{U}_k + \alpha\mathbf{F}_k, \end{aligned} \tag{4.37}$$

where $\mathbf{U}^{\text{aff}}$ is calculated and H, $\mathbf{B}^*$ are corrected only *once* prior to time stepping. Again, we note that if $\mathbf{F} = H\mathbf{U} - \mathbf{B}^*$, then $\mathbf{B}^* \to \mathbf{B}^* - \lambda\mathbf{U}^{\text{aff}}$.

Summary

This chapter concludes the methodology that is required in order to begin discussing the application of the model to given networks. While the previous chapter laid the foundation for understanding how the mechanical behaviour of a random fibre network can be discretised and assembled into a large linear system, this chapter introduced how to extract this mechanical behaviour after some external load or force is applied to the network, through discussion of matrix representation and the approaches for solving the global system.

The discussion of the direct methods presented here represents a timeline of LU based approaches that were explored as the model was gradually developed over the course of the research. By outlining iterative methods in the context of the global system previously derived, we have given a starting point for iterative experimentation that can be applied to two dimensional networks discussed in Chapter 5. In the final section of the chapter we have given an overview of time stepping based approaches, and more specifically the use of an elastic medium, which has applicability to two dimensional networks with low connectivity, but holds more significance for the discussion of random fibre networks in three dimensions that are close to the bending rigidity threshold.

Chapter 5

Modelling Networks in 2D

In the previous chapter we outlined the approaches that can be used to solve the system derived in the chapter before that. We gave particular focus to direct LU factorisation based methods for solving the system. If not explicitly stated, it should be assumed that the solutions of given networks presented here and thereafter were obtained using an LU factorisation or similar. Using the necessary background and methodologies of the previous chapters, we will now present the work undertaken on modelling fibre networks in 2D.

While random fibre networks are generally well understood in two dimensions, this chapter will serve as an important foundation for the following chapter. Moreover, we will also give an overview of some the issues that may be encountered in two dimensional modelling and, where possible, review some proposed solutions to these difficulties. The structure of the chapter will broadly cover three areas: network representation, lattice based 2D network modelling, and off-lattice 2D network modelling.

The section on network representation will detail the general procedure for generating networks in 2D, and discuss some of the specific considerations encountered during this process. Following the discussion of network representation topics, we will walk through a sample of networks used for validating the model. Some of the validation networks covered here are also directly applicable to Chapter 6 upon extension of the given network into three dimensions. An important example that will be evaluated in detail along with several variations is that of the self-evidently named λ-network. This example network was chosen early on as a suitable candidate for validating many core functionalities of the model in one simple network. Building from this, we will move on to lattice based triangular meshes. Here we discuss the algorithms used to generate such networks, provide comparison with literature, and present an original analysis of preconditioner performance. The final section will move to fully random 2D networks, validating known results against literature, and presenting novel work that has been published in [36].

5.1 Network Representation in 2D

With what has been discussed thus far, we already have a basic notion of what a network is within the context of what information can be extracted from it and used to predict mechanical behaviour. In some sense, throughout this process there has been an implicit assumption that the information we extract is drawn from some pre-existing network. This assumption is valid, as the model was designed with the motivation of predicting the mechanical properties of real world random fibre networks. However, there is a step inbetween in which we need to generate a meaningful discretised network that can be fed to the model. It is this step that we will now elaborate on the details of, and that will be the focus of this section. We will break the discussion of network generation into sub tasks that will be covered in their own respective sections. First we will outline the basic network generation routine at a high level. Then we will discuss the changes required for the fibre generation process to incorporate a randomised geometry. After this we will elaborate on how individual fibres can be connected together and how we choose to represent these connections. From this we can specify all of the permitted node types and look at some of the additional design decisions that should be considered in two dimensions. Finally we will conclude the section with a brief look at how these networks are reduced to a minimal amount of information that can be written to file, passed to the predictive model and reconstructed for visualisation purposes.

5.1.1 Generating Networks

The process for generating networks in 2D can be summarised as follows:

1. Generate a box of some given dimensions. This box will be the domain that contains the network.
2. Generate a pair of coordinates that will form the end points of a fibre.
   - For aperiodic systems, both coordinates must be completely contained by the box.
   - For periodic systems, the two coordinates may lie outside the box, but the coordinate pair should be translated to lie within the domain, and the fibre to be formed from the pair of coordinates should be formed between the closest periodic copy of these coordinates. We will elaborate on this and outline an algorithm that handles this automatically in Section 5.1.2.
3. Join the coordinate pair to form a fibre object. The fibre is assigned a radius that is used to determine its mechanical properties. We assume regular cylindrical objects, thus the radius is constant throughout.
4. Repeat steps 2-3 for as many fibres as desired.
5. Run a suitable connection routine (see Section 5.1.3) to identify nodes at which fibres intersect.

6. Collect the necessary information for reconstructing the network and use this data to write the network to a file (see Section 5.1.6).

5.1.2 Mikado Networks

The generation process detailed in Section 5.1.1 reveals no details about whether the network to be constructed follows an on-lattice architecture or a more randomised geometry. For networks with a regular structure, the coordinate pairs generated at step 2 can be chosen to lie on a desired lattice. For off-lattice networks, we adopt the Mikado approach. Here we will discuss the method in the context of 2D networks, and in Chapter 6 we will discuss how the method can be extended to 3D.

To build a Mikado network of N_f total fibres, for each of the N_f fibres we randomly generate a *single* position vector, $\mathbf{m}$, with coordinates (m_x, m_y) that lie within the dimensions of the box. Pairing $\mathbf{m}$ with a randomly generated unit direction vector, $\hat{\mathbf{n}}$, the end points of the fibre can be generated for some given length, ℓ by using $\mathbf{m}$ as a midpoint [see Figure 5.1 (i)]. Specifically, the end points are defined as

$$\begin{aligned} \mathbf{s} &= \mathbf{m} - \frac{\ell}{2}\hat{\mathbf{n}}, \\ \mathbf{e} &= \mathbf{m} + \frac{\ell}{2}\hat{\mathbf{n}}. \end{aligned} \tag{5.1}$$

Using the end points, a fibre can be formed as in step 3 of Section 5.1.1, but since we have only asserted that the midpoint should lie within the box, it is possible that $\mathbf{s}$ or $\mathbf{e}$ may lie outside the domain. Depending on periodicity, this should be handled in two different ways. For the *periodic* case, $\mathbf{s}$ and $\mathbf{e}$ should be wrapped back to a periodic copy of themselves that lies within the box [see Figure 5.1 $(ii - iii)$]. Note that the fibre should only be wrapped to be inside the

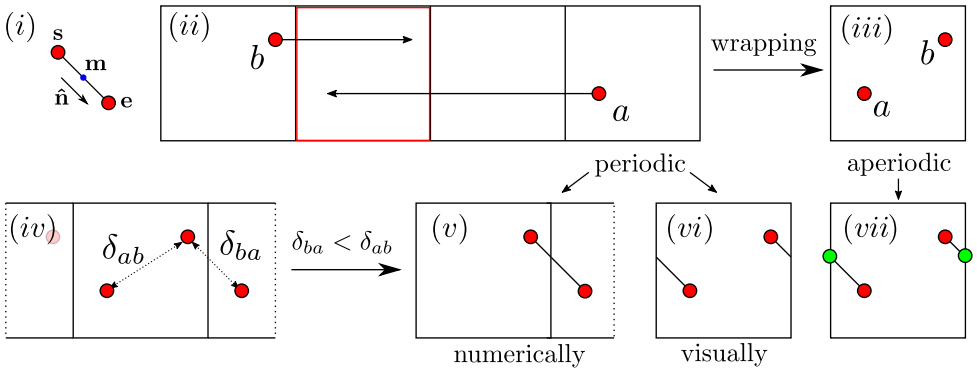


Figure 5.1: An overview of some of the visual and numerical considerations involved in node and fibre wrapping when generating Mikado networks. Part (iv) demonstrates how considering different periodic copies of a node could result in a different fibre being constructed.

dimensions of the box for network visualisation purposes as in Figure 5.1 (*vi*). When passing the fibre information to numerical model, the fibre should be represented as a continuous object [see Figure 5.1 (*v*)]. Therefore it could be that one of the two end points lies outside of the box dimensions as in Figure 5.1(*v*).

For the *aperiodic* case these nodes should be truncated [see Figure 5.1 (*vii*)]. To truncate the fibres, the end point of the fibres that lies beyond the domain of the box should be replaced with a new end point calculated at the point of intersection between the fibre and boundary, such that the fibre preserves its existing orientation.

5.1.3 Cross-linking Fibres in 2D

Having assembled a structure of slender elastic fibres in a two dimensional plane, we are left to cross-link the network into a mechanical structure which we may discretise into segments and nodes. To cross-link a given network in 2D, we loop over every pair of fibres and consider their intersection. Since each pair of lines lie within the same two dimensional plane, it follows that the lines must be parallel or intersect at a unique point. Provided the lines intersect, it is determined whether the point of intersection is within the range of the finite fibre lengths. For cases when the point of intersection lies on both fibres, this unique point is identified as a node of the fibre pair.

To elaborate on the details of this 2D intersection algorithm that has been implemented in the model, it is useful to begin with two arbitrary lines written in parametric notation.

$$\mathbf{r} = \mathbf{a} + \lambda \mathbf{b} \tag{5.2}$$

$$\mathbf{s} = \mathbf{c} + \mu \mathbf{d}, \tag{5.3}$$

where λ, μ parameterise points along each line, and $\mathbf{b}, \mathbf{d}$ are unit vectors. At a unique point of intersection, we have the case that $\mathbf{r} = \mathbf{s}$ or equivalently

$$\mathbf{a} - \mathbf{c} + \lambda \mathbf{b} - \mu \mathbf{d} = 0, \tag{5.4}$$

which can be written in terms of components as

$$a_x - c_x + \lambda b_x - \mu d_x = 0 \tag{5.5}$$

$$a_y - c_y + \lambda b_y - \mu d_y = 0. \tag{5.6}$$

Alternatively we can express the components as the matrix system

$$\begin{bmatrix} b_x & -d_x \\ b_y & -d_y \end{bmatrix} \begin{bmatrix} \lambda \\ \mu \end{bmatrix} = \begin{bmatrix} c_x - a_x \\ c_y - a_y \end{bmatrix}. \tag{5.7}$$

To solve for λ, μ, the matrix must be invertible, which is true for $\delta = b_y d_x - b_x d_y \neq 0$ where δ

is the determinant of the matrix. If $\delta = 0$ then $\mathbf{r}, \mathbf{s}$ are parallel, since

$$b_y d_x - b_x d_y = 0 \implies |\mathbf{b} \times \mathbf{d}| = 0.$$

In this case, the lines $\mathbf{r}$ and $\mathbf{s}$ are collinear and overlap, collinear and intersect at a unique point, or do not intersect. To avoid a near zero division resulting from a small δ, we test for $|\mathbf{b} \times \mathbf{d}| < \nu_{2D}$, where ν_{2D} is sufficiently close to 0 (generally on the order of 10^{-8}) and then disregard any lines that are approximately parallel for cross-linking. For $\delta \neq 0$, we can invert the matrix

$$\begin{bmatrix} \lambda \\ \mu \end{bmatrix} = \frac{1}{b_y d_x - b_x d_y} \begin{bmatrix} -d_y & d_x \\ -b_y & b_x \end{bmatrix} \begin{bmatrix} c_x - a_x \\ c_y - a_y \end{bmatrix}, \tag{5.8}$$

to find the values of λ, μ

$$\lambda = \frac{(c_x - a_x)d_y - (c_y - a_y)d_x}{b_x d_y - b_y d_x}, \quad \mu = \frac{b_x(c_y - a_y) - b_y(c_x - a_x)}{d_x b_y - d_y b_x}, \tag{5.9}$$

which in two dimensions can be given in vector notation as

$$\lambda = \frac{|(\mathbf{c} - \mathbf{a}) \times \mathbf{d}|}{|\mathbf{b} \times \mathbf{d}|}, \quad \mu = \frac{|\mathbf{b} \times (\mathbf{c} - \mathbf{a})|}{|\mathbf{d} \times \mathbf{b}|}. \tag{5.10}$$

Therefore, in the case where $\mathbf{r}$ and $\mathbf{s}$ are not parallel, the unique point of intersection is found by calculating λ, μ from (5.9) or (5.10) and substituting back into (5.2) and (5.3). After calculating the unique point of intersection, it is necessary to determine whether this point lies within the finite lengths of each of the fibres f_i, f_j, being considered. That is whether

$$0 \leq \lambda \leq |f_i|, \tag{5.11}$$

$$0 \leq \mu \leq |f_j|. \tag{5.12}$$

5.1.4 Node Classification

During the discretisation process, in which a network is decomposed into finite segments and nodes, nodes themselves can be categorised into 3 distinct types that have different implications that must be accounted for in the model implementation. The three node classifications are *internal*, *boundary*, and *dangling*.

- An *internal* node is a freely-hinged cross-link occurring at the point of intersection of two fibres.
- A *boundary* node is a freely-hinged cross-link occurring at the point of intersection of a fibre end and the boundary of the plane.
- A *dangling* node is a freely dangling end point of a fibre.

Any node n_i of a specific fibre f is a child of f and we may refer to f as a parent of n_i. It is immediately apparent from the nature of cross-linking in 2D that a node n_i may have more than one parent fibre if it is not dangling or boundary.

The decision to include dangling nodes was purely aesthetical, left as part of the network post-generation to provide visualisations that match real-world networks more closely. As dangling nodes do not contribute any energy to the system when the network is deformed, they are ignored during the assembly stage, and for visualisation purposes must be updated separately. The details of the algorithm for updating the positions of dangling nodes are provided in Appendix B.3.

5.1.5 Node Nearness in 2D

For random networks of fixed dimension and fibre length but increasing fibre density, there is an increasing probability of encountering clusters of nodes within close proximity, as seen in Figure 5.2.

The motivation for why this is problematic is discussed in Section 5.2.3. Here will we focus instead on discussing methods that can be employed to prevent this. Throughout the network generation process there are three separate stages at which node nearness can be considered that are immediately transferable to 3D. We will discuss each of these stages individually.

Fibre Generation

Referring to Section 5.1.2, each new fibre is generated from a random coordinate representing the midpoint of the fibre and a random direction vector. As soon as the start and end coordinates are determined from the midpoint and direction, we can compare the distance from each of these coordinates to the coordinates of all existing fibre end points. If this distance is within some tolerance, ν_e, we discard this new fibre before it has been saved to the `box` object, and continue to the generation of the next fibre. Here the chosen tolerance represents the radius of a circle (or sphere in 3D) centered at the respective fibre end point. Since the probability of an off-lattice network having a direct intersection at the ends of two fibres is neglible, we can use this approach to provide a guarantee that no two fibres will be generated with end points within a given distance. This approach is implemented at the fibre generation stage, and therefore does not alter the distances between internal nodes of nearby fibres, or between nodes within the same fibre.

It should be noted that selecting a small tolerance for discarding nearby fibres is important for keeping a short runtime. As fibre regeneration is random, running the algorithm for sufficiently dense networks of fibres will result in a rapidly increased runtime as the program attempts to obtain exactly N_f fibres by regenerating illegal fibres in a domain with less and less suitable locations for each new fibre added.

By ending the algorithm after k attempted fibre generations, we can avoid the risk of a

longer runtime, but as the number of fibres $N_f \leq k$, we cannot know the value of N_f until after the algorithm has run.

Cross-link Generation

During the cross-link generation stage, new nodes are progressively introduced along existing fibres. To prevent nodes being generated too closely at this stage, for each fibre pair identified as directly intersecting, we can calculate distance from the new intersection point to the coordinates of every existing node of the respective fibre pair. If the distance is within a given tolerance, ν_m, then we can replace the actual point of intersection with the coordinates of the *closest* existing node at the cost of a small error. Since substituting the actual intersection coordinates with nearby coordinates will generate a prestress in the fibre (due to the loss of colinearity of consecutive nodes), which we choose to disregard, it is important to select a relatively small tolerance. An analysis of the error introduced through node merging is covered in Section 5.2.4.

Fibre Truncation

When fibres are truncated at aperiodic boundaries, the existing end point is updated to a position along the fibre contained within the domain. This new position could be close to another existing node on the fibre, and can be checked for using the same procedure of node searching along a fibre as in the cross-link generation stage. Nearby nodes can be largely avoided at the truncation stage by ensuring that truncation is done only once during network generation and is performed prior to any additional nodes being added internally to a fibre.

Discussion

Whilst intuitively it should be possible to use node merging to achieve a minimum cross-link distance $\ell_c^{\min}$ in 2D, and a minimum mesh size $\xi^{\min}$ in 3D, it is not an automatic consequence in two dimensions.

To elaborate, consider the example in Figure 5.2 *(left)*. Here we observe a new point of intersection that has been identified at P, which lies some distance from nodes n_i and n_j. Following the steps outlined above any node, n_k, which satisfies the constraint $|n_k - P| < \nu_m$ for search tolerance, ν_m, may become the new candidate to take the place of intersection point P. Assume that we begin by comparing P with every node on f_i. When we compare n_i with P we note that $|n_i - P| < \nu_m$, and therefore update our search tolerance to $\nu'_m = |n_i - P|$, marking n_i as the current node for P to be merged to. Assuming no other nodes on the first fibre are found to be closer, the algorithm begins searching along fibre f_j. Upon comparing P with n_j, the tolerance is updated to $\nu''_m = |n_j - P|$ and n_j is the new candidate for P to be merged into. Having completed its search along both associated fibres, P is merged with n_j

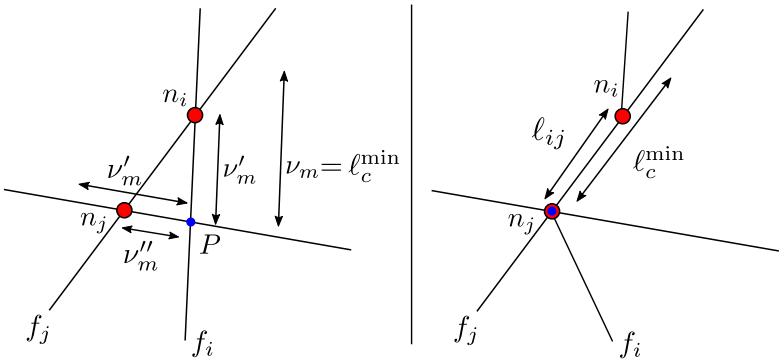


Figure 5.2: A network to demonstrate the case where $\ell_c^{\min} > \ell_{ij}$, for some segment n_i, n_j before *(left)* and after *(right)* node merging has occurred. The relative difference in size between ν'_m and ν''_m is dictated by the relative angle between f_i and f_j.

and we are left with Figure 5.2 *(right)*. Thus, since $\nu''_m < \nu'_m < \nu_m$ but only n_j was merged, the distance between n_i and the final position of P is still less than the original value of ν_m.

The likelihood of this situation occurring increases with the overall density of a given network, and becomes problematic in high density networks where it is useful to enforce such a minimum distance. One solution is to perform an additional pass over every segment of every fibre once all cross-links have been introduced. However, to maintain consistent fibre node list orderings, every fibre affected by a node merge must now be modified (whereas previously node merging was taking place *prior* to the creation of a node). By tracking the parent fibres of every node, the loop can be performed over the affected fibres. Otherwise, every fibre must be considered to search for the node being merged.

5.1.6 Network Data

For the model to apply an external force to a given network and solve for the resulting internal displacements, it needs to be provided with some basic network data. Using this same data it should also be possible to reconstruct and visualise the network before and after being put through the model.

Perhaps the most fundamental building block of the network is the node. For every node in the network, we need to know its respective coordinates (x, y) before any external force has been applied. For compatibility with 3D networks, we store these coordinates in a 3D vector (`x`,`y`,`z`) and set $z = 0$. To understand the respective contribution of a node to the network, we also need to know its `type`, which can be internal (`0`), dangling (`1`), or boundary (`2`). Finally to understand its relationship with other nodes and to which fibre it belongs, it should have a unique identifier or `id`. Combining all this information, we can represent a `node` object as single line of data separated by whitespace

`n id type x y z`.

Using a combination of nodes, we can form our fibres as ordered lists of nodes. To do this, it is sufficient to list each nodes respective unique integer ID in the order in which they appear along the fibre. Additionally, each fibre should also have a unique identifier, `id`, as well as a radius, `rad`, for providing a metric of stiffness when combined with its respective length. The length itself can be calculated from the difference in coordinates of the two end nodes of the fibre, and therefore does not need to be stored directly as a part of the fibre object. Thus combined, a single `fibre` object can be represented as the line of data

`f id rad nA nB ... nZ`.

Finally, while the fibres are already intrinsically linked to the nodes, this a one directional relationship that results in node double counting if one is to loop over the nodes via the node lists of the fibres. Furthermore the node lists of the fibres contain no node information themselves but rather *point* to the nodes which exist elsewhere. It is therefore more practical to provide a third object that will contain master lists of all the fibres and all the nodes. This object is the `box` that consists of all the `fibre` and `node` objects, but also any other additional information required to reconstruct the network. Specifically, the box should store the dimensions of the space containing the network (`xDim`, `yDim`, `zDim`), and the periodicity of the network (`xPer`,`yPer`, `zPer`). The periodicity of a network in a particular direction is represented by an integer value for that respective direction of `0` for aperiodic and `1` for periodic. To construct the master node and fibre lists of the network, and to inform the respective software (e.g. model or visualiser) how many lines of node and fibre information to expect when reading in from file, we also need to provide the total number of nodes (`numN`) and fibres (`numF`)

`b numN numF xDim yDim zDim xPer yPer zPer`.

5.2 Test Networks in 2D

As an early form of validation, it is valuable to consider various toy problems that can be used as soon as the first full model prototypes are complete. The goal here is to develop a set of test networks that are not necessarily representative of the real-world networks of interest, but have well defined solutions that can either be cross checked with hand calculations, or be simple enough networks that educated assumptions and predictions can be made.

The first of these networks that was a particularly useful toy problem in two dimensions was the so called λ-network. This network takes its name from its geometry and is formed by intersecting two fibres that are perpendicular to each other. One fibre, f_{cd}, spans the box, intersecting both boundaries and the other, f_{ab}, intersects only with the bottom boundary, as seen in Figure 5.3. These fibres contain the nodes n_a, n_e, n_b and n_c, n_e, n_d respectively, which have coordinates described by their corresponding position vectors, which for this λ-network

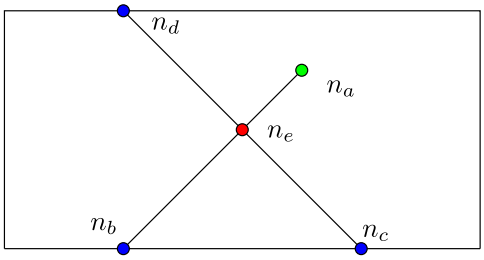


Figure 5.3: A basic λ-network, with a dangling node n_a.

are

$$\begin{aligned} \mathbf{s}_a &= [2.5, 1.5]^T,\ \mathbf{s}_b = [1.0, 0.0]^T,\ \mathbf{s}_e = [2.0, 1.0]^T, \\ \mathbf{s}_c &= [3.0, 0.0]^T,\ \mathbf{s}_d = [1.0, 2.0]^T. \end{aligned} \tag{5.13}$$

As nodes n_b, n_c, n_d, lie on a boundary we can define their respective displacement vectors for shear strain γ applied at the upper boundary $y = 2$ where $y = 0$ is fixed

$$\begin{aligned} \mathbf{u}_b &= [\gamma s_{b_y}, 0.0]^T, \quad \mathbf{u}_c = [\gamma s_{c_y}, 0.0]^T, \quad \mathbf{u}_d = [\gamma s_{d_y}, 0.0]^T, \\ &= [0.0,\ 0.0]^T, \qquad = [0.0,\ 0.0]^T, \qquad = [2\gamma,\ 0.0]^T. \end{aligned} \tag{5.14}$$

Nodes n_a, n_e lie within the domain and therefore $\mathbf{u}_a, \mathbf{u}_e$ are unknown. From the position vectors given in (5.13) we can calculate the length of the segment between nodes n_a and n_e

$$\ell_{ae} = |\mathbf{s}_e - \mathbf{s}_a| = \sqrt{\left(-\frac{1}{2}\right)^2 + \left(-\frac{1}{2}\right)^2} = \frac{1}{\sqrt{2}}, \tag{5.15}$$

then similarly we can define the remaining segments

$$\ell_{eb} = \ell_{ce} = \ell_{ed} = \sqrt{2}. \tag{5.16}$$

5.2.1 Lambda Case Stretch Only Validation

Calculate Local Stretching Contributions

Using the information provided above and in Figure 5.3, we can begin to assemble the individual contributions to the stretching energy of every segment considered in isolation. Let us start by evaluating the segments belonging to f_{ab}.

Segment $n_a n_e$ consists of a *dangling* node, n_a, and so we have a freely-hinged segment about the *internal* node, n_e. Since segment $n_a n_e$ is only constrained at node n_e the segment does not contribute to the total stretching energy of the system, irrespective of the type of

n_e. Furthermore since we assume freely rotating nodes, none of the segments of the network contribute rotational (or, in 3D, torsional) energy to the total energy of the system. As a result, we do not assemble the local system for this segment. The model identifies cases like this by checking whether either of the nodes in the pair are of *dangling* type prior to assembly.

Moving on to segment $n_e n_b$, it consists of *internal* node n_e and *boundary* node n_b. Thus, since the segment is constrained at each end, it can contribute energy to the system through stretching or compressing, and we assemble the local system by substituting the respective values into (3.42) and (3.43), which yields

$$H_{eb}^{\text{stretch}} = \frac{1}{2}\frac{\mu}{\ell_{eb}}\begin{bmatrix} 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \end{bmatrix}, \quad \mathbf{B}_{eb}^{\text{stretch}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{5.17}$$

where $\mu = \pi r^2 E^f$, and the radius, r, is assumed to be constant for all fibres of the network. Next we consider segment $n_c n_e$, consisting of *internal* node n_e and *boundary* node n_c. Thus as above we assemble local system

$$H_{ce}^{\text{stretch}} = \frac{1}{2}\frac{\mu}{\ell_{ce}}\begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}, \quad \mathbf{B}_{ce}^{\text{stretch}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{5.18}$$

Lastly, for final segment $n_e n_d$, of *internal* and *boundary* respectively, we have the local system

$$H_{ed}^{\text{stretch}} = \frac{1}{2}\frac{\mu}{\ell_{ed}}\begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}, \quad \mathbf{B}_{ed}^{\text{stretch}} = \frac{\gamma\mu}{\ell_{ed}}\begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \end{bmatrix}. \tag{5.19}$$

Assembling the Global System

In the λ-network we are considering a system with one internal node that is constrained by both of the opposing y boundaries. It follows that for 2 degrees of freedom, our global Hessian will be 2×2, and each of the local stretching systems will contribute one value, associated with n_e, from each of their 4 sub-blocks (as discussed in Section 3.4.2). Hence for 3 local matrices, there will be 12 contributions added to the global Hessian in total. For example, in H_{eb}^{stretch} there are 4 sub-blocks,

$$H_{eb_{xx}}^{\text{stretch}}, H_{eb_{xy}}^{\text{stretch}}, H_{eb_{yx}}^{\text{stretch}}, \text{ and } H_{eb_{yy}}^{\text{stretch}}. \tag{5.20}$$

and from each of these, we must the retrieve the single relevant contribution which is respectively

$$\frac{\partial^2 E_{eb}^{\text{stretch}}}{\partial u_{e_x} \partial u_{e_x}}, \frac{\partial^2 E_{eb}^{\text{stretch}}}{\partial u_{e_x} \partial u_{e_y}}, \frac{\partial^2 E_{eb}^{\text{stretch}}}{\partial u_{e_y} \partial u_{e_x}}, \text{ and } \frac{\partial^2 E_{eb}^{\text{stretch}}}{\partial u_{e_y} \partial u_{e_y}}. \tag{5.21}$$

Each of these are added to the existing values at the corresponding locations of the global Hessian, which for this network takes the form

$$H = \begin{bmatrix} \dfrac{\partial^2 E}{\partial u_{e_x} \partial u_{e_x}} & \dfrac{\partial^2 E}{\partial u_{e_x} \partial u_{e_y}} \\ \dfrac{\partial^2 E}{\partial u_{e_y} \partial u_{e_x}} & \dfrac{\partial^2 E}{\partial u_{e_y} \partial u_{e_y}} \end{bmatrix}. \tag{5.22}$$

Similarly for $\mathbf{B}_{eb}^{\text{stretch}}$, we retrieve $\frac{\partial E_{eb}^{\text{stretch}}}{\partial u_{e_x}}$, and $\frac{\partial E_{eb}^{\text{stretch}}}{\partial u_{e_y}}$, and add each of these to the existing values at the corresponding locations of the global RHS vector

$$\mathbf{B} = \begin{bmatrix} \dfrac{\partial E}{\partial u_{e_x}} \\ \dfrac{\partial E}{\partial u_{e_y}} \end{bmatrix}. \tag{5.23}$$

Performing this process for all local systems, (5.22) and (5.23) can be given as

$$H = \frac{\mu}{2} \begin{bmatrix} \frac{1}{\ell_{eb}} + \frac{1}{\ell_{ce}} + \frac{1}{\ell_{ed}} & \frac{1}{\ell_{eb}} - \frac{1}{\ell_{ce}} - \frac{1}{\ell_{ed}} \\ \frac{1}{\ell_{eb}} + \frac{1}{\ell_{ce}} + \frac{1}{\ell_{ed}} & \frac{1}{\ell_{eb}} - \frac{1}{\ell_{ce}} - \frac{1}{\ell_{ed}} \end{bmatrix}, \quad \mathbf{B} = \frac{\gamma\mu}{\ell_{ed}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}. \tag{5.24}$$

In this case $\ell_{eb} = \ell_{ce} = \ell_{ed} = \sqrt{2}$, and we simplify this to

$$H = \begin{bmatrix} \frac{3\sqrt{2}\mu}{4} & -\frac{\sqrt{2}\mu}{4} \\ -\frac{\sqrt{2}\mu}{4} & \frac{3\sqrt{2}\mu}{4} \end{bmatrix}, \quad \mathbf{B} = \frac{\gamma\mu}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \tag{5.25}$$

which can be solved algebraically to give global solution vector $\mathbf{U}$

$$\mathbf{U} = \begin{bmatrix} \frac{\gamma}{2} \\ -\frac{\gamma}{2} \end{bmatrix}. \tag{5.26}$$

We see from this that the solution of a stretch-only system is independent of μ and therefore independent of the radius r, since μ depends on r and E^f. From this solution vector (5.26), we have the x and y displacements of n_e. Combining this with known displacements of the boundary nodes n_b, n_c, n_d, we can substitute into (3.34) to calculate the total stretching energy of the network. Our numerical model agrees with the displacement vector prediction, $\mathbf{u}_e$, up to the numerical precision of PETSc's default `VecView` output (generally 6 significant figures), and agrees with the energy predictions up to a numerical precision of 16 decimal places [4, p. 215].

5.2.2 Lambda Case Stretch & Bend Validation

Calculate Local Bending Contributions

To assemble the individual contributions to the bending energy, we loop over pairs of consecutive segments of the fibres. In the case of the λ-network, fibre f_{ab} (consisting of the node triplet, n_a, n_e, n_b), does not bend since it contains the unconstrained dangling node n_a. Therefore n_c, n_e, n_d, is the only consecutive pair of segments that can contribute to the bending, and we assemble the local bending system

$$H^{\text{bend}}_{ced} = \frac{\kappa\sqrt{2}}{8}\begin{bmatrix} 1 & -2 & 1 & 1 & -2 & 1 \\ -2 & 4 & -2 & -2 & 4 & -2 \\ 1 & -2 & 1 & 1 & -2 & 1 \\ 1 & -2 & 1 & 1 & -2 & 1 \\ -2 & 4 & -2 & -2 & 4 & -2 \\ 1 & -2 & 1 & 1 & -2 & 1 \end{bmatrix}, \quad \mathbf{B}^{\text{bend}}_{ced} = \kappa\gamma\frac{\sqrt{2}}{4}\begin{bmatrix} -1 \\ 2 \\ -1 \\ -1 \\ 2 \\ -1 \end{bmatrix}, \tag{5.27}$$

Updating the Global System

Assuming we have already assembled all the stretching contributions as in (5.25), then we are left to add the values of (5.27) that associated with the global system, which are

$$\frac{\partial^2 E^{\text{bend}}_{ced}}{\partial u_{e_x}\partial u_{e_x}}, \frac{\partial^2 E^{\text{bend}}_{ced}}{\partial u_{e_x}\partial u_{e_y}}, \frac{\partial^2 E^{\text{bend}}_{ced}}{\partial u_{e_y}\partial u_{e_x}}, \frac{\partial^2 E^{\text{bend}}_{ced}}{\partial u_{e_y}\partial u_{e_y}}, \frac{\partial E^{\text{bend}}_{ced}}{\partial u_{e_x}}, \text{ and } \frac{\partial E^{\text{bend}}_{ced}}{\partial u_{e_y}}. \tag{5.28}$$

Adding these contributions to (5.25) gives

$$H = \begin{bmatrix} \frac{\kappa\sqrt{2}}{2} + \frac{3\sqrt{2}\mu}{4} & \frac{\kappa\sqrt{2}}{2} - \frac{\sqrt{2}\mu}{4} \\ \frac{\kappa\sqrt{2}}{2} - \frac{\sqrt{2}\mu}{4} & \frac{\kappa\sqrt{2}}{2} + \frac{3\sqrt{2}\mu}{4} \end{bmatrix}, \quad \mathbf{B} = \frac{\gamma}{\sqrt{2}}\begin{bmatrix} \kappa+\mu \\ \kappa-\mu \end{bmatrix}, \tag{5.29}$$

which can once again be solved algebraically to give

$$\mathbf{U} = \frac{\gamma}{2}\begin{bmatrix} \frac{4\kappa+\mu}{2\kappa+\mu} \\ -\frac{\mu}{2\kappa+\mu} \\ 0 \end{bmatrix}. \tag{5.30}$$

Thus for a system with stretching and bending contributions, the solution vector implicitly depends on the networks fibre radii, r, as well as the Youngs modulus, E^f, through the dependence on κ and μ. Combining the solved x and y displacements of node n_e with the known displacements of nodes n_b, n_c, n_d, we can substitute into (3.34) as before for the total stretching energy calculation, but we can now also substitute into (3.13) to calculate the total bending energy of the network. Our numerical model agrees with the displacement vector prediction, $\mathbf{u}_e$, up to the numerical precision of PETSc's default `VecView` output, and agrees with the

energy predictions up to a numerical precision of 16 decimal places [4, p. 215].

Discussion

As $\kappa \to 0$ we see $u_{e_x} \to \frac{\gamma}{2}$ and $u_{e_y} \to -\frac{\gamma}{2}$ which is what we have previously calculated from the stretching only case. As κ is increased, the energy becomes dominated by bending. This means that the fibres in the network will stretch more readily at the expense of trying to minimise bending. In this case, we expect f_{ab} to only stretch, since it has a zero contribution to the bending. Furthermore, f_{cd} resists bending, and the result is an affine displacement in the horizontal direction. This is supported by the above calculations, which show $u_{e_x} \to \gamma$ and $u_{e_y} \to 0$ as $\kappa \to \infty$.

5.2.3 Effect of Segment Length on Condition Number

With the small modification of adding a single intersecting fibre to the λ-network, we can confirm the expectation that short segments increase the condition number of the matrix (see Figure 5.4). To collect the data for this experiment, the fibre intersecting the λ-network was

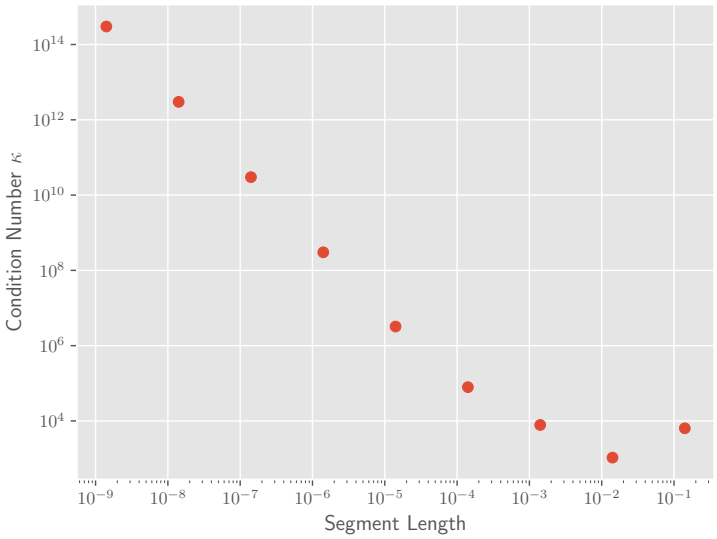


Figure 5.4: A plot to show the relationship between the shortest segment length in a simple lambda network variation and the condition number of the global Hessian assembled from the respective network.

oriented parallel to the x-axis and varied in the y-direction to create variations in the segment lengths (see Figure 5.5). The condition number was then calculated through singular value decomposition using PETSc.

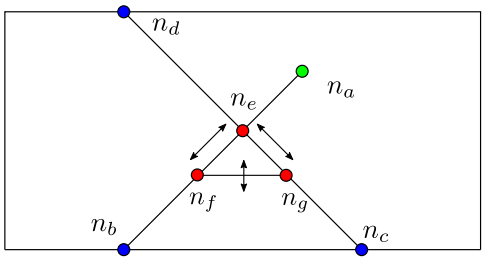


Figure 5.5: The λ-network with an intersecting fibre parallel to the x-axis. The distance between the intersecting fibre and the central node, n_e, is varied.

5.2.4 Node Merging Validation

In addition to using the lambda case network to validate expectations of how the model should behave, it can also be used to confirm the validity of the node merging procedure by demonstrating the insensitivity of the final result (if the tolerance is small). To do this, an additional fibre can be added to the lambda base case network, as seen in Figure 5.6. By modelling the

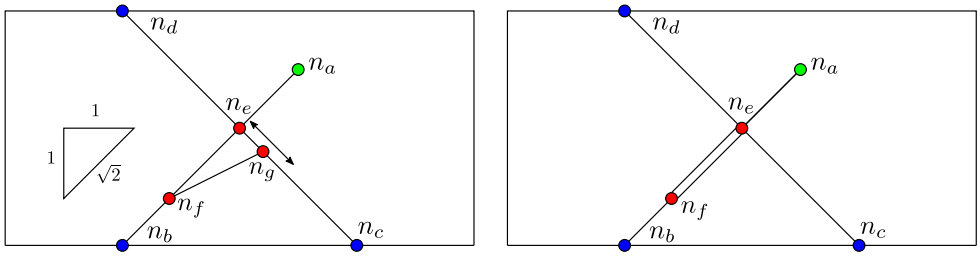


Figure 5.6: A basic λ-network with a fibre added before *(left)* and after *(right)* node merging has taken place.

mechanical behaviour of this network by varying the distance between nodes n_e, n_f and n_g while varying the node merging tolerance, we can obtain a clearer picture of the potential side effects of node merging.

The benchmark case, Figure 5.6 *(left)*, differs from the standard λ-network seen in Figure 5.3, in that there is an additional strut connecting segments n_e, n_b and n_c, n_e. This additional segment allows the network to bend along the triplets n_e, n_f, n_b and n_c, n_g, n_e, the extent of which depends on its interaction with the stretching and compressive forces of nearby segments and the bending over node triplet n_g, n_e, n_d.

In an ideal scenario, small variations in the spacing of nodes would lead to merging of all nearby nodes within a given cluster. In this case, node merging would still result in updated coordinates of merged nodes on the length scale of ν_m, but there would be no overlapping segment lengths. In a worst case scenario, large variations in node spacing would lead to node

merges that result in unphysical overlapping of segments on a length scale greater than ν_m. We expect the reality of node merging to lie somewhere between these best and worst case scenarios, so the purpose of this validation is to quantify the error that results from an idealised node merging, and assess the impact of undesirable segment overlapping on network property modelling.

We first discuss the effects of merging both nodes, n_f, n_g, of the new intersecting fibre f_{fg}, which will be achieved by selecting a node merging tolerance, ν_m, such that

$$\ell_{ge} \leq \ell_{ef} < \nu_m. \tag{5.31}$$

We will then discuss the effects of merging only node n_g with central node n_e, through a selection of ν_m such that

$$\ell_{ge} < \nu_m < \ell_{ef}. \tag{5.32}$$

To set up the experiment, the spacing between the internal nodes was chosen to exaggerate the effects of merging only one node. To do this, node coordinates were varied to give an order of magnitude difference between the relevant node distances, i.e.

$$10 \times \ell_{ge} = \ell_{ef}, \tag{5.33}$$

where ν_m respects either (5.31) or (5.31) depending on how many nodes should be merged. All other nodes were fixed, taking the same coordinates as in the previous λ-network, and all fibres were given a radius of $r = 10^{-3}$.

Best Case: Two Node Merge

Best case merging results in a network that is in fact identical to the λ-network. In this case the only common internal node between the merged network and the base network is n_e. Applying a linear shear to the upper boundary of the networks and solving, results in a matching displacement calculation, $\mathbf{u}_e$, for all tested variations of ν_m , accurate to the precision of PETSc's default `VecView` output (where ν_m is tested via variations in ℓ_{ge}, ℓ_{ef}) [4, p. 215].

Table 5.1 shows the relative error comparing the merged case with the benchmark of selected mechanical properties for the range of node merging tolerances tested. From this we see the network properties have the highest relative errors when segment lengths are larger - leading to bigger differences in network structure. Intuitively, we expect to see a larger error in the total bending energy since the node merging prevents any bending along segments ℓ_{ef} and ℓ_{ge}.

Worst Case: Single Node Merge

In the worst case, node merging results in network like Figure 5.6 *(right)*. In this case, the merging of node n_g with node n_e results in an unphysical segment overlap (i.e. segment n_e, n_f overlaps with the new position of segment n_g, n_f). Comparing the merged network with the

merge tol. (ν_m)	E_s	E_b	E	G
$\sqrt{2}$e-1 $< \nu_m$	3.490e-13	7.674e-03	1.934e-09	1.934e-09
$\sqrt{2}$e-2 $< \nu_m$	2.855e-12	3.741e-03	9.416e-10	9.416e-10
$\sqrt{2}$e-3 $< \nu_m$	2.628e-11	3.154e-03	8.173e-10	8.173e-10
$\sqrt{2}$e-4 $< \nu_m$	1.569e-10	1.868e-03	6.248e-10	6.248e-10
$\sqrt{2}$e-5 $< \nu_m$	1.473e-10	1.757e-04	1.912e-10	1.912e-10
$\sqrt{2}$e-6 $< \nu_m$	2.339e-11	2.829e-06	2.409e-11	2.409e-11

Table 5.1: Relative error for two node merge case. The merge tolerance decrements were chosen to coincide with the change in distance between nodes n_f, n_g and central node n_e. The positions of n_f, n_g were updated in logarithmic units [see Figure 5.6 *(left)*]

unmerged network Figure 5.6 *(left)*, the internal nodes in common are n_e and n_f. After the application of a linear shear to the upper boundary and solving, we find that

$$\mathbf{u}_e^{\text{unmerged}} = \mathbf{u}_e^{\text{merged}} \text{ for all } \nu_m,$$
$$\mathbf{u}_f^{\text{unmerged}} = \mathbf{u}_f^{\text{merged}} \text{ for } \nu_m \leq 10^{-5},$$

accurate to the precision of PETSc's default `VecView` output [4, p. 215]. The relative error of the $\mathbf{u}_f$ calculation for choices of $\nu_m > 10^{-5}$ are given in Table 5.2 along with the relative errors of selected mechanical properties for the full range of node merging tolerances tested. Much

merge tol. (ν_m)	$\mathbf{u}_f$	E_s	E_b	E	G
$\sqrt{2}$e-2 $< \nu_m <$ $\sqrt{2}$e-1	5.881e-03	3.553e-13	7.674e-03	1.934e-09	1.934e-09
$\sqrt{2}$e-3 $< \nu_m <$ $\sqrt{2}$e-2	6.570e-04	2.856e-12	3.741e-03	9.416e-10	9.416e-10
$\sqrt{2}$e-4 $< \nu_m <$ $\sqrt{2}$e-3	6.407e-05	2.628e-11	3.154e-03	8.173e-10	8.173e-10
$\sqrt{2}$e-5 $< \nu_m <$ $\sqrt{2}$e-4	5.001e-06	1.569e-10	1.868e-03	6.248e-10	6.248e-10
$\sqrt{2}$e-6 $< \nu_m <$ $\sqrt{2}$e-5	-	1.473e-10	1.757e-04	1.912e-10	1.912e-10
$\sqrt{2}$e-7 $< \nu_m <$ $\sqrt{2}$e-6	-	2.339e-11	2.830e-06	2.409e-11	2.409e-11

Table 5.2: Relative error for single node merge case. The merge tolerance decrements were chosen to coincide with the change in distance between nodes n_f, n_g and central node n_e. The positions of n_f, n_g were updated in logarithmic units [see Figure 5.6 *(left)*]

like the previous results in Table 5.1, the property with the worst relative error is the bending energy calculation, and the relative errors generally show a consistent decrease with decreasing value of node merging tolerance. Tables 5.1 and 5.2 give estimates for the size of error to be expected when a high proportion of the internal nodes are merged together. By calculating the relative error of the worst case from the best case (see Table 5.3) we can also see an estimate of the order of error we can expect from worst case node merging instead of cleaner best case node merging. These errors are several orders of magnitude smaller than the errors representing the difference between an unmerged and merged system.

The largest error across both worst and best case is no more than 1% over the ranges considered, and although we observe that the bending energy predictions have larger errors

ℓ_{gh}	E_s	E_b	E	G
$\sqrt{2}$e-1	6.284e-15	5.000e-08	6.400e-15	6.673e-15
$\sqrt{2}$e-2	6.982e-16	5.000e-09	3.491e-16	3.813e-16
$\sqrt{2}$e-3	0	5.000e-10	1.164e-16	3.813e-16
$\sqrt{2}$e-4	2.327e-16	4.979e-11	1.164e-16	3.813e-16
$\sqrt{2}$e-5	2.327e-16	1.079e-11	1.164e-16	3.813e-16
$\sqrt{2}$e-6	3.491e-16	2.535e-10	1.164e-16	3.813e-16

Table 5.3: Relative error of the single and two node merge cases compared. The merge tolerance decrements were chosen to coincide with the change in distance between nodes n_f, n_g and central node n_e. The positions of n_f, n_g were updated in logarithmic units [see Figure 5.6 *(left)*]

relative to the other property predictions in Tables 5.1 and 5.2, we note that in this toy network we have a high percentage of the internal nodes merged (50% of nodes merged in the worst case and 100% of nodes merged in the best case). In practice, in the networks we will model later in this chapter and in the next, we expect a much lower percentage of the total internal nodes to be merged in the case of node merging being included.

5.3 Lattice Based Networks in 2D

Lattice based networks are well understood in two dimensions [11, 63, 88], but typically do not offer characteristics that are representative of the real-world materials that motivated this project. However, the fact that exact analytical solutions exist for these lattice based networks makes them very attractive for model validation. In particular, we focus discussion on triangular lattice based networks. We begin by presenting a simple test that was validated against a known result for regular triangular meshes. We then conclude the section with an overview of work undertaken on the disordered, bond-diluted triangular meshes. Prior to the validation, we first introduce mesh generation routines for each of these test cases.

5.3.1 Regular Triangular Lattices

A regular triangular mesh can be depicted as a tessellation of equilateral triangles each with sides of length ℓ_t as seen in Figure 5.7. There are multiple approaches for generating such meshes, but here we will outline just two which are relevant to this project. The first is that which was implemented early on, independent of the main C++ generator, as a python script. The second is that which builds upon what has already been discussed in Section 5.1, and can be implemented and integrated with the existing C++ network generator.

Node Driven Mesh Generation

A node driven approach to mesh generation considers each of the lattice points that will form fibre intersections from the outset. In this way an algorithm for identifying fibre intersections

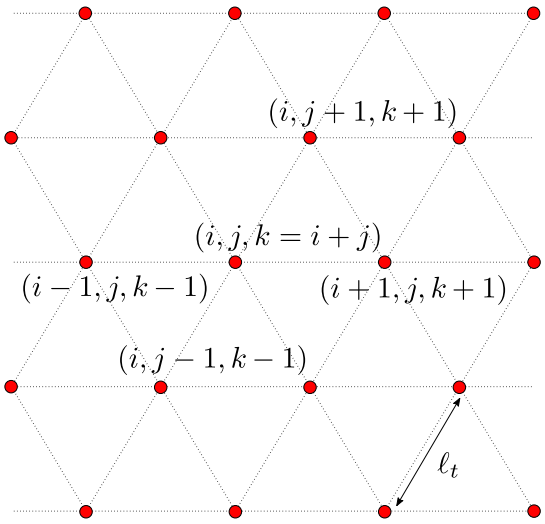


Figure 5.7: An example of a regular undiluted triangular mesh with sides of length ℓ_t. The row index is given by i and the column indices are given by j and k.

becomes obsolete, and the task diverts to deriving a sensible indexing scheme such that the associated nodes of each fibre can be easily identified. An outline of the algorithm for a given number of rows and columns is as follows:

1. For each row, i (see Figure 5.7)
   (a) Place node on lattice coordinates (at every multiple of ℓ_t)
2. For each row, i
   (a) Create a fibre consisting of the nodes on that row
3. For each column, j, with a north-east orientation
   (a) Create a fibre consisting of the nodes on that column
4. For each column, k, with a north-west orientation
   (a) Create a fibre consisting of the nodes on that reverse column

Fibre Driven Mesh Generation

Node driven triangular mesh generation was an attempt to remove some of the complexities of modelling lattice based networks using the Mikado generation algorithm discussed previously. It has the advantage of avoiding node duplication and offers an intuitive approach where, since nodes have already been created, there is no need for a separate connection routine. However, this comes at the cost of additional complexities when it comes to diluting segments between nodes, due to the need for a clear-cut indexing scheme that allows fibres to be correctly associated with existing nodes.

Alternatively, a fibre driven approach to triangular mesh generation follows the same methodology of the network generation approach introduced in Section 5.1.1. Whereas in the general algorithm we have previously stated (**step 2**) *"Generate a pair of coordinates that will form the end points of a fibre"*, we now choose these coordinate pairs to be fixed points with equal spacing at the $x, y = 0$, $x = W$ and $y = H$ boundaries of the domain. This can be split into three loops, one for fibres oriented parallel to the x direction and two for diagonal oriented fibres. Once all fibres have been generated, all that is left to do is to pass the network through the standard 2D cross-linking routine to determine all the points (or nodes) of intersection, taking care not to duplicate nodes.

Model Validation

It is known from the theory of linear elasticity that for an isotropic body, the shear modulus in two dimensions is given as

$$G = \frac{2E}{A\gamma^2}, \tag{5.34}$$

for energy, E, shear strain, γ, and area of the network, A [51]. Using (5.34), we can derive the shear modulus of a periodic regular triangular mesh to be

$$G_{\text{aff}} = k_t \frac{\sqrt{3}}{4}, \tag{5.35}$$

where k_t is the stretching constant for each of the segments of length ℓ_t. Here we are assuming that a macroscopic affine displacement at the boundaries can be translated in a uniform manner to all internal nodes at the microscopic level. We refer to this as the *affine assumption*, as introduced in Section 2.2.5 and provide the details of the derivation in Appendix B.1.

This result suggests that a regular 2D triangular network with *aperiodic* boundaries (see Figure 5.8) should approach a similar value of shear modulus G as the aspect ratio

$$a = \frac{L}{H} = \frac{\text{\# of nodes per row}}{\text{\# of rows of fibres}}, \tag{5.36}$$

increases. Stated explicitly, for a regular triangular mesh with fixed segment length between two adjacent nodes along a fibre, ℓ_t, fixed number of horizontally oriented fibres (or rows), H, and varying nodes per row, L, we expect $G \to k_t \frac{\sqrt{3}}{4}$ as $a \to \infty$.

To verify this expectation, the segment length was fixed at $\ell_t = 1$ and the number of rows was fixed at $H = 3$. The nodes per row, L, was varied such that $1 \leq a \leq 14$, and then the shear modulus was calculated using (5.34). The results can be seen in Figure 5.9, where the red line seen in Figure 5.9 *(left)* represents the expected value of G_{aff} for *periodic* boundaries.

From Figure 5.9 *(left)* there is evidence of asymptotic convergence to the true value of the shear modulus for $H = 3$. However from this figure in isolation, it is not clear whether this an indication of extremely slow convergence to $\frac{\sqrt{3}}{4}$ or whether this is an indication of convergence

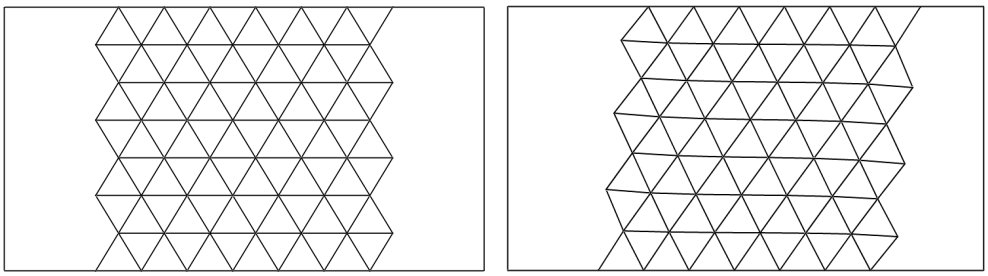


Figure 5.8: An example of a triangular mesh with $L = 7$, $H = 8$, and $\ell_t = 1$ before *(left)* and after *(right)* a small shear is applied.

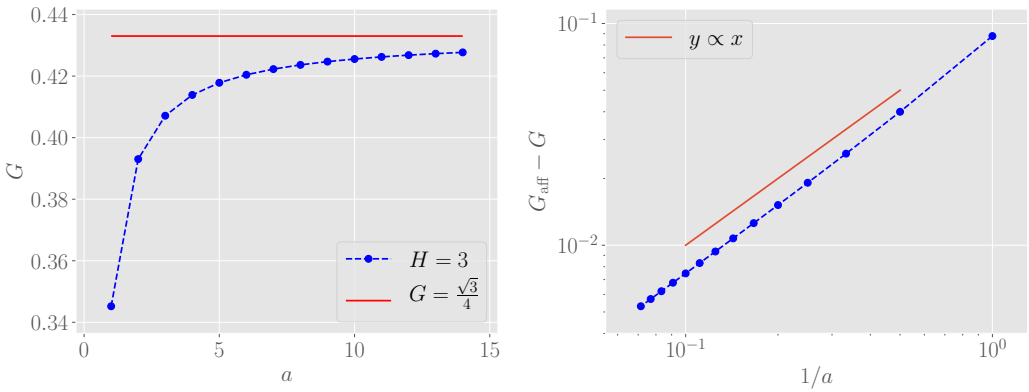


Figure 5.9: A plot to show shear modulus against aspect ratio for $H = 3$ *(left)*, and a log-log plot to show $G_{\text{aff}} - G$, against the aspect ratio for $H = 3$ *(right)*.

to a different value. Looking at Figure 5.9 *(right)*, we can see that the deviation is proportional to $1/a$, and thus the shear modulus is converging asymptotically to $\frac{\sqrt{3}}{4}$, and can be expressed as the power law $G_{\text{aff}} - G \propto \frac{1}{a^p}$, where a line fit gives $p = 1.089563$ with an error of 7.24867×10^{-3}. For $G_{\text{aff}} - G$ to be small it follows that a^p should be large, which implies that a large aspect ratio, a, is needed before we can expect to see the model producing results for regular aperiodic networks that are representative of regular periodic networks.

In addition to testing the effects of aspect ratio on the system convergence for aperiodic systems, we can also test the effects of overall system size (i.e. total nodes) on convergence for periodic systems. In this case, we hypothesise that for sufficiently large enough aspect ratio, increasing the system size will give a more accurate estimate of the behaviour of periodic systems. That is, for a regular triangular mesh with fixed segment length, ℓ_t, and fixed aspect ratio, L/H, the deviation should tend to 0 as the system size becomes infinitely large.

To test this hypothesis, the segment length was fixed at $\ell_t = 1$ and the aspect ratio was fixed at $L/H = 4$, 8, 12, 16, and 20. The system size, LH, was varied for fixed aspect ratio, and then G was calculated and used to find the deviation from G_{aff}. From Figure 5.10, we observe convergence to a finite value for increasing system size. The fact that the system quickly converges to a nonzero deviation value suggests that system size is not as important as aspect ratio in approximating the behaviour of periodic systems.

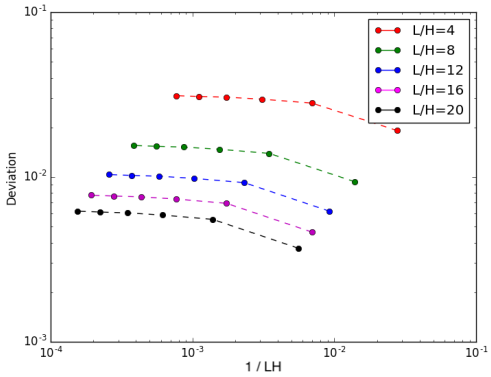


Figure 5.10: A log-log plot to show deviation $G_{\text{aff}} - G$, against system size, LH, (i.e. total nodes), for a range of fixed aspect ratios, $a = L/H$.

For the results outlined here the tested system size was at most 5760 internal nodes, due to the memory limitations of the NumPy direct solver (`linalg.solve` based on standard LU-decomposition) used in early Python prototypes of the model.

The results of these experiments provided a sound motivation for introducing periodicity into the model, however prior to the discussion of the experiments involving periodic regular and diluted triangular meshes, we will first briefly discuss the performance of simple block preconditioners with aperiodic networks .

Preconditioner Performance Analysis for Undiluted Meshes

Recall from the block preconditioners introduced in Section 4.2.2, one simple choice was

$$P_{Db} = \begin{bmatrix} H_{xx} & O \\ O & H_{yy} \end{bmatrix}.$$

A first experiment looked at how this preconditioner performed as the network size was increased. For this experiment, a regular aperiodic undiluted triangular mesh was used, with the model accounting for stretching behaviour and some small bending contributions (parameter choices are given in Table 5.4). With the fibre property parameters fixed, a nominal shear strain was applied to the upper boundary of the network and the number of iterations required for the block preconditioner to reach a tolerance of 10^{-10} was measured for an increasing mesh size. This iteration count was also recorded for our matrix H after diagonal scaling (where we apply the standard Jacobi preconditioner, $P_J = \text{diag}(H)$), which provided us with a benchmark for the block preconditioner.

Mesh Size	Unknowns	Iterations for P_J	Iterations for P_{D_b}
6x6	48	37	18
12x12	240	79	22
24x24	1056	147	22
48x48	4416	293	21

Table 5.4: Preconditioned Conjugate Gradient (PCG) results for different aperiodic undiluted mesh sizes. Parameters are $k_t = 1$; $\ell_t = 1$; $\kappa = 2.5 \times 10^{-5}$; $r = 0.01$; $E^f = 3183.1$; $\gamma = 0.1$.

Referring to Table 5.4, we observe a reasonable performance for the block preconditioner. Comparing the iteration count with that obtained through diagonal scaling, we can see that the number of iterations taken for the diagonally scaled system to converge increases with the mesh size, whereas the iteration count of the block preconditioned system remains constant. Despite the block preconditioner being a good choice for regular undiluted meshes, we will observe in the section that follows that the same is not true for diluted meshes.

5.3.2 Diluted Triangular Lattices

To increase the disorder observed in a triangular lattice formed network, we can remove segments at random. The result of diluting a network in such a way is a variation in fibre lengths and therefore fibre rigidity in the network. As with the previous section, we will first outline the approach for generating the networks, which we will motivate by looking at periodicity, and then discuss some of the results we obtained using diluted triangular meshes that we were able to use to validate the model against existing literature.

Periodic Mesh Generation

To reduce the effects of finite size, we can introduce periodicity into our networks. However, for successful periodic implementation within the framework of the model, system spanning fibres that cross periodic boundaries should be avoided, or else we need a scheme that handles ambiguous fibre end points. Additionally, recall from the discussion in Section 5.1.2 that correctly visualising fibre segments requires the nearest periodic copy of a node pair [see Figure 5.1 (*iv-vi*)]. To identify the nearest periodic copy of a consecutive node pair, n_i, n_j along a fibre we impose the constraints

$$|s_{j_x} - s_{i_x}| < \frac{|x_{\max} - x_{\min}|}{2}, \tag{5.37}$$

$$|s_{j_y} - s_{i_y}| < \frac{|y_{\max} - y_{\min}|}{2}. \tag{5.38}$$

To maintain consistency with previous aperiodic work, and to allow us to maintain how we impose our macroscopic shear strain to the networks, we choose to introduce periodicity in the x direction only. In this way, we keep constraint (5.37) and drop (5.38). To handle system spanning fibres with ambiguous end points, each continuous horizontal fibre should be split at least once. The simplest approach for splitting each fibre is to use knowledge of existing nodes on the lattice. In this way, a split can be performed by picking two consecutive lattice points at random and removing the segment between them. Splitting a horizontal fibre exactly once results in one continuously wrapped fibre due to x-periodicity. Further splitting leads to multiple fibres per row, which requires one fibre to retain the original fibre ID and the others to be assigned a new ID. By randomising the initial consecutive lattice points that are selected for the fibre splitting, we can increase the overall disorder of the network.

Recall from Section 5.3.1 that $G = G_{\text{aff}} = k_t \frac{\sqrt{3}}{4}$ for a regular periodic triangular mesh, given the affine assumption. We hypothesised that we would approach this limit of G_{aff} for a mesh with a fixed height and an increasing width. We therefore expect that introducing x-periodicity brings us to the affine limit and produces G_{aff}, regardless of system size. This expectation was verified for a variety of x-periodic network sizes. For example, for a network with a 24×24 mesh size and spring constant $k_t = 3.141593 \times 10^{-4}$ the calculated value for the shear modulus is $G = 1.360350 \times 10^{-4}$, which is equal to the calculated affine shear modulus $G_{\text{aff}} = k_t \frac{\sqrt{3}}{4}$ as expected.

Diluted Mesh Generation

Diluted mesh generation can be seen as an extension to the fibre splitting that takes place to generate a periodic triangular mesh. Rather than looping over every periodic horizontal fibre and cutting a segment at random, we instead loop over every segment of the network and cut it with a probability that reflects the desired dilution of the network.

As diluted triangular meshes can result in much more randomised architectures, post-

processing is used to remove any badly defined areas of the network. Problematic features in need of treatment include:

- node types that need updating due to the loss of nearby segments,
- fibres or nodes that are entirely detached from the rest of the network,
- regions lacking rigidity in the absence of bending,
- floppy modes, with a zero eigenvalue.

While node type labelling and smaller disconnected node and fibre components are often simple to treat in a post-processing step, zero energy modes and areas of the network lacking rigidity are difficult to identify without first trying to solve the associated global system. In the case of solver failure, regenerating the network with a new randomisation of the dilution is often the most convenient and cost effective treatment.

Validation and Comparison with Literature

To validate the model predictions for the shear modulus of diluted x-periodic triangular meshes, our results were compared with the work of Broedersz et al.[11]. Specifically, we generated a series of x-periodic 24×24 regular triangular meshes with bond length $\ell_t = 1$ and radius r. Each bond was diluted with probability $1 - p$, for bond occupation probability p. The shear modulus was measured for varying p, as seen in Figure 5.11. Our results are in good agreement with [11, Figure 2(a)] for the range of values we considered, showing how the shear modulus decreases with increasing bond dilution, from undiluted to just below the the central force isostatic threshold (occurring at $p_{\text{cf}} = 0.651$). We observe a momentary increase in G for lower choices of r as p decreases from $p = 0.9$ to $p = 0.8$. This suggests that the smaller mesh sizes used here in comparison with the literature may be allowing for sharper variations in the network geometry for relatively small changes in dilution. Smoother curves are expected for larger meshes and/or results averaged over more network dilutions.

Preconditioner Performance Analysis for Diluted Meshes

Following the analysis performed with the undiluted aperiodic triangular mesh in Section 5.3.1, we can now explore the performance of the same preconditioner for *fixed* network size but varying dilution. For this experiment, a regular x-periodic diluted triangular mesh was used, with the model accounting for stretching-only behaviour, with additional parameter details given in Table 5.4. Here, after a nominal shear strain was applied to the upper boundary of the network, the iteration count required to reach a tolerance of 1×10^{-10} was measured for our block preconditioned system and benchmarked against diagonal scaling for a fixed mesh size of 24×24 with increasing dilution. The mesh was diluted as outlined above, by removing segments at random. In addition to iteration count, condition numbers were estimated and the shear modulus was calculated.

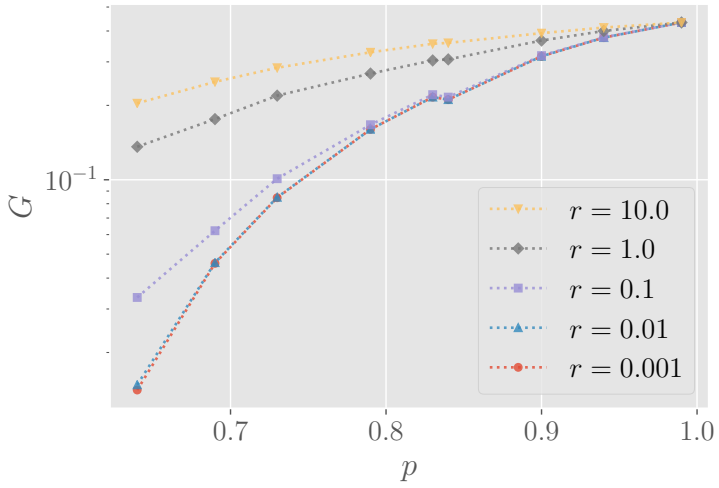


Figure 5.11: Validation of the shear modulus G, in units μ/ℓ_t, against the bond dilution probability $1 - p$ for 5 choices of radius, r.

Due to the possibility of generating a problematic mesh (see above), which increases with the level of dilution, this experiment averaged the results over 9 mesh generations per dilution. For $p = 0.84$, only 5/9 meshes were sufficiently well formed to solve, and for $p = 0.79$, only 4/9. In these two cases, the iterations and condition numbers were averaged over the solvable cases, and G was averaged over all 9 (with $G = 0$ for the unsolvable meshes).

Referring to Table 5.5, we see that although our block preconditioned system converges in less iterations than the diagonal scaling for every dilution, the iteration count is no longer constant. This indicates that the same simple block preconditioner that is well suited for

p	Iterations for P_J	Iterations for P_{D_b}	$\kappa(P_J^{-1}H)$	$\kappa(P_{D_b}^{-1}H)$	G
0.99	132.22	1.00	1225.51	75.33	0.4330
0.94	159.78	27.22	1649.82	649.68	0.3735
0.90	175.44	34.89	2109.67	994.91	0.3183
0.84	204.60	45.80	3304.84	2266.51	0.1294
0.79	309.75	90.25	9470132.12	62177700.22	0.0764

Table 5.5: PCG results for a diluted 24×24 x-periodic mesh averaged over 9 networks, collected using MATLAB. Parameters are $k_t = 1$; $\ell_t = 1$; $\kappa = 0$; $r = 0.01$; $E^f = 3183.1$; $\gamma = 0.1$.

undiluted triangular meshes may be less appropriate for networks with randomness introduced into their structures. The large condition number for $P_{D_b}^{-1}H$ at a dilution of $p = 0.79$ further supports this. The iteration count for the block conditioned system with a dilution of $p = 0.99$ appears erroneous, although at the time of writing the cause was not clear. There is little consistency between the 24×24 undiluted aperiodic mesh (with stretching and bending) (see Table 5.4) and the $p = 0.99$, 24×24 diluted x-periodic mesh (with stretching only). Moreover,

we would expect an iteration count of 1 to be accompanied with a condition number of 1.

The decision to remove bending from this experiment was made so that this experiment could also serve as a validation against the work of Broedersz et al.[11]. As discussed in Chapter 2, Broedersz et al. found a distinct threshold density $p_{\text{cf}} = 0.651$, for $\kappa = 0$, below which the network loses rigidity and the shear modulus vanishes to zero. Measuring the shear modulus of our network, we were able to reproduce the same decrease in G as we diluted the network.

The results from the experiments presented here and in Section 5.3.1 show that a good choice of preconditioner for a structured on-lattice network is not necessarily a good choice of preconditioner for an unstructured off-lattice network. However, it is also important to acknowledge that in the preconditioner analysis of the diluted triangular meshes, the networks were fixed in size. Thus, although the results suggest decreased performance for networks with less structure, we have not yet revealed how the iteration counts scale with the network size for a fixed bond occupation. This will be discussed in the context of fully random networks in Section 5.4.2.

5.4 Random Networks in 2D

Using two dimensional networks produced using the generation process discussed in Section 5.1.2, we now move on to validate the model against existing off-lattice work seen in the literature. Following this we will discuss the performance of systems resulting from 2D random fibre networks and present the novel results published in [36].

5.4.1 Validation and Comparison with Literature

A simple first validation of our model results was to verify that the calculated energy, E, and shear modulus, G, were bounded by the affine predictions E_{aff} and G_{aff} respectively. To do this, we increased the number of fibres, N_f, for fixed length, $\ell = 0.25$, and radius, $r = 0.01$, and plotted E and G alongside their corresponding affine predictions. For each interval of N_f, the results were averaged over 10 reproducibly-seeded randomly generated networks. Examples of individual network generations with varying N_f can be seen in Figure 5.16.

Figure 5.12 demonstrates that E *(left)* and G *(right)* lie below the limits expected by the affine predictions. We can also see that E increases linearly with N_f, as we would expect for sufficiently large N_f.

Next, as a step towards comparison with literature, we investigated the mechanical properties of the networks. In particular for $N_f = 100$, with $\ell = 0.25$, we varied r and measured the shear modulus, G, and ratio of bending energy over total energy, E_{bend}/E. From Figure 5.13 *(right)* we can see a smooth crossover from a regime with comparable stretching and bending energies for small r, to a stretching dominated regime for large r. Figure 5.13 *(left)* demonstrates a smooth transition between affine (i.e. uniform deformation field) dominated behaviour

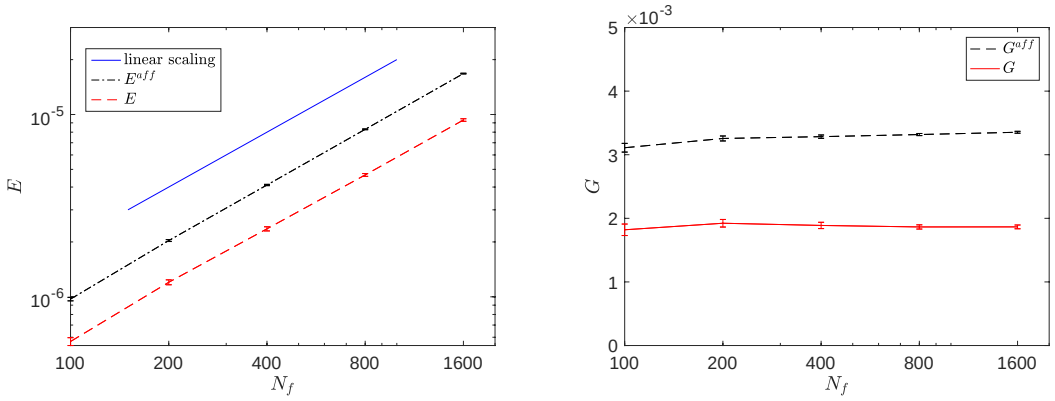


Figure 5.12: Calculated values of the energy *(left)* and shear modulus *(right)* verified against the affine predictions for an increasing number of fibres with fixed length, $\ell = 0.25$ and radius, $r = 0.01$. The solid line is added to verify linear scaling.

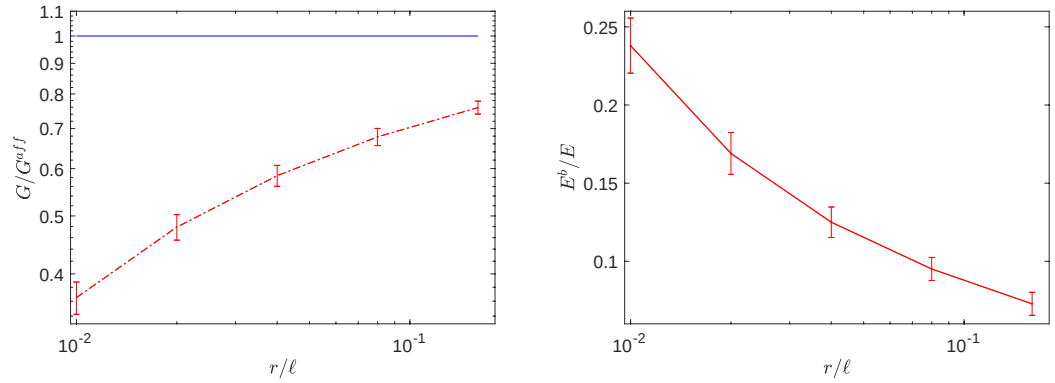


Figure 5.13: r/ℓ, for $\ell = 0.25$ and $N_f = 100$ in a 0.5×0.5 plane against G/G_{aff} *(left)* and against E_{bend}/E *(right)*. Nodes within a distance of 10^{-3} were merged into a single node prior to solution. The line $G/G_{\text{aff}} = 1$ *(left)* corresponding to affine response is also shown.

at large r to non-affine behaviour for small r. This coincides well with the known result that stretching-dominated networks become close to affine [3, 25, 26, 91].

To validate the 2D random fibre networks against the work of Head et al.[25, 26], a series of randomly seeded networks were produced with varying fibre stiffness, in this case characterised by the dimensionless cross link density, ℓ/ℓ_c, and ℓ_b/ℓ for $\ell_b = \sqrt{\kappa/\mu}$. Figure 5.14 *(left)* shows the relationship between shear modulus, G, and cross link density for 3 choices of the ratio ℓ_b/ℓ, compared with the affine shear modulus G_{aff}. In good agreement with the literature, we see

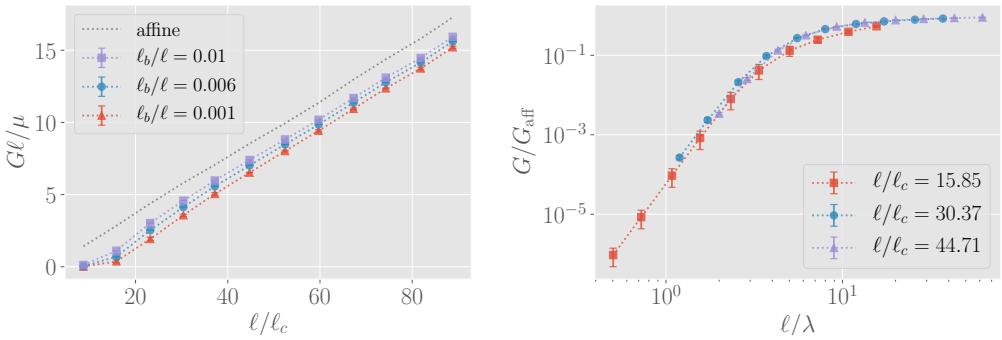


Figure 5.14: Validation of dimensionless shear modulus $G\ell/\mu$ against the dimensionless cross link density ℓ/ℓ_c for 3 choices of ℓ_b *(left)*, and of G/G_{aff} against ℓ/λ with $\lambda = \sqrt[3]{\ell_c^4/\ell_b}$ *(right)*. Both figures were averaged over 3 seeds. Error bars of *(left)* are no larger than the symbols.

that G vanishes for any choice of ℓ_b at the rigidity percolation transition $\ell/\ell_c \approx 5.9$. Visually, we may observe the effect of increasing cross link density for fixed $\ell_b/\ell = 0.006$, and a fixed box size, $(2\ell, 2\ell, 0)$, in Figure 5.15. Figure 5.14 *(right)* demonstrates the affinity of the network

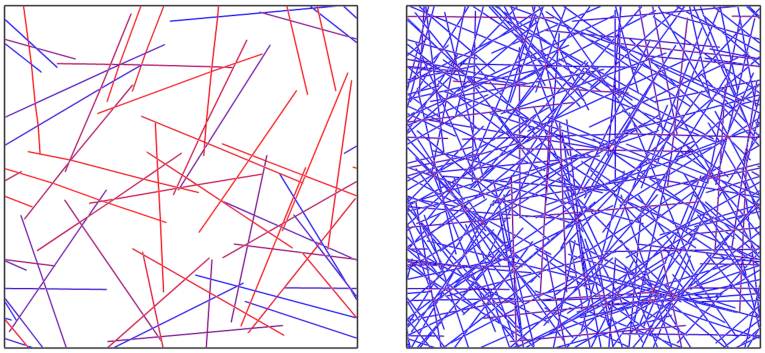


Figure 5.15: Example networks after an imposed shear of $\gamma = 0.01$ coloured by relative bending (red) and stretching (blue) energy with $\ell/\ell_c = 8.24$ *(left)* and $\ell/\ell_c = 45.06$ *(right)*.

deformation (expressed as the ratio of G/G_{aff}) as a function of the ratio of affine and nonaffine regions present in the network. This ratio is written as ℓ/λ, where $\lambda = \sqrt[3]{\ell_c^4/\ell_b}$. These results

match well with the predictions given by Head *et al.* [26, Figure 5, 8].

5.4.2 Performance Analysis for the Solution of 2D Networks

Beyond the validation and comparison of random 2D networks with literature, we also investigated the performance of various block preconditioners, of which the choices were introduced in Section 4.2.2 as (4.22) and (4.23)

$$P_{Db} = \begin{bmatrix} H_{xx} & O \\ O & H_{yy} \end{bmatrix}, \quad P_{Ab} = \begin{bmatrix} H_{xx} & H_{xy} \\ O & H_{yy} \end{bmatrix}, \quad P_{AS} = \begin{bmatrix} H_{xx} & H_{xy} \\ O & S \end{bmatrix}, \quad P_S = \begin{bmatrix} H_{xx} & O \\ O & S \end{bmatrix}.$$

Note that here we have also introduced a symmetric version of P_{AS} given as P_S. To evaluate the performance of the different choices of preconditioner, we measured the number of iterations required to a relative tolerance of 10^{-3}, using the MINRES and GMRES solvers in MATLAB. The rate of convergence was measured for increasing system size by increasing N_f, with fixed $\ell = 0.25$ and $r = 0.01$ (see Figure 5.16).

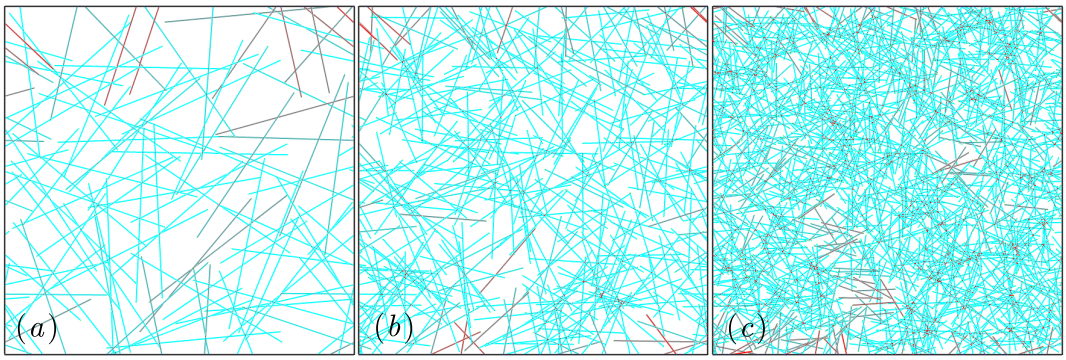


Figure 5.16: Examples of generated 2D random fibre networks increasing in area with a fixed fibre density. The number of fibres, N_f, is 100 (a), 400 (b), and 1600 (c), and fibre length, ℓ, and radius, r, are fixed at 0.25 and 0.01 respectively, but r is not to scale. Coloured by number of nodes on a fibre, where red represents a low number of nodes, and cyan represents a high number of nodes. Dangling ends have not been removed.

As can be seen from Table 5.6, P_{Db} shows a consistent performance as N_f increases, with good evidence that we would expect to see the iteration number remain constant if N_f were to increase further. P_S also performs well, and P_{Ab} consistently shows the lowest iteration count for increasing N_f, but the additional cost in building the P_S preconditioner, and the additional cost per iteration of GMRES versus MINRES leaves P_{Db} as potentially the preferred choice. The diagonally scaled preconditioner D performs the worst, with large fluctuations in the iteration count and a large standard error.

As a further indication of performance we also collected estimates for the condition numbers of the preconditioned system and for H. To do this we consider condition numbers of the

N_f	100	200	400	800
D	300.2 ± 54.0	480.8 ± 67.3	570.6 ± 77.3	410.9 ± 103.0
P_{Db}	8.0 ± 0.8	8.5 ± 0.5	7.7 ± 0.7	7.2 ± 0.6
P_S	12.3 ± 1.1	14.6 ± 1.2	13.8 ± 1.3	14.1 ± 0.9
P_{Ab}	5.3 ± 0.2	5.4 ± 0.2	5.4 ± 0.2	5.3 ± 0.2

Table 5.6: Number of iterations required to converge for different preconditioners and an increasing number of fibres, N_f, of length 0.25 and radius 0.01. Standard error was calculated from a sample of 10 networks.

preconditioned system and for H. To do this we consider a similar range of N_f as previously and provided the default seed to the `condest` function in MATLAB for 10 network generations at each interval. From the results in Table 5.7 we find additional evidence for the poor performance of D, with condition number estimates consistently higher than H without any preconditioning. P_S reduces the condition number of H by roughly an order of magnitude, and P_{Db} and P_{Ab}

N_f	100	200	400	800	1600
$\kappa(H)$	3.74e6	7.58e6	1.81e7	4.34e7	9.72e7
$\kappa(D^{-1}H)$	8.51e6	2.41e7	7.09e7	1.98e8	4.54e8
$\kappa(P_{Db}^{-1}H)$	7.32e4	1.47e5	4.85e5	1.60e6	5.12e6
$\kappa(P_S^{-1}H)$	1.39e5	4.97e5	1.18e6	4.22e6	-
$\kappa(P_{Ab}^{-1}H)$	6.98e4	1.96e5	5.99e5	2.11e6	-

Table 5.7: Estimated condition numbers, κ, of different preconditioners applied to H, for varied number of fibres, N_f.

demonstrate the best performance, showing similar estimates to each other with the symmetric having slightly lower values on average for larger N_f.

Our investigation of 3D fibre networks, which we will discuss in the chapter to follow, marked the move from the MATLAB/C hybrid model to a PETSc compatible full C implementation. In moving to PETSc, a sample of timings for the assembly of the global systems of various 2D network sizes [see Figure 5.17 *(left)*], shows that the unoptimised PETSc system assembly routine significantly outperforms our MATLAB hybrid model assembly. We also measured the solve time of three LU based methods available in PETSc, and compared the performance with a selection of solvers from MATLAB and the standard/sparse LU solvers of Python [see Figure 5.17 *(right)*]. The three solvers tested in PETSc were the default sparse LU method, SuperLU, and MUMPS (see Section 4.2.1). The MATLAB solvers that were tested were standard LU, MINRES preconditioned with P_{Db}, and the built-in direct method [invoked as `A\b` or `mldivide(A,b)`] which uses a sophisticated algorithm for deciding how to solve a matrix [59]. For 2D networks where MATLAB is able to store the resulting systems in memory, the solve time of MATLAB is competitive. In fact, referring to Figure 5.17 *(right)*, we see that while the direct LU based methods available in PETSc are able to compete with a range of

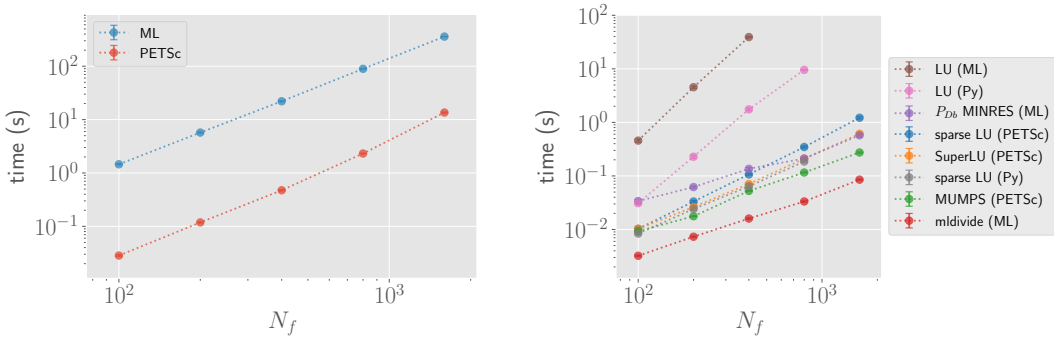


Figure 5.17: Time in seconds required to assemble *(left)* and solve *(right)* systems corresponding to random 2D networks of an increasing number of fibres, N_f, of length 0.25 and radius 0.01. Standard error was calculated from a sample of 9 networks.

Python and MATLAB based solvers, the built-in backslash method `A\b` of MATLAB shows the strongest performance. Nonetheless, the intention was to develop a 3D model, and our reason for testing the potential of PETSc at this point, in addition to the easy interchangeability of various solvers and opportunity for parallelism in future work, was the anticipation that in 3D the problem size would be too large for MATLAB.

Summary

A variety of topics have been covered in this chapter. Having previously discussed the numerical components of the model, we have now also covered the structural aspects of network modelling using Mikado networks as a framework. In the process of this, we have introduced some of the difficulties surrounding random network modelling, and evaluated approaches for handling these difficulties. In particular, we have studied the consequences of node merging, a technique that can be used to prevent nodes from being generated within close proximity. Aside from off-lattice networks, we have also considered lattice based networks, and the possible algorithms for generating such networks. Using both on and off lattice networks, we have validated our model against results given in literature, which gives further support to analytical calculations used to validate the model for the simple lambda case. Following validation of the model for the networks studied in this chapter, we have also given an original performance analysis of various simple block preconditioners. The strong performance of a diagonal block preconditioner coupled with the relative simplicity of this choice suggests it could be attractive for researchers seeking a faster solve time of random 2D networks. We concluded the chapter with an assembly time comparison and solve time analysis for various LU based direct methods available in PETSc, benchmarked against methods of Python and MATLAB. This allowed us to understand the potential of PETSc moving forward to 3D systems in the next chapter, and confirmed that PETSc is able to compete with existing methods implemented in Python and MATLAB.

Chapter 6

Modelling Networks in 3D

The previous chapter brought together the theory and methods of the early chapters to discuss our model in a complete form for 2D networks. This discussion included an overview of network representation, testing and validation, concluding with some experimental results evaluating the effectiveness of our model.

Here we will continue the discussion of modelling the collective behaviour of semiflexible polymers as random networks, now extending into 3D space. A prerequisite to this discussion is the introduction of some additional considerations required to represent networks in a 3rd dimension. With these considerations in mind, we will continue by discussing a selection of toy problems used to test 3D model functionality. This will lead into our first extension to 3D networks, which will form a major component of this chapter: layered fibrous networks.

Layered fibrous networks are motivated by nonwoven fabrics seen in industry, which consist of layered fibre network sheets that are cross-linked using a needle punching technology, or by other means of mechanical bonding (see Section 1.1). We begin with the modelling process, outlining the procedure for constructing layered networks. Cross-linking layered networks can be approached in a variety of ways, of which we compare a small selection that we considered. Following the overview of sheet network generation, we present the results of some tests used to verify model predictions for these networks. We then open an investigation into the effect of anisotropy on the mechanical properties of layered random fibre networks.

On concluding this discussion we move to the final part of the chapter, which concerns the mechanical properties of randomly oriented 3D fibre networks that are no longer confined to lie within layers. This section opens with a brief look at how these networks are modelled, followed by a discussion of preliminary results collected.

6.1 Network Representation in 3D

6.1.1 Cross-linking Fibres in 3D

Perhaps the most fundamental difference in network representation in two and three dimensions is that of fibre cross-links. Whereas nearby non-parallel fibres in a 2D plane will always have a unique point of intersection, working in 3D offers no such promise. In 3D, fibres must be manipulated in such a way that they are forced to connect. Many approaches in the literature exploit methods for creating direct intersections in 3D (see Section 2.2.2), for example by creating fibres on a lattice structure [11, 12], or by introducing weak attractive forces [40, 95]. Here we discuss two alternative approaches that leave randomly oriented fibres in their original undisplaced positions, and introduce connectivity either by addition of new short rod-like fibres, or by associating nearby fibres using a node coupling method. Both of these approaches are employed between points on fibre pairs that satisfy a minimum distance criteria. The algorithm for calculating this minimum distance between two fibres will be discussed in a dedicated section that follows the cross-linking approaches below.

A Cross-linking Fibre Approach

A calculation of the minimum distance between a fibre pair reveals a point on each respective fibre of closest distance. Assuming we already have this pair of points from respective nearby fibres, we wish to introduce a cross-linking between these points. In the cross-linking fibre approach, we insert a physical rod-like fibre between these two points. The end points of this new cross-linking fibre become nodes, n_i, n_j, that must also be associated with their respective parent fibres. The two cross-linking nodes are independent of each other, associated with separate indexes, i, j, in the global system (see Figure 6.1).

Note that in 2D, we defined cross-links as the internal nodes that exist at unique points of intersection, whereas here a cross-link is defined as two internal nodes n_i, n_j, and a *cross-linking fibre*, f_{ij} that connects them. As with 2D cross-links, the internal nodes are freely rotating, and the cross-linking fibre is spring-like. Since these fibres are free to stretch and compress, each cross-linking fibre contributes to the overall stretching energy of the system, but as they are not permitted to bend they do not contribute any bending energy. We can choose to make these cross-linking fibres relatively rigid so that this additional stretching energy is minimised.

A Node Coupled Approach

As with the cross-linking fibre approach (see above), we begin our discussion with the assumption that we have a pair of points from respective parent fibres of close proximity, and that we want to introduce a cross-linking between these points. In the node coupling approach, we treat each of the points as internal nodes and create a coupling constraint between them. Due

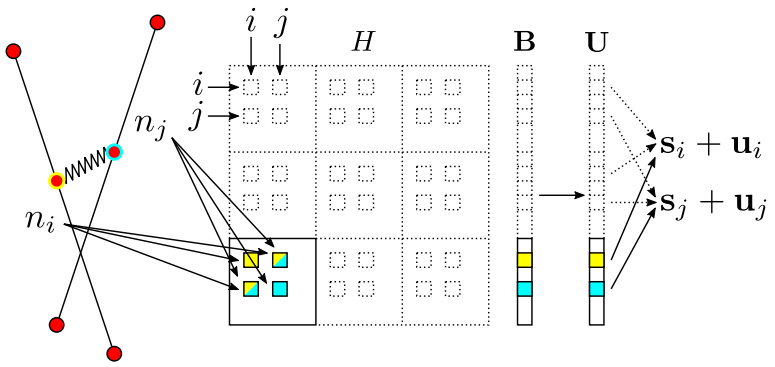


Figure 6.1: An example of two fibres cross-linked via an inserted cross-linking fibre *(left)*, with a schematic to show how the involved nodes contribute to the global system *(right)*. Individual global system indexes, represented by the small squares, are coloured such that they match with the colouring of the contributing internal node in the diagram, with split colouring representing a mixed contribution.

to the coupling, while each of the nodes have independent coordinates,

$$(x_a, y_a, z_a),$$
$$(x_b, y_b, z_b),$$

they share a global ID, i, and thus contribute to the same locations of the global Hessian and RHS vector. Thus a single displacement vector will be computed and applied to both nodes equally. This leads to a smaller global system, and therefore fewer unique displacements in the solution vector after solving. After solving, each displacement $\mathbf{u}_i$ must be applied to all nodes with the shared global ID of i as seen in Figure 6.2. We represent a coupling as

$$C_i = \{n_a, n_b\} \tag{6.1}$$

where a, b are the usual respective ID's of internal nodes n_a, n_b, and i represents the ID of the coupling as well as the shared global ID of both n_a and n_b. We may typically think of a couple as a grouping of two nodes but, as we will see in Section 6.1.2, we allow for node couplings with ≥ 2 nodes.

Mechanically, node coupled cross-links are stiff, permanent and not freely-rotating. Because the solution vector provides a single displacement for all the nodes in a coupling, the distance between nodes is constant and therefore does not contribute any stretching, compressive or bending energy to the overall system energy. An advantage of the node coupled approach is the smaller system size, but this comes at the cost of a coarser grained model of deformation at the microscopic level of the network.

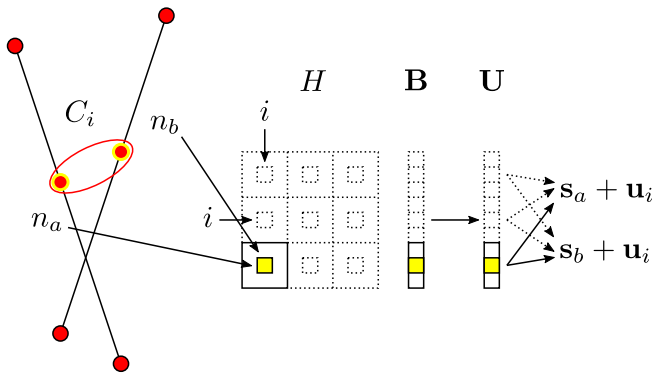


Figure 6.2: An example of two fibres cross-linked via a node coupling *(left)*, and a schematic demonstrating how the coupled nodes contribute to the global system *(right)*. Individual global system indexes, represented by the small squares, are coloured such that they match with the colouring of the coupled nodes in the diagram.

Minimum Distance Calculation

To find the minimum distance between two fibres in 3D, we first find the minimum distance between their respective unbounded lines. Here we will derive the minimum distance following the reasoning outlined by Eberly [19], but alternatives are equally suitable, such as the more geometric approach given by Sunday [81]. For the lines

$$\mathbf{r}(\lambda) = \mathbf{a} + \lambda\mathbf{b}, \tag{6.2}$$

$$\mathbf{s}(\mu) = \mathbf{c} + \mu\mathbf{d}, \tag{6.3}$$

where $-\infty < \lambda < \infty$ and $-\infty < \mu < \infty$ parameterise points along the lines and $\mathbf{b}$ and $\mathbf{d}$ are unit vectors, we can express the squared distance between the lines as a quadratic function with respect to λ and μ

$$Q(\lambda, \mu) = |\mathbf{r}(\lambda) - \mathbf{s}(\mu)|^2, \tag{6.4}$$

$$= \lambda^2 + 2\lambda b_r + 2\lambda\mu a_{rs} + 2\mu b_s + \mu^2 + c, \tag{6.5}$$

where $b_r = \mathbf{b} \cdot (\mathbf{a} - \mathbf{c})$, $a_{rs} = -\mathbf{b} \cdot \mathbf{d}$, $b_s = -\mathbf{d} \cdot (\mathbf{a} - \mathbf{c})$ and $c = |\mathbf{a} - \mathbf{c}|^2$.

To find the minimum squared distance we first calculate the gradient of the quadratic function

$$\nabla Q = \left(\frac{\partial Q}{\partial \lambda}, \frac{\partial Q}{\partial \mu} \right),$$

$$= (2\lambda + 2b_r + 2\mu a_{rs}, 2\lambda a_{rs} + 2b_s + 2\mu). \tag{6.6}$$

Then the minimum occurs when both of these partial derivatives are 0, thus we have a set of equations

$$2(\lambda + b_r + \mu a_{rs}) = 0, \tag{6.7}$$

$$2(\lambda a_{rs} + b_s + \mu) = 0, \tag{6.8}$$

which we can rewrite as the matrix system

$$\begin{bmatrix} 1 & a_{rs} \\ a_{rs} & 1 \end{bmatrix} \begin{bmatrix} \lambda \\ \mu \end{bmatrix} = - \begin{bmatrix} b_r \\ b_s \end{bmatrix}. \tag{6.9}$$

To solve for λ, μ, the matrix must be invertible which is true for $\delta = 1 - a_{rs}^2 > 0$, (since $\mathbf{b} \cdot \mathbf{d} \leq 1$) where δ is the determinant of the matrix. If $\delta = 0$ then $a_{rs} = -\mathbf{b} \cdot \mathbf{d} = 1$ and $\mathbf{r}, \mathbf{s}$ are parallel. In this case, there is no unique minimum distance, thus we can choose any arbitrary values for λ, μ. To avoid a near zero division resulting from a small δ, we test for $\delta < \nu_{3D}$, where ν_{3D} is sufficiently close to 0 (generally on the order of 10^{-8}) and then process any lines that are approximately parallel in the same manner. For $\delta > 0$, we can invert the matrix

$$\begin{bmatrix} \lambda \\ \mu \end{bmatrix} = \frac{1}{1 - a_{rs}^2} \begin{bmatrix} 1 & -a_{rs} \\ -a_{rs} & 1 \end{bmatrix} \begin{bmatrix} -b_r \\ -b_s \end{bmatrix}, \tag{6.10}$$

to find the values of λ, μ

$$\lambda = \frac{b_s a_{rs} - b_r}{1 - a_{rs}^2}, \quad \mu = \frac{b_r a_{rs} - b_s}{1 - a_{rs}^2}. \tag{6.11}$$

To find the minimum squared distance, these λ, μ values can be substituted into (6.6). By assigning a threshold value, $\xi_{\max}$, we can choose to cross-link any fibre pair with a minimum squared distance less than $\xi_{\max}$. When cross-linking via the cross-linking fibre approach, we introduce an additional constraint, $\xi_{\min}$, to prevent the generation of nearby nodes in high fibre density networks.

For fibres that satisfy the cross-linking criteria, we can substitute λ and μ into $\mathbf{r}$ and $\mathbf{s}$ to find the respective coordinates along each of the unbounded lines where the minimum distance occurs. In the case that $0 \leq \lambda \leq |f_{\mathbf{r}}|$ and $0 \leq \mu \leq |f_{\mathbf{s}}|$, the coordinates of each end of the minimum distance will lie within the bounds of the finite fibre segments $f_{\mathbf{r}}$ and $f_{\mathbf{s}}$ respectively, and the minimum distance calculation is complete. Cross-linking is then performed subject to this distance lying within the threshold values $\xi_{\min}$ and $\xi_{\max}$ (see Figure 6.3). Otherwise, at least one of λ, μ is out of range and the fibre pair can be left not cross-linked, or additional calculations should be performed. For example, if $\lambda > |f_{\mathbf{r}}|$ then we consider the furthest point of $f_{\mathbf{r}}$, which is the end point

$$f_{\mathbf{r}}^{\text{end}} = \mathbf{a} + \lambda \mathbf{b} \quad \text{where } \lambda = |f_{\mathbf{r}}|,$$

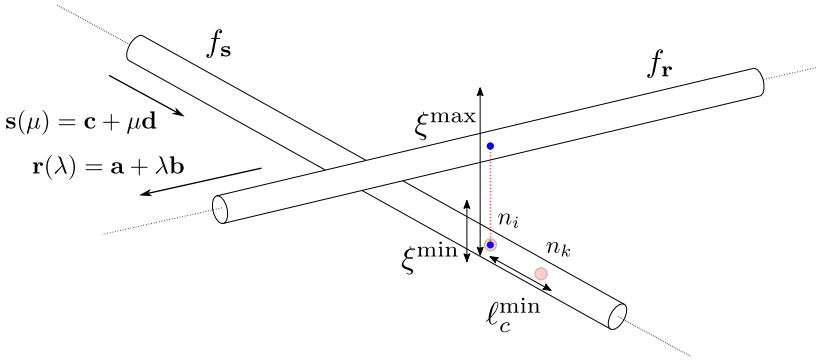


Figure 6.3: The constraints involved in the calculation of the minimum distance between the finite length fibres $f_{\mathbf{r}}$ and $f_{\mathbf{s}}$ which lie on $\mathbf{r}$ and $\mathbf{s}$ respectively.

and find the minimum squared distance between the point $f_{\mathbf{r}}^{\text{end}}$ and the fibre segment $f_{\mathbf{s}}$. This can be done by projecting $f_{\mathbf{r}}^{\text{end}}$ onto the unbounded line $\mathbf{s}$, and then finding the distance between $f_{\mathbf{r}}^{\text{end}}$ and the projection (see [19, p. 641]). The case of $\lambda < 0$ can be handled in similar manner, and the cases of $\mu < 0$ and $\mu > |f_{\mathbf{s}}|$ are accounted for in the additional point-to-segment minimum distance calculation. We omit a description of this algorithm as we typically choose not to cross-link fibres end-to-end, but for details see [19].

6.1.2 Node Nearness in 3D

Node Merging in 3D

In the extension to 3D, we still perform node merging as discussed in Section 5.1.5. Here node merging should occur immediately after calculating the minimum distance between bounded pairs of fibres. Each node introduced at the end points of this minimum distance should be checked for existing nearby nodes on their respective parent fibres (see Figure 6.3). Any new node, n_i, within the minimum cross-link distance, $\ell_c^{\min}$, of an existing node, n_k, should be merged to n_k.

Couple Merging

Much like node merging, we can choose to merge couples together. However it is important to note that the motivations for doing so are different. Node merging can be used to improve the condition of the network, by increasing the spacing between nodes in the network. Couple merging prevents index conflicts in the resulting global system of a network. Therefore node merging is beneficial, couple merging is necessary. This is because in the model implementation every internal node can only be associated with one global ID. If couple merging is not performed, then the global ID of nodes included in multiple couplings will be overwritten and only some of their respective contributions will be accounted for.

Couple merging brings the possibility of couple chains (see Figure 6.4), the likelihood of which depends on the tolerance for initial node merging. Long chains are undesirable because as discussed in Section 6.1.1, cross-links formed via node couplings are fixed, unable to stretch, compress or rotate. Thus long chains could restrict overall network deformation and reduce the applicability to real-world materials.

The algorithm for couple merging can be summarised as follows

1. Take a list of sets, where each set contains the IDs of coupled nodes.

   (a) If it is not the first pass and the number of coupled sets has not changed since the last pass, then end.

   (b) Else, compare pairs of sets i and j.

      i. If sets i and j have a common node ID then copy the contents of set j to set i, ignoring any duplicates. Then delete set j, and restart the loop.

      ii. Else, move to the next set $j + 1$

Note that the loop restart at step $1(b)(i)$ is necessary to prevent skipping past merges that may occur after copying node IDs from set j to set i. To demonstrate the algorithm, we can

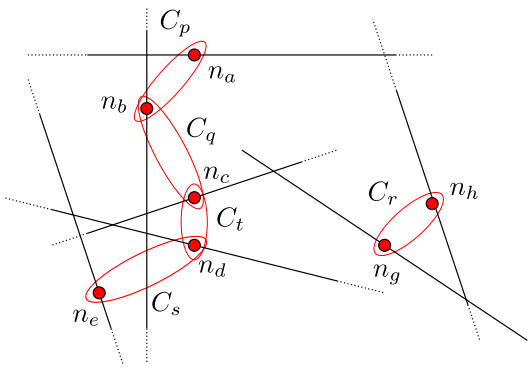


Figure 6.4: An example of a network with multiple node couplings that are chained together.

represent Figure 6.4 as the list of sets

$$[\{a, b\}_p, \{b, c\}_q, \{g, h\}_r, \{d, e\}_s, \{d, c\}_t], \tag{6.12}$$

where set $\{a, b\}_p$ represents the IDs of nodes n_a, n_b that belong to the coupling C_p. Applying the algorithm results in three partial passes and one full pass through the list before terminating.

The respective sets after each of these passes can be written as

$$\begin{aligned}
&\textbf{Pass 1:} \quad [\{a,b,c\}_p, \{g,h\}_r, \{d,e\}_s, \{d,c\}_t],\\
&\textbf{Pass 2:} \quad [\{a,b,c,d\}_p, \{g,h\}_r, \{d,e\}_s],\\
&\textbf{Pass 3:} \quad [\{a,b,c,d,e\}_p, \{g,h\}_r],\\
&\textbf{Pass 4:} \quad [\{a,b,c,d,e\}_p, \{g,h\}_r].
\end{aligned} \tag{6.13}$$

where the unchanged list of sets between **Pass 3** and **Pass 4** is the necessary condition for the algorithm to end.

6.2 Test Networks in 3D

6.2.1 Tripod Case Stretch & Bend Validation

Analogous to the validation undertaken with the λ-network in Section 5.2, we first validated the model extension to 3D with a simple tripod network. As can be seen from Figure 6.5, the tripod network consists of a three fibre intersection, with one fibre, f_{ab}, spanning the height of the domain and another two, f_{ce}, f_{de}, acting as legs from the central point. The nodes of the

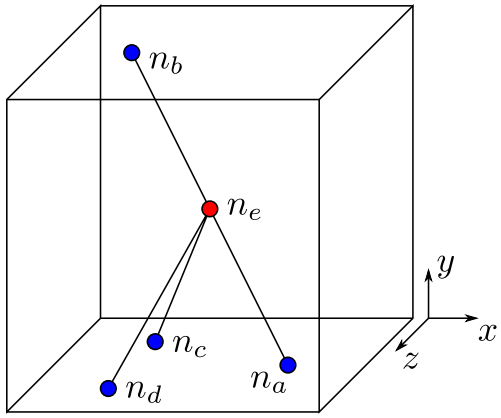


Figure 6.5: The tripod network. The network spanning fibre f_{ab} permits bending between segments $n_a n_e$ and $n_e n_b$, whereas the other fibres f_{ce} and f_{de} can only stretch and compress.

network are n_a, n_b, n_e, n_c, n_d, each described by coordinates given in their respective position vectors

$$\begin{aligned}
&\mathbf{s}_a = [0.75, 0.0, 0.5]^T, \quad \mathbf{s}_b = [0.25, 1.0, 0.5]^T, \quad \mathbf{s}_e = [0.5, 0.5, 0.5]^T,\\
&\mathbf{s}_c = [0.25, 0.0, 0.25]^T, \quad \mathbf{s}_d = [0.25, 0.0, 0.75]^T.
\end{aligned} \tag{6.14}$$

From the position vectors (6.14), the segment lengths can be given as

$$\ell_{ae} = \frac{\sqrt{5}}{4}, \quad \ell_{eb} = \frac{\sqrt{5}}{4}, \quad \ell_{ce} = \frac{\sqrt{6}}{4}, \quad \ell_{de} = \frac{\sqrt{6}}{4}. \tag{6.15}$$

For shear strain, γ, applied at the upper boundary $y = 1$ where $y = 0$ is fixed, we define the displacement vectors of boundary nodes n_a, n_b, n_c, n_d as

$$\begin{aligned} \mathbf{u}_a &= [\gamma s_{a_y}, 0.0, 0.0]^T, & \mathbf{u}_b &= [\gamma s_{b_y}, 0.0, 0.0]^T, & \mathbf{u}_c &= [\gamma s_{c_y}, 0.0, 0.0]^T, & \mathbf{u}_d &= [\gamma s_{d_y}, 0.0, 0.0]^T, \\ &= [0.0,\ 0.0, 0.0]^T, & &= [\gamma,\ 0.0, 0.0]^T, & &= [0.0,\ 0.0, 0.0]^T, & &= [0.0,\ 0.0, 0.0]^T. \end{aligned} \tag{6.16}$$

Node n_e lies in the centre of the domain, and thus $\mathbf{u}_e$ is the only unknown displacement vector.

Calculate Local Stretching Contributions

To calculate the local stretching contributions of each fibre segment, we can largely follow the same procedure given in Section 5.2, noting that we now have a 3rd dimension to our matrices and vectors. Thus segment n_a, n_e can be described by the local stretching system

$$H_{ae}^{\text{stretch}} = \mu\frac{4\sqrt{5}}{25}\begin{bmatrix} 1 & -1 & -2 & 2 & 0 & 0 \\ -1 & 1 & 2 & -2 & 0 & 0 \\ -2 & 2 & 4 & -4 & 0 & 0 \\ 2 & -2 & -4 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{B}_{ae}^{\text{stretch}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{6.17}$$

segment n_e, n_b by the local stretching system

$$H_{eb}^{\text{stretch}} = \mu\frac{4\sqrt{5}}{25}\begin{bmatrix} 1 & -1 & -2 & 2 & 0 & 0 \\ -1 & 1 & 2 & -2 & 0 & 0 \\ -2 & 2 & 4 & -4 & 0 & 0 \\ 2 & -2 & -4 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{B}_{eb}^{\text{stretch}} = \gamma\mu\frac{4\sqrt{5}}{25}\begin{bmatrix} 1 \\ -1 \\ -2 \\ 2 \\ 0 \\ 0 \end{bmatrix}, \tag{6.18}$$

segment n_c, n_e by the local stretching system

$$H_{ce}^{\text{stretch}} = \mu\frac{\sqrt{6}}{9}\begin{bmatrix} 1 & -1 & 2 & -2 & 1 & -1 \\ -1 & 1 & -2 & 2 & -1 & 1 \\ 2 & -2 & 4 & -4 & 2 & -2 \\ -2 & 2 & -4 & 4 & -2 & 2 \\ 1 & -1 & 2 & -2 & 1 & -1 \\ -1 & 1 & -2 & 2 & -1 & 1 \end{bmatrix}, \quad \mathbf{B}_{ce}^{\text{stretch}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{6.19}$$

and finally segment n_d, n_e by the local stretching system

$$H_{de}^{\text{stretch}} = \mu\frac{\sqrt{6}}{9}\begin{bmatrix} 1 & -1 & 2 & -2 & -1 & 1 \\ -1 & 1 & -2 & 2 & 1 & -1 \\ 2 & -2 & 4 & -4 & -2 & 2 \\ -2 & 2 & -4 & 4 & 2 & -2 \\ -1 & 1 & -2 & 2 & 1 & -1 \\ 1 & -1 & 2 & -2 & -1 & 1 \end{bmatrix}, \quad \mathbf{B}_{de}^{\text{stretch}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{6.20}$$

Calculate Local Bending Contributions

Looping over all consecutive segments, there is just one node triplet that we assemble local bending contributions for. It follows that the triplet n_a, n_e, n_b has a local bending system given by

$$H_{aeb}^{\text{bend}} = \kappa\frac{64\sqrt{5}}{125}\begin{bmatrix} 4 & -8 & 4 & 2 & -4 & 2 & 0 & 0 & 0 \\ -8 & 16 & -8 & -4 & 8 & -4 & 0 & 0 & 0 \\ 4 & -8 & 4 & 2 & -4 & 2 & 0 & 0 & 0 \\ 2 & -4 & 2 & 1 & -2 & 1 & 0 & 0 & 0 \\ -4 & 8 & -4 & -2 & 4 & -2 & 0 & 0 & 0 \\ 2 & -4 & 2 & 1 & -2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5 & -10 & 5 \\ 0 & 0 & 0 & 0 & 0 & 0 & -10 & 20 & -10 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5 & -10 & 5 \end{bmatrix}, \quad \mathbf{B}_{aeb}^{\text{bend}} = \gamma\kappa\frac{128\sqrt{5}}{125}\begin{bmatrix} -2 \\ 4 \\ -2 \\ -1 \\ 2 \\ -1 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{6.21}$$

Assembling the Global System

Assembling the global system gives

$$H = \begin{bmatrix} \frac{1024\sqrt{5}\kappa}{125} + \frac{2\sqrt{6}\mu}{9} + \frac{8\sqrt{5}\mu}{25} & \frac{512\sqrt{5}\kappa}{125} - \frac{16\sqrt{5}\mu}{25} + \frac{4\sqrt{6}\mu}{9} & 0 \\ \frac{512\sqrt{5}\kappa}{125} - \frac{16\sqrt{5}\mu}{25} + \frac{4\sqrt{6}\mu}{9} & \frac{256\sqrt{5}\kappa}{125} + \frac{8\sqrt{6}\mu}{9} + \frac{32\sqrt{5}\mu}{25} & 0 \\ 0 & 0 & \frac{256\sqrt{5}\kappa}{25} + \frac{2\sqrt{6}\mu}{9} \end{bmatrix}, \tag{6.22}$$

$$\mathbf{B} = \gamma \begin{bmatrix} \frac{512\sqrt{5}\kappa}{125} + \frac{4\sqrt{5}\mu}{25} \\ \frac{256\sqrt{5}\kappa}{125} - \frac{8\sqrt{5}\mu}{25} \\ 0 \end{bmatrix}, \tag{6.23}$$

which can be solved to give

$$\mathbf{U} = \gamma \begin{bmatrix} \frac{36864\sqrt{6}\kappa^2+110592\sqrt{5}\kappa^2+5520\sqrt{5}\kappa\mu+8280\sqrt{6}\kappa\mu+125\sqrt{5}\mu^2+150\sqrt{6}\mu^2}{4\left(13824\sqrt{6}\kappa^2+55296\sqrt{5}\kappa^2+2610\sqrt{5}\kappa\mu+5880\sqrt{6}\kappa\mu+125\sqrt{5}\mu^2+150\sqrt{6}\mu^2\right)} \\ -\frac{\sqrt{30}(48\kappa+5\mu)}{144\sqrt{30}\kappa+2880\kappa+40\sqrt{30}\mu} \\ 0 \end{bmatrix}. \tag{6.24}$$

As $\kappa \to 0$ we see $u_{e_x} \to \frac{\gamma}{4}$ and $u_{e_y} \to -\frac{\gamma}{8}$. In this case there is no energy cost associated with bending, and central node n_e displaces to minimise the total network (stretching) energy. In the $\kappa \to \infty$ limit, there is negligible energy cost associated with fibre stretching and compression, relative to the enormous energy cost of fibre bending. In this limit, all of the μ terms of (6.22) and (6.23) are dropped, leading to an underconstrained problem with no unique solution but rather a line of solutions. This line of solutions lies along the updated position of f_{ab} after node n_b has been displaced, and can be explained by an infinite bending energy cost, that allows fibres to freely stretch and compress. The result is that n_e can effectively "slide" along the length of f_{ab} with no additional energy cost, and any position along f_{ab} is equally suitable for minimising the total energy. Of this infinite number of solutions, there is exactly one which corresponds with the affine solution.

6.2.2 Planar Networks in 3D

Taking an existing 2D network and placing it into 3D can serve as a useful validation technique and acts as a stepping stone towards sheet based networks discussed in Section 6.3. Provided the network is well defined in 2D, and is fixed between opposing boundaries, then keeping the network fixed at the same boundaries when it is placed within a three-dimensional box, we can state the following assumptions;

- In the linear regime, shearing in the direction of the plane containing the 2D network will produce a total network energy that is equal (within numerical precision) to the total energy obtained by shearing the same 2D network in a 2D domain. In the nonlinear regime, the possibility of deformation in the z direction due to buckling could lower the total energy of 2D plane (in 3D) below the strictly 2D network [63].

- Imposing a linear shear in the direction perpendicular to the plane results in an affine deformation with a zero total network energy.

We can verify this first assumption analytically by comparing the assembled global system of the λ-network seen in equations (5.29) and (5.30), with the assembled global system of the

λ-network placed in the central z-plane of 3D box [see Figure 6.6 *(left)*]

$$H = \sqrt{2}\begin{bmatrix} \frac{\kappa}{2}+\frac{3\mu}{4} & \frac{\kappa}{2}-\frac{\mu}{4} & 0 \\ \frac{\kappa}{2}-\frac{\mu}{4} & \frac{\kappa}{2}+\frac{3\mu}{4} & 0 \\ 0 & 0 & \kappa \end{bmatrix}, \quad \mathbf{B} = \frac{\gamma}{\sqrt{2}}\begin{bmatrix} \kappa+\mu \\ \kappa-\mu \\ 0 \end{bmatrix}, \tag{6.25}$$

where solving gives

$$\mathbf{U} = \frac{\gamma}{2}\begin{bmatrix} \frac{4\kappa+\mu}{2\kappa+\mu} \\ -\frac{\mu}{2\kappa+\mu} \\ 0 \end{bmatrix}. \tag{6.26}$$

As can be seen, extending the λ-network to 3D results in a solution vector with the same x, y components and a z component of 0. Additionally, we observe that in the global system the xy interactions contained in the H_{xx}, H_{xy}, H_{yx}, and H_{yy} sub-blocks remain consistent with the 2D system. Furthermore, there are no mixed xz or yz interactions, seen by the zero valued $H_{xz}, H_{zx}, H_{yz}, H_{zx}$ sub-blocks, and the $H_{zz} = \sqrt{2}\kappa$ entry leads to $\sqrt{2}\kappa u_z = 0 \implies u_z = 0$.

The need to account for bending contributions when considering 2D networks in 3D space can be seen through the observation that the average coordination number will always be less than the 3D isostatic central force point ($z < z_{\text{cf}} = 6$), with the exception of lattice geometries. Furthermore, we can see from (6.25) that assembling the global system without bending contributions leads to a zero valued H_{zz} sub-block.

To verify our second assumption, we can assemble and solve the sytem representing the same λ-network sheared in z direction (or equivalently rotated and sheared in the x-direction - see Figure 6.6 *(right)*). In this case the Hessian remains the same, but the global RHS vector

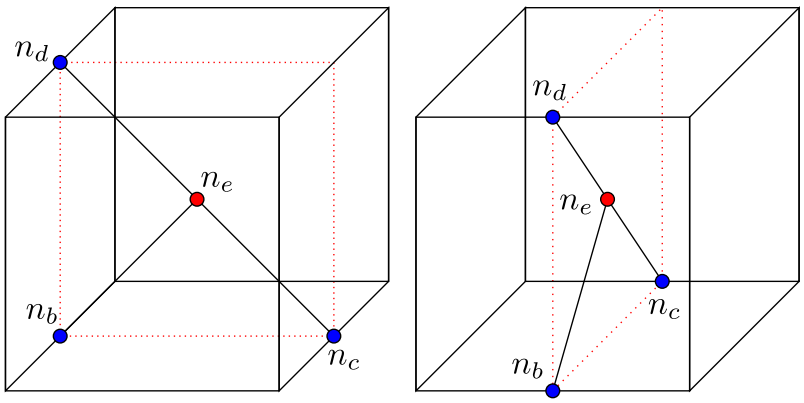


Figure 6.6: An extension of the λ-network (see Figure 5.3) into 3D. We can investigate anisotropy by varying the shear direction or by considering a rotation *(right)* of the original network *(left)*.

changes to reflect the new shear direction

$$H = \sqrt{2}\begin{bmatrix} \frac{\kappa}{2}+\frac{3\mu}{4} & \frac{\kappa}{2}-\frac{\mu}{4} & 0 \\ \frac{\kappa}{2}-\frac{\mu}{4} & \frac{\kappa}{2}+\frac{3\mu}{4} & 0 \\ 0 & 0 & \kappa \end{bmatrix}, \quad \mathbf{B} = \gamma\sqrt{2}\begin{bmatrix} 0 \\ 0 \\ \kappa \end{bmatrix}. \tag{6.27}$$

Solving (6.27) gives an affine deformation

$$\mathbf{U} = \begin{bmatrix} 0 \\ 0 \\ \gamma \end{bmatrix}, \tag{6.28}$$

which will therefore result in a zero total network energy under linear shear as expected.

6.2.3 Layered λ-networks

To better understand the interaction between cross-linked planar networks, we can investigate variations of layered λ-networks as seen in Figure 6.7. Beginning with identical λ-networks that

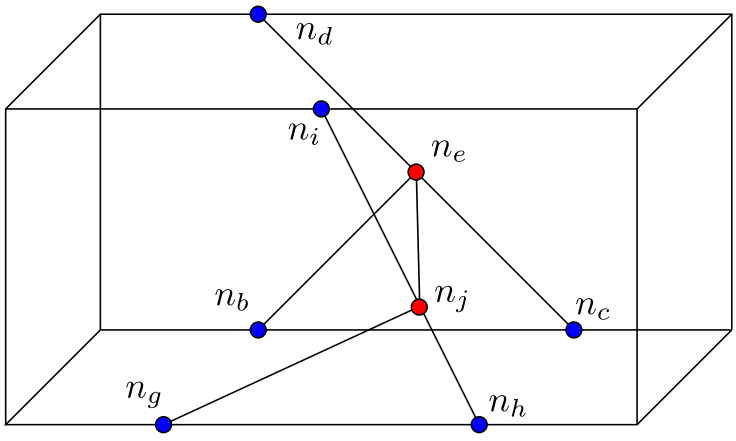


Figure 6.7: Layered λ-networks were used to validate early approaches to sheet network modelling.

lie perfectly parallel to each other, the assembled global system is given as

$$H = \begin{bmatrix} \frac{\sqrt{2}(2\kappa+3\mu)}{4} & 0 & \frac{\sqrt{2}(2\kappa-\mu)}{4} & 0 & 0 & 0 \\ 0 & \frac{\sqrt{2}(2\kappa+3\mu)}{4} & 0 & \frac{\sqrt{2}(2\kappa-\mu)}{4} & 0 & 0 \\ \frac{\sqrt{2}(2\kappa-\mu)}{4} & 0 & \frac{\sqrt{2}(2\kappa+3\mu)}{4} & 0 & 0 & 0 \\ 0 & \frac{\sqrt{2}(2\kappa-\mu)}{4} & 0 & \frac{\sqrt{2}(2\kappa+3\mu)}{4} & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2}\kappa+5\mu & -5\mu \\ 0 & 0 & 0 & 0 & -5\mu & \sqrt{2}\kappa+5\mu \end{bmatrix}, \tag{6.29}$$

$$\mathbf{B} = \frac{\gamma}{\sqrt{2}} \begin{bmatrix} \kappa+\mu \\ \kappa+\mu \\ \kappa-\mu \\ \kappa-\mu \\ 0 \\ 0 \end{bmatrix}, \tag{6.30}$$

which can be solved to give

$$\mathbf{U} = \frac{\gamma}{2} \begin{bmatrix} \frac{4\kappa+\mu}{2\kappa+\mu} \\ \frac{4\kappa+\mu}{2\kappa+\mu} \\ -\frac{\mu}{2\kappa+\mu} \\ -\frac{\mu}{2\kappa+\mu} \\ 0 \\ 0 \end{bmatrix}. \tag{6.31}$$

thus the resulting displacement is consistent with the single λ-network displacement seen previously in (6.26). Furthermore, we observe that the layers remain in their respective planes, perpendicular to the shear direction. From (6.29) we deduce that deformations in the x and y directions are completely decoupled for nodes n_e and n_j.

In contrast, when we have λ networks layered in a way that they are not parallel (see Figure 6.7), there will be an out of plane (z-direction) displacement. This is due to the xz component and yz component interactions that lead to the global Hessian having nonzero values at the locations that represent these interactions. For example, given a network where nodes n_i, n_j, are described by the position vectors

$$\mathbf{s}_i = [2.0, 2.0, 0.6]^T, \quad \mathbf{s}_j = [2.625, 0.75, 0.6]^T, \tag{6.32}$$

respectively, and all other nodes are left with unchanged position vectors, the global Hessian

can be written as

$$H = \frac{8000\sqrt{789}\mu}{622521}\begin{bmatrix} A_{e_{xx}}(\kappa,\mu) & -\frac{25}{8} & A_{e_{xy}}(\kappa,\mu) & \frac{5}{4} & 1 & -1 \\ -\frac{25}{8} & A_{j_{xx}}(\kappa,\mu) & \frac{5}{4} & A_{j_{xy}}(\kappa,\mu) & -1 & 1 \\ A_{e_{yx}}(\kappa,\mu) & \frac{5}{4} & A_{e_{yx}}(\kappa,\mu) & -\frac{1}{2} & -\frac{2}{5} & \frac{2}{5} \\ \frac{5}{4} & A_{j_{yy}}(\kappa,\mu) & -\frac{1}{2} & A_{j_{yy}}(\kappa,\mu) & \frac{2}{5} & -\frac{2}{5} \\ 1 & -1 & -\frac{2}{5} & \frac{2}{5} & A_{e_{zz}}(\kappa,\mu) & -\frac{8}{25} \\ -1 & 1 & \frac{2}{5} & -\frac{2}{5} & -\frac{8}{25} & A_{j_{zz}}(\kappa,\mu) \end{bmatrix}, \tag{6.33}$$

where $A_{e_{xy}}(\kappa,\mu)$ represents the xy components of the node n_e contributions of the local systems as some function of κ and μ. To confirm this results in an out of plane displacement, we can solve the system for some given values of κ and μ. For example, when $\kappa = 7.85398 \times 10^{-9}$ and $\mu = 3.141593 \times 10^{-4}$, the global displacement vector is found to be

$$\mathbf{U} = \begin{bmatrix} 0.00499928 \\ 0.00140852 \\ -0.00499968 \\ -0.00304621 \\ -0.00952402 \\ 0.00413602 \end{bmatrix}, \tag{6.34}$$

which has nonzero entries for the z displacements ($u_{e_z} = -0.00952402$ and $u_{j_z} = 0.00413602$).

6.2.4 Lattice Based Test Networks

In a step towards solving randomly oriented three dimensional networks built from the Mikado approach, we were able to obtain preliminary results by solving networks with predetermined direct intersections between fibres (see Figure 6.8). These have irregularity introduced by overlaying different lattice based components together to form a rigid structure. More specifically we take 25 vertical fibres extending the height of a $1\times1\times1$ cube and insert lattice forming plates through the vertical fibres with varied orientations. Neighbouring vertical fibres are linked with additional fibres that are in the plane of these plates.

The example seen in Figure 6.8 has an average coordination number of $z_c = 4.31$ with fibres of varied length, ℓ, radius $r = 0.01$, and a prescribed but irregular orientation. Given a shear strain $\gamma = 0.05$, the network is fixed in the plane $y = 0$, and sheared in the $y = 1$ plane in the positive x direction. Figure 6.8 (a) shows the structure prior to any shear, Figure 6.8 (b) shows the structure displaced after shearing, where the displacements are calculated using a direct solver. Figure 6.8 (c) shows the structure with every node displaced affinely. The total energy in this case is $E_{\text{aff}} = 5.0e^{-7}$. Comparing (b) and (c), we can see regions of the network in (b) where bending is more favourable than affine displacement. The total energy here is $E = 1.4e^{-7}$, where bending energy $E_{\text{bend}} = 1.3e^{-7}$ is the main contribution.

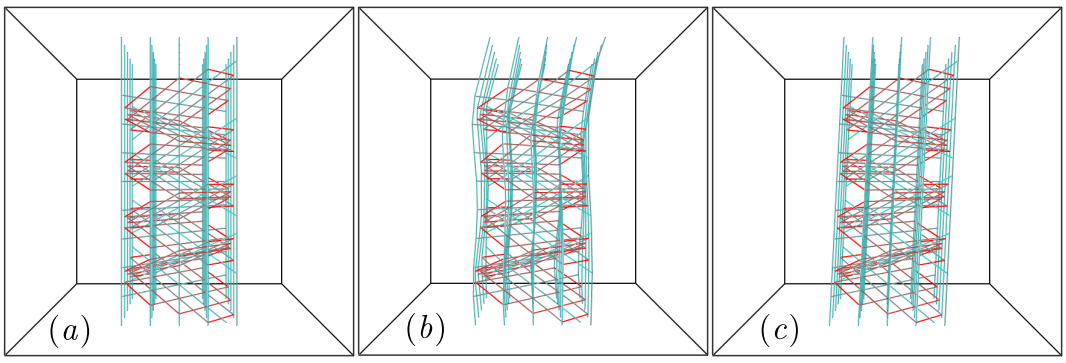


Figure 6.8: Visualisations of one of the lattice based 3D networks, for the undisplaced (a), displaced (b) and affinely displaced (c) generations of the network where $\gamma = 0.05$.

6.3 Sheet Based Networks in 3D

6.3.1 Modelling Sheet Based Networks

To begin we generate a simple 2D Mikado network, which is confined to the xy plane where $z = 0$. From here we wish to stack additional sheets on the original, by adding more xy oriented Mikado networks each with a larger z value. To maximise the disorder of the system, these additional networks can each be generated from a new random seed. Alternatively one can also achieve a reasonable level of disorder by generating a copy of the original network and applying a rotational transformation to each new sheet, which is what we shall do here. For simplicity we restrict rotation on any given sheet to integer multiples of 90°. This prevents the need for any additional fibre truncation after truncating the initial unrotated 2D Mikado sheet.

After generating the desired number of additional sheets, the network is cross-linked in two stages. The first stage is within sheet cross-linking, performed by looping over each individual sheet and cross-linking plane confined fibre pairs, and the second stage is between sheet cross-linking. For the experiments discussed in the sections to follow, if not stated otherwise it should be assumed that the first stage of cross-linking is completed using the 2D fibre intersection algorithm discussed in Section 5.1.3, and the second stage of cross-linking is completed using the latter of the approaches discussed in Section 6.3.2.

Any necessary post-processing is performed after cross-linking is complete. As we are interested in sheet network deformation under a variety of imposed shear directions, one post-processing step we perform is a network transformation that changes the orientation of the network. Alternatively, the network orientation can be fixed and shear direction can be varied, however we chose to translate and rotate node coordinates to minimise the changes to the numerical model. Figure 6.9 shows an example of possible network orientations that we consider in more detail in the sections to follow.

In Chapter 5 we generally fixed the lower and upper boundaries of our 2D networks ($y = 0$,

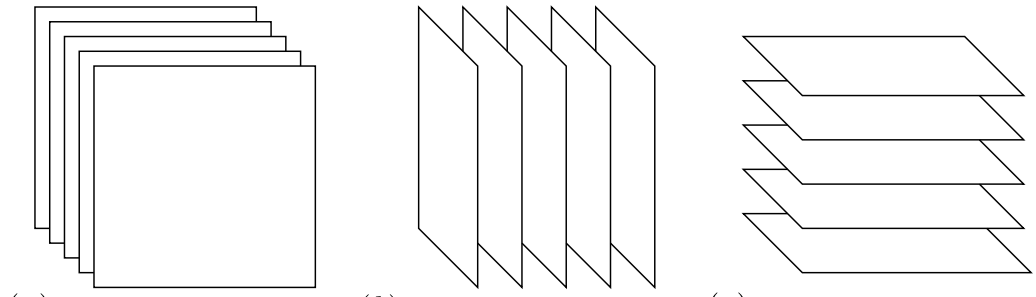


Figure 6.9: Sheet networks can be sheared with respect to 3 different orientations: the xy (a), yz (b), and xz (c).

$y = H$ respectively) while keeping the left and right boundaries ($x = 0$, $x = W$) free. Accounting for this requires another post-processing step to update any incorrect node types resulting from odd integer rotations of the original 2D Mikado sheet. Furthermore, introducing periodicity in the x direction requires these rotated sheets to be truncated again. Rather than implementing a post-processing step to resolve these issues we instead choose to fix all 6 boundaries. This allows any boundary intersecting fibres of the original sheet to be truncated once, and any of the introduced boundary nodes to have their node type set once.

6.3.2 Comparing Approaches for Cross-linking Sheets

In discussing a selection of approaches for cross-linking sheet based networks, we will begin with the approach that matches the closest to the needlepunching technology discussed in Section 1.1.1. Next we discuss a simplification that rests on the existing robust 2D intersection algorithm. We finish with an approach that is arguably more abstracted from the application, but allows for variation in the angle of cross-linking fibres. All three approaches assume that fibres lying within each sheet are initially connected using the standard 2D connection algorithm outlined in Section 5.1.3. Once connected in the plane in which they lie, these sheets can then be connected out of plane via one of the approaches that are outlined below.

Network Spanning Needle Cross-linking

To directly mimic the needleboard that punches through the various sheets of fibres, we can model a set of network spanning fibres or "needles" that pass through every sheet at an angle perpendicular to plane that the sheet fibres lie in. These network spanning needles can be arranged in a lattice formation as seen in Figure 6.10. For each needle, the coordinates at which it passes through every sheet represent points lying on the central line of the needle. Nodes can then be generated at each of these central points, and should be looped over for introducing a

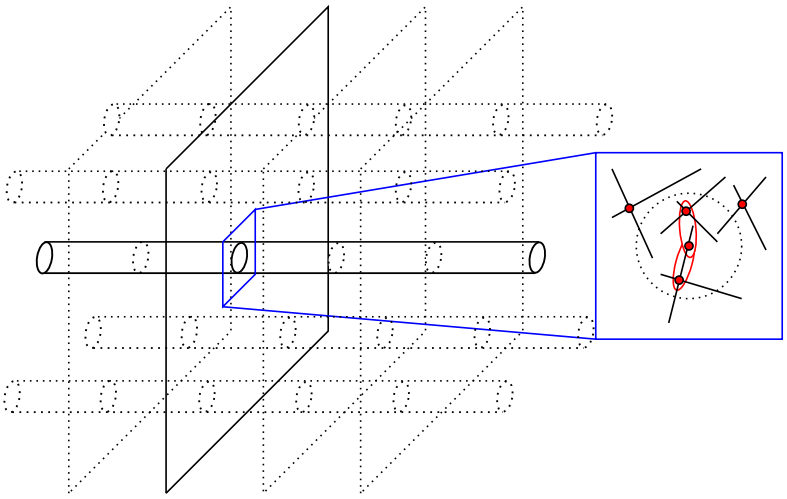


Figure 6.10: An example of how a layered network could be modelled with network spanning needles.

cross-linking between each of the needles to the sheets, where the needle radius represents the cross-linking search tolerance from each of the nodes generated on the central line. When an existing sheet node is found to be within the search distance of a needle centre point, it is cross-linked via node coupling to the needle. Depending on the radius of the needles, there could be many existing sheet nodes that are coupled to each needle. Without further investigation it is difficult to determine the exact implications of this approach on the mechanical properties. Though even without implementation, it is clear that there are certain design considerations that need to be accounted for. For example, a cross-linking search tolerance that exceeds half the distance between needles could lead to nodes being coupled to multiple needles.

Layer Based Cross-linking

An alternative approach is to cross-link the network by considering neighbouring sheets and cross-link them independently of non-neighbouring sheets. Aside from increasing the overall network disorder, an advantage of this approach is that it lends itself to the existing 2D cross-linking algorithm. To do this, neighbouring sheets are temporarily mapped into the same plane as can be seen in Figure 6.11. From this mapping, new direct points of intersection can be found by comparing the existing fibres of one sheet with the existing fibres of another. Each of these new intersection points can then be mapped into the two original planes, where they become the coordinates of two cross-linking nodes that form the ends of a new cross-linking fibre.

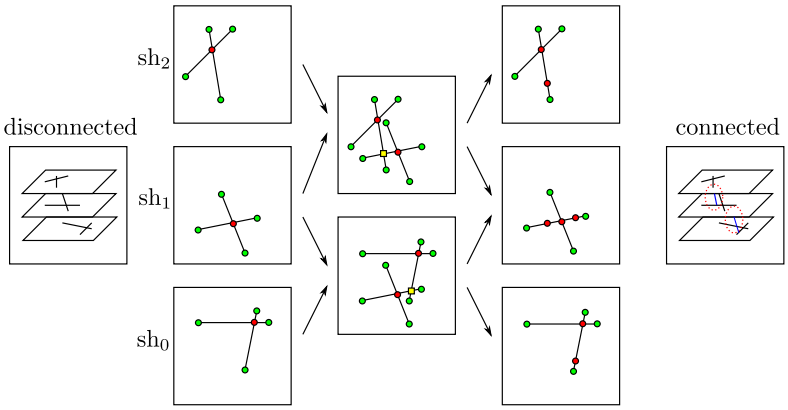


Figure 6.11: An example of how a layered network could be cross-linked using the existing 2D intersection algorithm.

Map Based Cross-linking

In both of the approaches discussed above, cross-linking fibres are introduced at perpendicular angles to the fibre sheets by inserting new nodes within each sheet. In the case that all the cross-linking fibres are exactly perpendicular to the sheets, they would not contribute to the mechanics of the network. An alternative to adding further nodes as part of the cross-linking process, is to exploit the existing architecture and choose to map directly between nodes of neighbouring sheets (see Figure 6.12). In this way we also reduce system anisotropy by varying the angles of

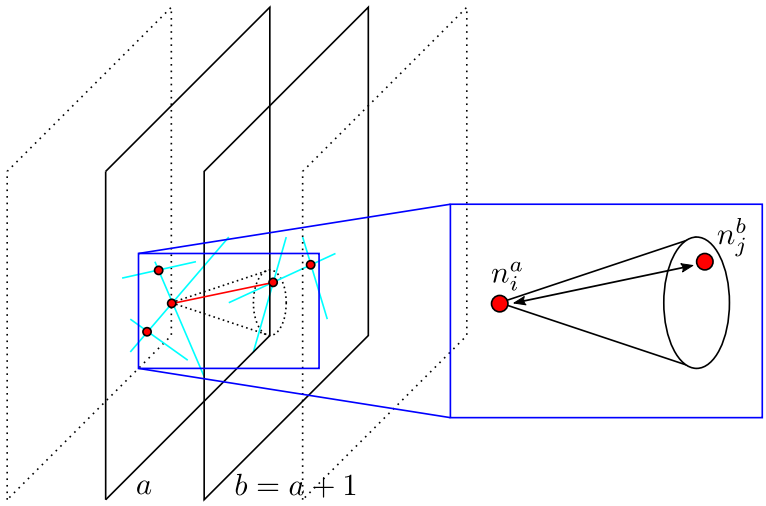


Figure 6.12: An example of 1-1 cross-link mapping between sheets of a layered network. Each node n_i^a of sheet a has a conical search tolerance for finding the closest n_j^b of sheet b.

needles that connect fibre sheets. Nodes can be mapped to many nodes of neighbouring sheets, but we choose to constrain the cross-linking to a maximum of a 1-1 mapping by recording whether the nodes of each sheet have existing connections to its neighbouring sheet.

To elaborate further, for every internal node n_i^a of every fibre in sheet a and every internal node n_j^b of every fibre in sheet $b = a+1$, find the closest n_j^b to n_i^a within a given search tolerance. If no such node exists then we leave n_i^a disconnected to sheet b and we move on to the next node of sheet a. Otherwise we connect n_i^a and n_j^b provided n_j^b has not already been connected to any other nodes in sheet a. Here we acknowledge the possibility of introducing a small bias by assigning a direction to move between sheets (e.g. $b = a + 1$ rather than $b = a - 1$), but we expect there to be no significant difference to the mechanical properties for the shears we are considering.

6.3.3 Sheet Based Network Verification

In building a sheet based network, there are various verification steps that can be taken using simplified sheet based architectures prior to considering more complex approaches for connectivity. The networks discussed in this section derive from simple transformations of random 2D networks with the same parameters as those used for the literature validation presented in Section 5.4.1.

The first assumption we can make about a 3D network consisting of a single sheet, is that the energy resulting from a linear macroscopic shear in the direction of the plane in which it lies, should be equal (within numerical precision) to the energy of the same network in 2D resulting from a linear macroscopic shear. Table 6.1 verifies this assumption for aperiodic networks with varying fibre counts and fixed radius and Table 6.2 adds verification for x-periodic networks with varying fibre counts and fixed radius. Both tables demonstrate relative errors that are consistent with usual floating point errors.

N_f	E_{stretch}	E_{bend}	E	G
100	3.924632e-13	1.487914e-12	3.276565e-16	3.508026e-16
150	1.340552e-12	1.017655e-11	2.453975e-16	2.292863e-16
200	5.571751e-12	7.570341e-11	2.836967e-16	2.668641e-16
250	5.993895e-12	1.176886e-10	7.701072e-16	7.657722e-16
300	1.306570e-11	3.721053e-10	3.551340e-16	3.217689e-16

Table 6.1: A table comparing the relative error between various mechanical property estimates of aperiodic 3D in-plane networks and their aperiodic 2D counterparts for various fibre counts. Each 3D case consists of a single sheet in the xy plane sheared in the x direction [see Figure 6.9 (a)] with $r = 1.996 \times 10^{-3}$ and $\ell \approx 0.5$. Networks are fixed at all boundaries and results are averaged over 9 seeds.

For sheet networks sheared in the xy direction, with constant fibre length, radius, and density, we expect a constant increase in G for every additional sheet generated from the same seed,

N_f	E_{stretch}	E_{bend}	E	G
100	2.054979e-12	5.335187e-12	3.201005e-16	3.174933e-16
150	5.108393e-12	2.985982e-11	3.444348e-16	3.288404e-16
200	4.561160e-13	5.130491e-12	3.083158e-16	2.890491e-16
250	4.687666e-12	7.799513e-11	6.377136e-16	6.359604e-16
300	1.215183e-11	2.926312e-10	2.681824e-16	2.677332e-16

Table 6.2: A table comparing the relative error between various mechanical property estimates of x-periodic 3D in-plane networks and their x-periodic 2D counterparts for various fibre counts. Each 3D case consists of a single sheet in the xy plane sheared in the x direction [see Figure 6.9 (a)] with $r = 1.996 \times 10^{-3}$ and $\ell \approx 0.5$. Results are averaged over 9 seeds.

provided there are no connecting fibres between sheet and each sheet has the same orientation. We verify this expectation in Figure 6.13, where we see G increase at a constant rate for a variety of fixed numbers of fibres and increasing sheets. Some additional verification performed at

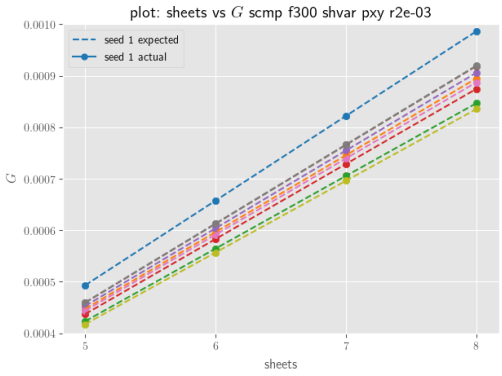


Figure 6.13: Verification of a constant increase in G for a fixed number of fibres per sheet, $N = 300$, and increasing number of disconnected and unrotated sheets, M. Dashed lines represent the expected value of G for a given seed and the points represent the calculated value. Lines and points are coloured to represent the different seeds used to generate the network instances.

this stage included recording the consistency of various ratios for increasing number of sheets, including the ratio of bending to stretching [see Figure 6.14 *(left)*] and the ratio of affine to nonaffine energy [see Figure 6.14 *(right)*].

6.3.4 Mechanical Property Analysis

To describe the networks used for results collection, we reiterate the generation procedure discussed in Section 6.3.1 with some notation and terminology added. Generated networks are composed of M parallel 2D sheets that together fill a volume, where each sheet is an isotropic 2D network of N fibres. Each Mikado sheet of a given network is generated from the same

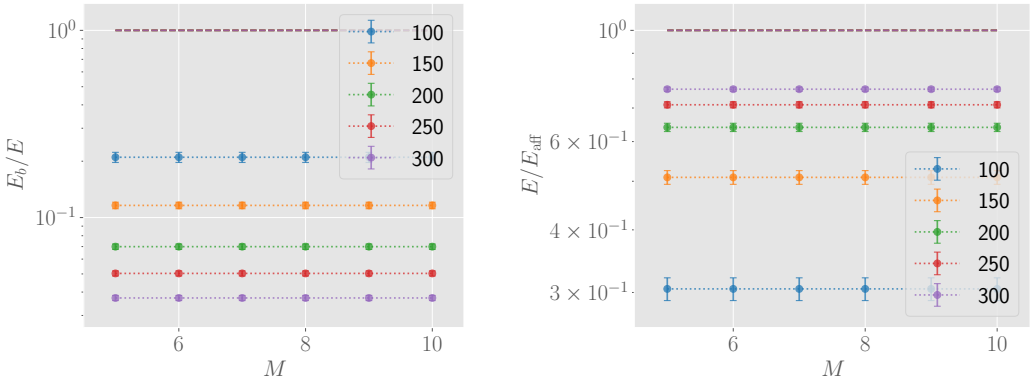


Figure 6.14: Figures verifying the consistency of the model for connected but not rotated sheet networks sheared in the xy plane. The ratio of bending energy over total energy E_{bend}/E *(left)* and of total energy over affine energy E/E_{aff} *(right)* against an increasing number of sheets.

random seed, but has been rotated and translated. Individual sheets have an area A, and the separation distance between each sheet pair is $h \propto M^{-1}$. Since sheets of a given network are generated from the same seed, they share a common value of fibre density $\rho = \frac{N\ell}{A}$. The original Mikado *intrasheet* fibres are cross-linked within the same sheet in the usual manner (see Section 5.1.3), and are connected between sheets by *intersheet* cross-linking fibres using the map based cross-linking discussed above. For a fixed shear direction, network sheets are collectively re-oriented as a post-processing step to monitor the effect of material anisotropy on properties. Equivalently, the post-processing could be removed and shear direction could be varied.

Finite Size Effect Testing

We validate that the finite size effects are small enough to justify aperiodic fixed boundaries by comparing how the shear modulus grows with increasing sheet dimensions for fixed fibre density. From Figure 6.15 we confirm that the growth in sheet size has no significant effect on predicted value of the shear modulus for fixed fibre density.

Intrasheet Fibre Density

Fixing the number of sheets, M, each with dimension $2\ell \times 2\ell$, we can observe the effect of increasing overall fibre density for different shear directions, controlled by increasing the number of intrasheet fibres. This in turn increases the number of intersheet cross-linking fibres. We can write the number of fibres per sheet as N, and the number of cross-linking fibres per sheet (i.e. connecting a pair of sheets) as N_{cf}.

From 6.16 *(left)*, we can see that shearing an xy oriented network produces a much larger shear modulus than the yz or xz oriented networks, for all of the tested fibre densities. Moreover,

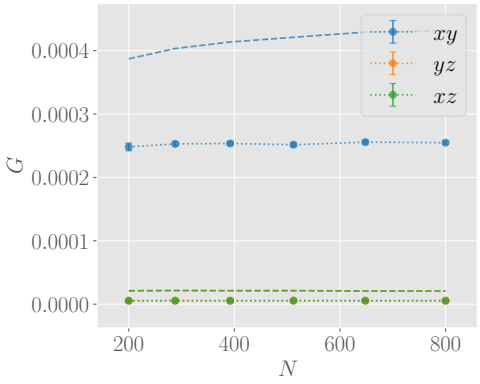


Figure 6.15: A plot to demonstrate effects of finite size on the shear modulus predictions, G, of the sheet networks. Network sheet dimensions were increased with fibre density kept constant. Dashed lines correspond to the affine predictions of the modulus, G_{aff}, for the respective shear directions. Lines corresponding with yz shear lie underneath the lines for xz shear. The flatness of curves show G remains constant for increasing system size.

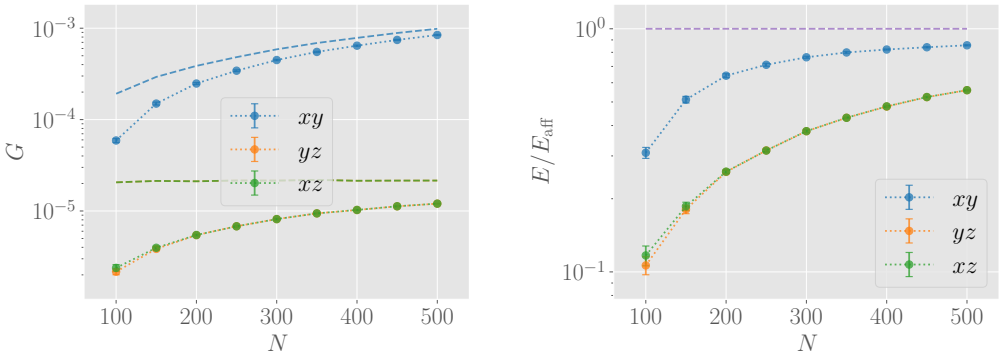


Figure 6.16: Plots showing the shear modulus, G *(left)*, and affinity, G/G_{aff} *(right)*, for an increasing number of fibres per sheet, N, and varied shear directions, where the sheet count and fibre radius are fixed at $M = 5$ and $r = 1.996 \times 10^{-3}$ respectively. Dashed lines correspond to the affine predictions of the moduli for the respective shear directions. In *(right)* this is given as the purple line with slope $y = 1$.

all orientations show an increase in shear modulus for increasing fibre density, as expected. From Figure 6.16 *(right)*, we observe that the xy oriented network behaves closer to affine than the yz or xz oriented networks for the considered range. The xy orientation shows a sharper initial increase to affinity.

From Figure 6.17 *(left)* we see that all orientations receive an increasing stretch energy

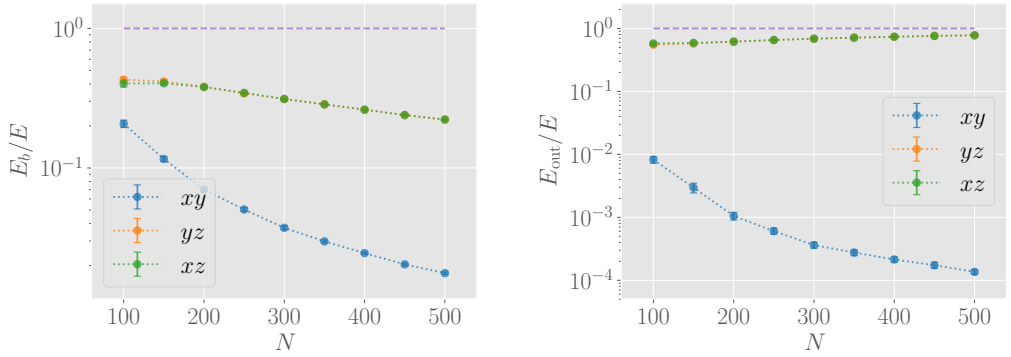


Figure 6.17: Plots showing the bend to stretch energy ratio, E_{bend}/E *(left)*, and intersheet to intrasheet energy ratio, E_{out}/E *(right)*, for an increasing number of fibres per sheet, N, and varied shear directions, where the sheet count and fibre radius are fixed at $M = 5$ and $r = 1.996 \times 10^{-3}$ respectively. Dashed lines with slope $y = 1$ represent bending dominated energy *(left)* and intersheet dominated energy *(right)* respectively. Lines corresponding with yz shear lie underneath the lines for xz shear.

contribution to the total energy as fibre density is increased. For the range of fibre densities considered, the yz and xz orientations have a higher bending contribution to the total energy. Figure 6.17 *(right)* shows that the energy of the xy oriented network becomes increasingly dominated by intrasheet fibre deformations for increasing fibre density. In contrast yz and xz orientations that have a larger intersheet energy, become more dominated by the intersheet energy with increasing fibre density. In all of the discussed plots, there is a strong similarity in the yz and xz orientation values, suggesting a relationship between the two.

Varying the Sheet Separation Distance

To investigate the effects of sheet separation distance, recall $h \propto M^{-1}$. It follows that we can fix the number of intrasheet and intersheet fibres per sheet and increase the number of sheets within a fixed domain.

Figure 6.18 *(left)* shows a steady decrease in shear modulus for an increase in the sheet separation distance for any of the three sheet orientations. We note that Figure 6.18 *(left)* shows the total network shear modulus for all sheets. To get a clearer picture of the effect of increased sheet separation distance for fixed fibre density, we can approximate the energy per single sheet and single set of connecting cross-linking fibres by scaling the total energy by the number of sheets [see Figure 6.18 *(right)*]. From this we observe a steady decrease of the

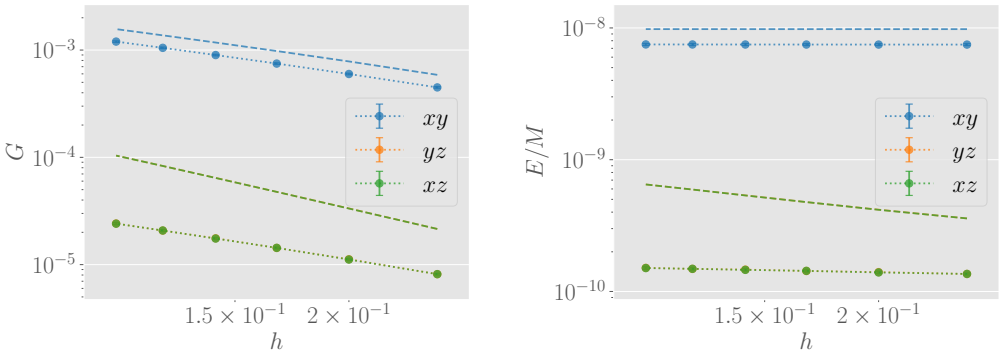


Figure 6.18: Plots showing the total network shear modulus, G *(left)*, and the energy per sheet E/M (where here M refers to the number of sheets excluding boundary sheets) for an increasing sheet separation distance, h, and varied shear directions, where the number of fibres per sheet and fibre radius are fixed at $N = 300$ and $r = 1.996 \times 10^{-3}$ respectively. Dashed lines represent the affine predictions of the moduli for the respective shear directions. Lines corresponding with yz shear lie underneath the lines for xz shear.

energy per sheet for all orientations, similar to the total network shear modulus. Figure 6.19 shows how affinity has no dependence of separation distance for the xy oriented network, but

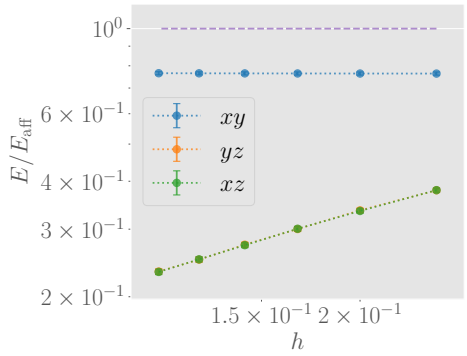


Figure 6.19: A plot showing affinity, G/G_{aff} *(right)*, for an increasing sheet separation distance, h, and varied shear directions, where the number of fibres per sheet and fibre radius are fixed at $N = 300$ and $r = 1.996 \times 10^{-3}$ respectively. The dashed line with slope $y = 1$ represents pure affine behaviour. Points for yz shear lie underneath the points for xz shear.

increases with increasing separation distance for both the yz and xz orientations.

Figure 6.20 *(left)* shows how increased sheet separation distance has no effect on the ratio of bending to stretching energy for xy oriented networks, but promotes a higher contribution of stretching energy in the yz and xz oriented networks. In terms of the intrasheet/intersheet energy ratios, Figure 6.20 *(right)* demonstrates decreasing dependence of the total energy on intersheet energy for xy oriented networks as sheet separation increases, and increasing dependence on intersheet energy for yz and xz oriented networks.

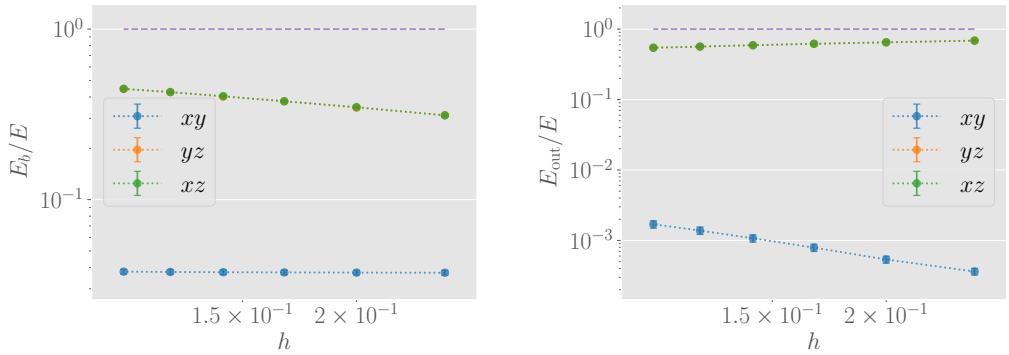


Figure 6.20: Plots showing the bend to stretch energy ratio, E_{bend}/E *(left)*, and intersheet to intrasheet energy ratio, E_{out}/E *(right)*, for an increasing sheet separation distance, h, and varied shear directions, where the number of fibres per sheet and fibre radius are fixed at $N = 300$ and $r = 1.996 \times 10^{-3}$ respectively. Dashed lines with slope $y = 1$ represent bending dominated energy *(left)* and intersheet dominated energy *(right)* respectively. Lines corresponding with yz shear lie underneath the lines for xz shear.

Varying the Radius

A further investigation of the mechanical properties of the networks can be undertaken by varying the fibre radius, both inter- and intrasheet for a fixed number of sheets and a fixed number of fibres.

From Figure 6.21 *(left)* we see that all network orientations show an increase in shear

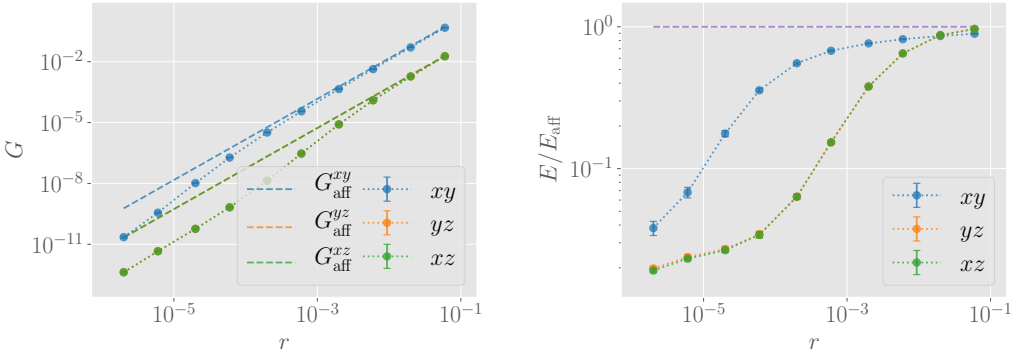


Figure 6.21: Plots showing the shear modulus, G *(left)*, and affinity, G/G_{aff} *(right)*, for an increasing fibre radius, r, and varied shear directions, where the number of fibres per sheet and number of sheets are fixed at $N = 300$ and $M = 5$ respectively. Dashed lines represent the affine predictions of the moduli for the respective shear directions. Lines corresponding with yz shear lie underneath the lines for xz shear.

modulus, G, for increasing radius as expected. We also see from Figure 6.21 *(right)* that all network orientations demonstrate a transition towards affinity under the imposed shear for increasing fibre radius. The xy orientation shows increased affinity for smaller choices of radius than for the yz and xz orientations.

From 6.22 *(left)* we see that the yz and xz oriented networks move from a stretching dom-

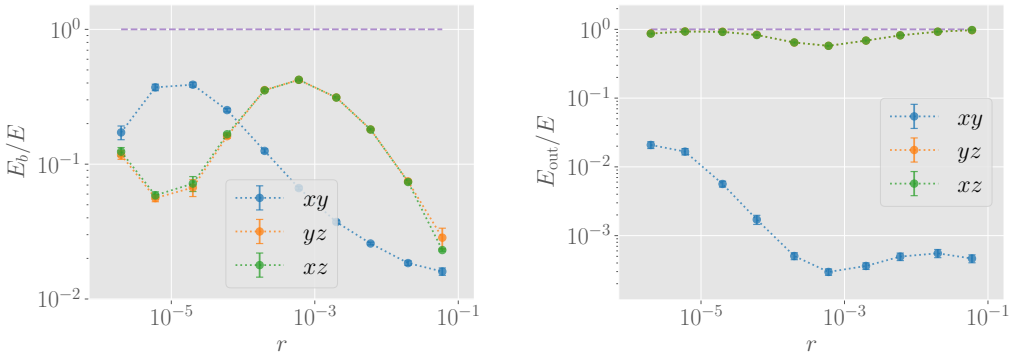


Figure 6.22: Plots showing the bend to stretch energy ratio, E_{bend}/E *(left)*, and intersheet to intrasheet energy ratio, E_{out}/E *(right)*, for an increasing fibre radius, r, and varied shear directions, where the number of fibres per sheet and number of sheets are fixed at $N = 300$ and $M = 5$ respectively. Dashed lines with slope $y = 1$ represent bending dominated energy *(left)* and intersheet dominated energy *(right)* respectively. Lines corresponding with yz shear lie underneath the lines for xz shear.

inated regime to a regime with higher bending contribution for increasing radius, whereas the xy oriented network moves from a mixed stretching and bending regime to a stretching dominated regime. Figure 6.22 *(right)* shows that yz and xz oriented networks are dominated by intersheet energy, with a slight increase in intersheet energy for intermediate radius values. The xy oriented network is largely dominated by intrasheet energy, with the largest contribution of intersheet energy occurring for a small radius and the ratio decreasing for increasing radius. The plot suggests a possible minimum in intersheet energy contribution towards a radius of $r = 10^{-3}$.

We note that some of the interesting features of Figure 6.22 (such as the peak of the xy orientation and trough of the xz and yz orientations on the left of Figure 6.22 *(left)*) occur at unphysical (extremely small or large) values of the radius. Therefore we expect this behaviour is irrelevant to real materials and unlikely to be observed in actual experimental data. Figure 6.23 shows the shear modulus G *(left)* and ratio E_{bend}/E *(right)* of a single sheet of 300 fibres for varied radii in 2D modelled both with fixed and periodic boundaries. From this we see that the recorded values agree well between the two modelled networks except for when the radius is extremely small or extremely large. This suggests that the effects of fixed boundaries could be stronger for these unphysical choices of radius. Since this only arises for unphysical radii, we have not investigated the effects of different boundary conditions further.

Network Orientation Relationships

As observed in the results discussed above, there is no significant difference in the predicted mechanical properties of yz and xz oriented networks. To clarify this relationship, we can

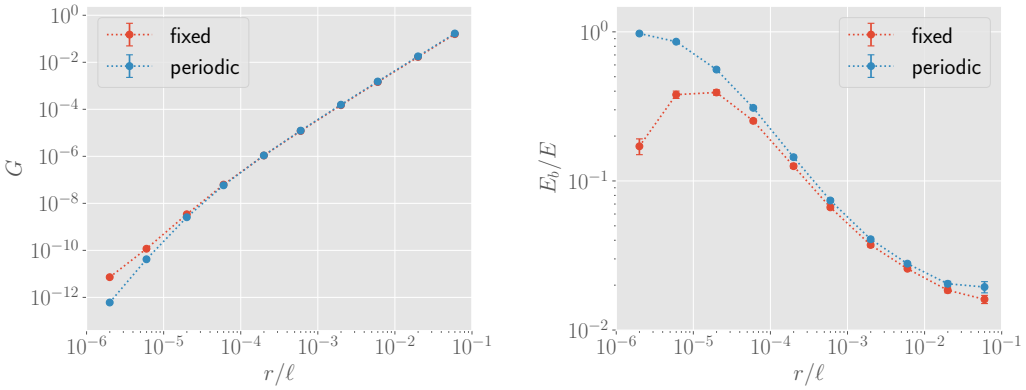


Figure 6.23: A plot comparing shear modulus G *(left)* and the relative bending energy E_{bend}/E *(right)* of a periodic and fixed boundary 300 fibre single sheet network in 2D for various choices of radius.

plot the ratio of shear modulus of the different network orientations for each of the discussed experiments. Figure 6.24 *(left)* shows the ratios of the shear modulus of the xy orientation,

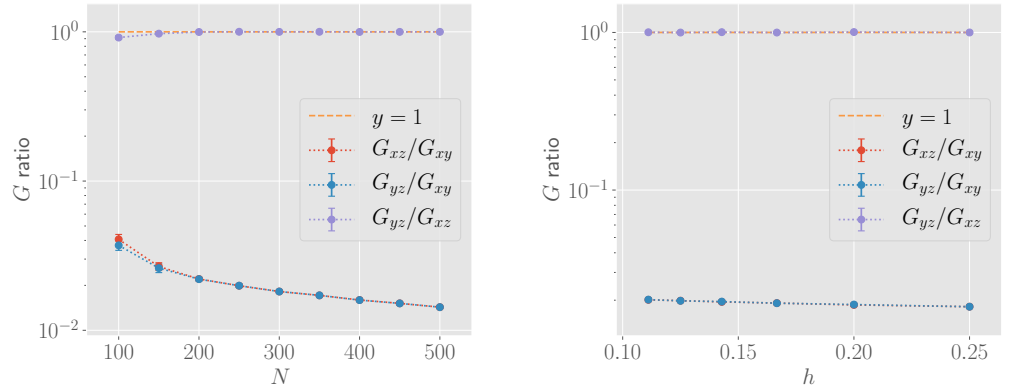


Figure 6.24: Plots to demonstrate the existing relationship between the yz and xz oriented sheet networks, by taking the ratios of the shear moduli for each orientation for increasing fibre density per sheet *(left)* and increasing separation distance, h, *(right)*.

G_{xy}, yz orientation, G_{yz}, and xz orientation, G_{xz}, for increasing fibre density. Figure 6.24 *(right)* shows these same ratios for increasing sheet separation distance h.

6.3.5 Energy Scaling

We now present a scaling argument which we will use to support our interpretation of the results presented thus far [24]. Using arguments given elsewhere (see Appendix B.4), we give

the expected number of cross-links within a single sheet as

$$\langle n_\times \rangle = \frac{\rho^2}{\pi}, \tag{6.35}$$

noting that the prefactor $1/\pi$ given in (6.35) is only exact for an infinite (or periodic) system. Thus $\langle n_\times \rangle \sim N^2$, since $\rho = N\ell/A$ from equation (2.19) in Chapter 2. The number of connecting cross-linking fibres between sheets, N_{cf}, are the result of a 1-1 mapping between selected cross-links, $N_{cf} \sim \langle n_\times \rangle$ with some undetermined constant of proportionality. It follows that N_{cf} should also scale with N^2, as confirmed by Figure 6.25, where a line can be fit to give an

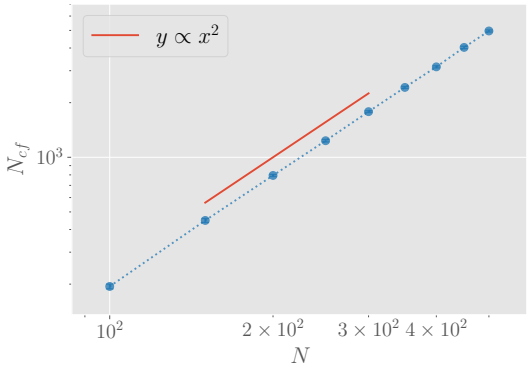


Figure 6.25: A plot to show the relationship between the number of intrasheet fibres per sheet per unit area, N, and the intersheet cross-linking fibres per sheet per unit area, N_{cf}. The trend is quadratic as verified by the slope of $y \propto x^2$.

exponent of 2.007796 with an error of 6.66698×10^{-3}.

Assuming the range for intersheet cross-linking is relatively only slightly larger than h, and the connected cross-links minimise the distance within the search tolerance, the length of cross-linking fibres will be bounded by, but close in value to, the separation distance, h, and generally oriented almost (but not exactly) perpendicular to the Mikado network sheets.

To derive expressions for the scaling of the intrasheet and intersheet total stretching energies, we begin by assuming the deformation is affine. Recall from Chapter 3 that the stretching energy of a single fibre of length ℓ with change in length $\delta\ell$ can be given as

$$E_{\text{stretch}} = \frac{1}{2}\frac{\mu}{\ell}\delta\ell^2. \tag{6.36}$$

For intrasheet fibres, we note that our assumption of affinity implies that segments do not need to be considered individually and fibres can be treated as a single unit with dangling ends included. As the extension, $\delta\ell$, of an intrasheet fibre scales with $\gamma\ell$, the energy of the fibre scales with

$$\sim \frac{\mu}{\ell}(\gamma\ell)^2 \sim \mu\gamma^2\ell. \tag{6.37}$$

To get the scaling for the energy of the entire sheet per unit area, we multiply through by the

number of intrasheet fibres per unit area, N/A, such that

$$\frac{E_{\text{in}}^{\text{aff}}}{A} \sim \frac{N}{A}\mu\gamma^2\ell. \tag{6.38}$$

Using the definition of fibre density, (6.38) is equivalent to $\frac{1}{A}\frac{\rho A}{\ell}\mu\gamma^2\ell$, thus

$$\frac{E_{\text{in}}^{\text{aff}}}{A} \sim \rho\mu\gamma^2. \tag{6.39}$$

For intersheet cross-linking fibres, we use the sheet separation distance, h, to approximate their length such that we take the spring constant to be $k \approx \mu/h$. In this case, the extension scales as $\sim \gamma h$, thus the per-fibre energy scaling is

$$\sim \frac{\mu}{h}(\gamma h)^2 \sim \mu\gamma^2 h. \tag{6.40}$$

We know that the total number of cross-linking fibres per sheet per unit area scales with the expected number of cross-links per sheet per unit area, $\langle n_\times \rangle$, which is $\sim \rho^2$ from (6.35), thus the total intersheet energy over all fibres per unit area scales with

$$\frac{E_{\text{out}}^{\text{aff}}}{A} \sim \frac{N_{cf}}{A}\mu\gamma^2 h \sim \rho^2\mu\gamma^2 h. \tag{6.41}$$

We note that the geometric prefactor such that $\delta\ell \sim \gamma h$ will be small since fibres are almost perpendicular to the Mikado sheets.

6.3.6 Discussion of Trends

We open our discussion with some general points that will guide the interpretation of the results. Firstly we note that, as in the case of random 2D networks, although the mean deformation must be affine in order to satisfy the applied strain, the total energy can be reduced by local nonaffine deformation. Secondly, intersheet connecting fibres each consist of just two nodes at cross-link fibre endpoints, and can therefore only deform through stretching. Since these cross-linking fibres will be almost perpendicular to the sheets, they will only weakly deform in any of the considered orientations. Lastly, differences in how intrasheet and intersheet energy is minimised by node displacements leads to competing effects. We split our discussion into the trends seen for each of the individual orientations, beginning with the xy orientation and following with the yz and xz orientations.

For the xy oriented networks, we see that the total network energy is dominated by the intrasheet energy, E_{in}, [see Figure 6.26 *(left)*] due to the small geometric prefactor of E_{out}. It follows that the total energy and total nonaffinity are largely independent of h, and demonstrates the same trends with ρ as in the case of pure 2D sheets [see Figure 6.27 *(left)* and Figure 6.28 *(left)*]. Nonaffine displacements may reduce intrasheet and intersheet energies below their

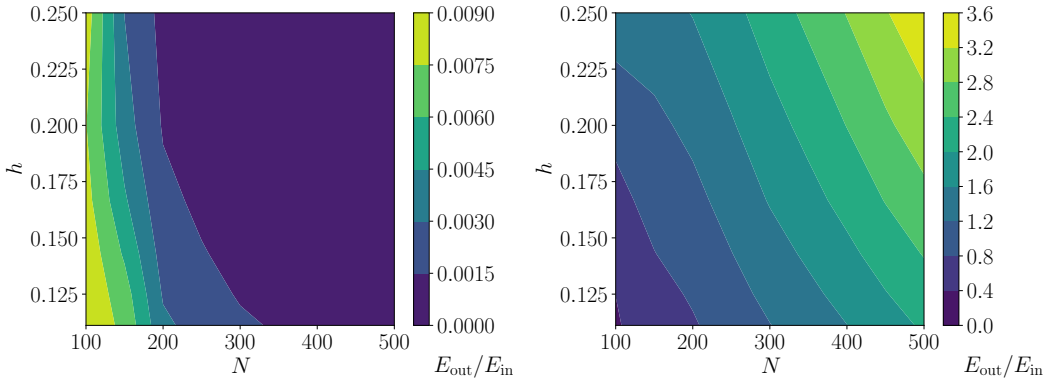


Figure 6.26: Contour plots to show the general trend of the intersheet/intrasheet energy competition, $E_{\text{out}}/E_{\text{in}}$, as ρh increases, for xy oriented *(left)* and xz oriented *(right)* networks, where we recall that $\rho \sim N$.

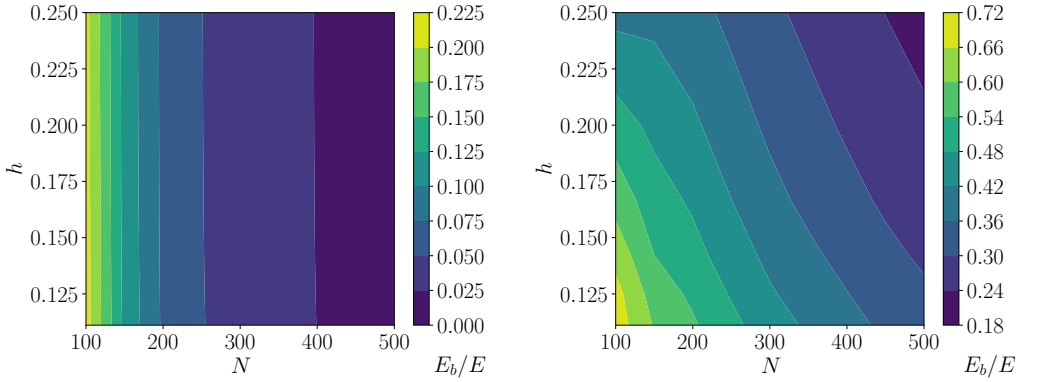


Figure 6.27: Contour plots to show the general trend of bending to stretching energy competition, expressed through the ratio of E_{bend}/E, as ρh increases, for xy oriented *(left)* and xz oriented *(right)* networks.

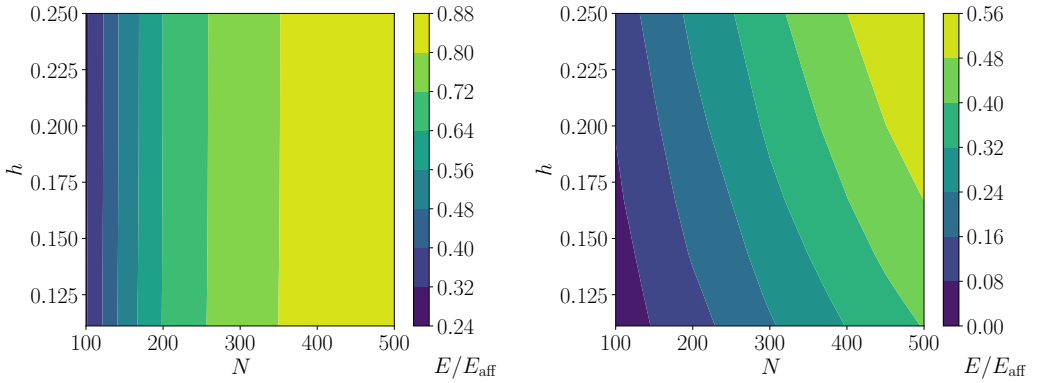


Figure 6.28: Contour plots to show the general trend of affinity, expressed through the ratio of E/E_{aff}, as ρh increases, for xy oriented *(left)* and xz oriented *(right)* networks.

respective affine predictions, but the exact distribution of the energy across intrasheet and intersheet fibres is decided based on what minimises the total network energy. From equations (6.39) and (6.41), we note that $E_{\text{out}} \sim \rho h E_{\text{in}}$. Thus for larger ρ and h, node displacements that relax intersheet fibre stretching will result in a larger total energy reduction. We have already noted that displacements minimising intersheet and intrasheet energy will not necessarily be the same, so we do not expect a similar decrease in intrasheet energy. Thus $E_{\text{out}}/E_{\text{in}}$ should decrease as seen in Figure 6.26 *(left)*. On the other hand, we expect node displacements that primarily reduce intrasheet energy to give a greater reduction in total energy when ρ and h are smaller. Thus although we still expect $E_{\text{out}}/E_{\text{in}} \ll 1$, it should be higher for low ρ, h, as confirmed by Figure 6.26 *(left)*.

For yz and xz oriented networks, the affine prediction of the intrasheet energy is zero, whereas the intersheet affine energy prediction scales as in equation (6.41) $\frac{E_{\text{out}}^{\text{aff}}}{A} \sim \rho^2 \mu \gamma h$. As before, the nonaffine deformations may be favoured locally in order to minimise the energy, which in this case will increase the intrasheet energy from zero. For the shears tested here, one could argue that E_{out} cannot be significantly reduced from the affine prediction by nonaffine deformation, and we therefore expect $E_{\text{out}} \sim \rho h E_{\text{in}}$ to hold approximately true [as supported by Figure 6.26 *(right)*], though we note that this is only a crude, scaling-level argument and a more precise theory is desirable. As with the xy oriented networks, the degree of affinity of the yz and xz oriented networks also seems to broadly increase with ρ and h [see Figure 6.28 *(right)*]. This suggests the increase in affinity derives from increasing constraints imposed by the intersheet cross-links for increasing ρh. This is similar to the increased affinity observed for large ρ in 2D networks.

6.4 Random Networks in 3D

While there are many industrial applications for the layered disordered networks modelled thus far, there is also much interest in the modelling of fully 3D random fibre networks, for example in applications where random networks are more representative of the real system (such as the biopolymer networks discussed in Chapter 1). Further interest also comes from the general study of fibre network mechanics where much existing work uses random networks (see Chapter 2), allowing for a more direct comparison of results with other studies. Here we discuss some preliminary work on our 3D Mikado networks, giving a brief outline of the generation procedure, before moving on to some investigation of the properties of such networks.

6.4.1 Generating Mikado Networks in 3D

Beginning in the same way as in Section 5.1.1, the process for generating the 3D Mikado networks can be summarised as follows:

1. Generate a 3D box of some given dimensions.

2. Generate a pair of coordinates that will form the end points of a fibre.
3. Join the coordinate pair to form a regular cylindrical fibre object with a given radius.
4. Repeat steps 2-3 for as many fibres as desired.
5. Loop over all fibre pairs and calculate the minimum distance.
   (a) If the minimum distance does not lie within a given tolerance, continue to the next fibre pair.
   (b) Otherwise, connect using either the cross-linking fibre approach or through node coupling (see Section 6.1.1).
6. If node coupling was used, loop over all the newly created node couplings and execute the couple merging procedure.

6.4.2 Preliminary Results

Using the procedure outlined above, we generated several 3D Mikado networks that were cross-linked using the coupled node approach. Networks were periodic in the x and z directions with box dimensions $(1 \times 1 \times 1)$, consisting of fibres with length $\ell \approx 0.5$ and radius $r = 0.001$. These networks were then passed into the numerical model to provide mechanical property predictions for a macroscopic shear strain of $\gamma = 0.01$ applied at the upper boundary. Figure 6.29 shows a sample of post-solve network visualisations for (a) 1200, (b) 1600, and (c) 2000 fibres respectively. As expected, we see that the 1200 fibre network has visibly more disconnected fibres (coloured green) than the 2000 fibre network. It is difficult to draw any clear observations from the visualisations on the difference in relative proportion of bending dominated fibres (coloured red) to stretching dominated fibres (coloured blue).

Table 6.3 shows the calculated mechanical properties of the networks for a range of fibre densities. Here we see that unlike their 2D counterparts, for the densities considered here,

N_f	$E_{\text{stretch}}/E_{\text{aff}}$	$E_{\text{bend}}/E_{\text{aff}}$	E/E_{aff}
2000	1.74e-04±1.48e-05	4.39e-03±1.22e-04	4.57e-03±1.34e-04
2400	9.03e-04±5.17e-05	1.04e-02±2.01e-04	1.13e-02±2.47e-04
2800	3.07e-03±1.25e-04	1.94e-02±2.92e-04	2.25e-02±4.03e-04
3200	7.56e-03±2.35e-04	3.07e-02±4.86e-04	3.83e-02±6.94e-04

Table 6.3: A table comparing various mechanical property estimates scaled to the affine energy of x and z periodic random 3D Mikado networks cross-linked using node coupling. Results are averaged over 9 seeds and presented with their respective standard errors. All fibres were generated with a length of $\ell \approx 0.5$ and a radius of $r = 0.001$.

the networks are bending dominated. Additionally, we also note that through fibre bending, nonaffine node displacements can minimise the total network energy well below the affine predictions.

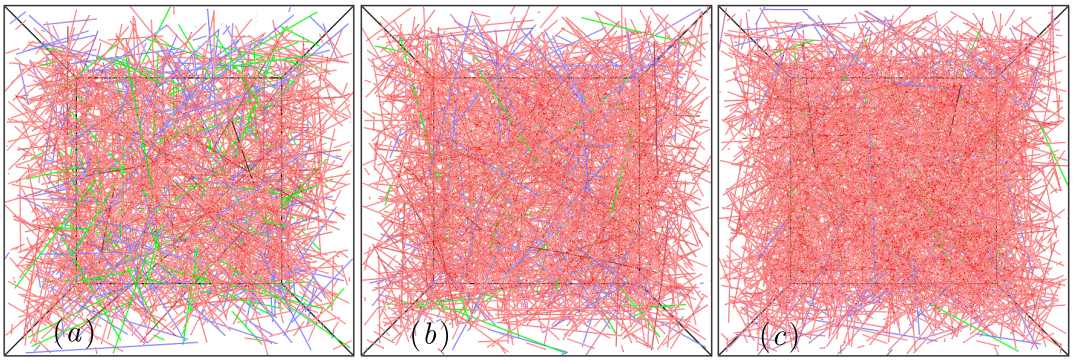


Figure 6.29: Network visualisations for (a) 1200, (b) 1600, and (c) 2000 fibres respectively. The fibres are coloured on a scale of red to blue for bending to stretching dominated energy respectively, and the intensity of the colour represents the relative energy contribution of the fibre. Green fibres signify unstressed fibres with zero energy.

6.4.3 Conditioning of Random Architectures

Randomly seeded network generations display similar characteristics at the macroscopic level but may have large variances in local network geometry that are difficult to quantify. In a step towards a better quantitative understanding of our networks, we can calculate condition number estimates for varied values of our three basic connectivity constraints:

- $\xi_{\min}$ - the minimum permitted distance between fibres for connectivity to occur,
- $\xi_{\max}$ - the maximum permitted distance between fibres for connectivity to occur,
- $\ell_c^{\min}$ - the minimum permitted distance between nodes along a common fibre.

Table 6.4 shows the condition number estimates of systems representing networks of varied connectivity constraints for a low fibre density network of 1200 fibres in a $2\ell \times 2\ell \times 2\ell$ box. The first thing we note from this table is how badly conditioned the global Hessian matrices are that result from the variations of a 1200 fibre network. We see that very broadly, the best conditioned matrices seem to result from the smallest minimum permitted fibre distance paired with the largest maximum permitted fibre distance, with the lowest condition number occurring for the largest choice of minimum distance between two nodes along a fibre. One possible solution for improving the condition of the resultant matrices could be to employ preconditioning (see Section 7.1).

6.4.4 Elastic Medium Embedding

The fibre networks generated for the contents of this thesis have essentially been modelled as though they are in a vacuum. This poses no real issues for networks far above the rigidity transition, but for networks containing floppy modes we find situations where no unique solutions

$\xi^{\min}$	$\xi^{\max}$	$\ell_c^{\min}$	$\kappa(H)$
1.00e-06	2.00e-03	2.00e-03	2.30e+51
		6.00e-03	3.23e+50
		1.00e-02	1.04e+51
1.00e-06	6.00e-03	2.00e-03	9.08e+33
		6.00e-03	2.33e+33
		1.00e-02	2.10e+33
1.00e-06	1.00e-02	2.00e-03	1.23e+19
		6.00e-03	1.57e+19
		1.00e-02	3.11e+18
2.00e-03	6.00e-03	2.00e-03	2.04e+49
		6.00e-03	7.05e+50
		1.00e-02	5.73e+49
2.00e-03	1.00e-02	2.00e-03	1.41e+20
		6.00e-03	1.67e+19
		1.00e-02	5.39e+23
6.00e-03	1.00e-02	2.00e-03	3.04e+38
		6.00e-03	6.78e+38
		1.00e-02	2.28e+39

Table 6.4: A table to show the variation in condition number, κ, (calculated using the PETSc SVD routine) for systems representing networks of varied connectivity constraints for 1200 fibres in a $2\ell \times 2\ell \times 2\ell$ box.

can be found. A potential solution to this problem was discussed in Section 4.3.1, where the fibre network is embedded in an elastic medium much weaker than the original network. Although described in the context of the time stepping approach to solving the global system, this technique of embedding a network in a medium can also be used when solving the corresponding system using direct methods. Once the global Hessian has been updated to $H \to H + \lambda_{EM} I$ [from (4.31)] and the global RHS vector to $\mathbf{B}^* \to \mathbf{B}^* + \lambda_{EM}\mathbf{U}^{\text{aff}}$ [from (4.36)], the system is solved as before using one of the existing PETSc direct methods. To verify that our elastic medium approach behaves as expected, we can compare the results of a simple toy problem such as the λ-network in 3D (accounting for both fibre stretching and bending), and measure the calculated total energy for a varied elastic constant λ_{EM}. Table 6.5 shows how the relative change increases with λ_{EM}, which matches our expectation that increasing the contribution of λ_{EM} will make no significant difference for a weak medium (λ_{EM} small), but there will then be a larger divergence as λ_{EM} increases.

We can then apply elastic medium modelling to find solutions to problems that were not previously solvable, such as the same λ-network where bending is no longer accounted for. In this case as discussed in Section 6.2.2, the direct solver fails due to the zero value in the diagonal of Hessian H. However, embedding this network in an elastic medium we can increase the value of the elastic constant until we achieve a large enough contribution from the embedded medium that a solution can be found. Table 6.6 shows the calculated total energy for λ-network with

elastic constant (λ_{EM})	relative difference (ϵ)
10^{-16}	1.163631e-16
10^{-15}	0
10^{-14}	2.327263e-16
10^{-13}	8.145419e-16
10^{-12}	1.015850e-13
10^{-11}	1.013220e-11
10^{-10}	1.013119e-09
10^{-9}	1.012298e-07
10^{-8}	1.004149e-05
10^{-7}	9.277988e-04
10^{-6}	4.818023e-02

Table 6.5: A table to show how the relative difference, $\epsilon = (E_{EM} - E)/E$, between the standard and elastic medium total energy prediction, varies with the size of the spring constant, λ_{EM}, of the elastic background medium.

fibres modelled without bending for varied λ_{EM}. We see that for this example, it is possible to attain a solution for a sufficiently large value of λ_{EM}, greater than 10^{-14}, and that this solution does not significantly depend on the value of λ_{EM} for the range considered.

elastic constant (λ_{EM})	E
10^{-16}	-
10^{-15}	-
10^{-14}	-
10^{-13}	1.110721e-10
10^{-12}	1.110721e-10
10^{-11}	1.110721e-10
10^{-10}	1.110721e-10
10^{-9}	1.110721e-10
10^{-8}	1.110732e-10
10^{-7}	1.111751e-10
10^{-6}	1.164236e-10

Table 6.6: A table to show the predictions for the total network energy, E, of a stretch-only λ-network embedded in an elastic medium, for varied value of elastic medium constant, λ_{EM}.

6.4.5 Time Stepping Discussion

As discussed in Section 4.3, time stepping is another approach for handling the solution of systems that result from networks with problematic geometries. As in Section 6.4.4 above, we can take the standard λ-network in 3D, and use it to verify that our time stepping implementation behaves as expected. Table 6.7 shows the relative error of the total energy of the λ-network for a fixed α, and varied stopping tolerance τ with the number of steps needed to reach the respective stopping tolerance. We also see the calculation of the norm of the error between

τ	steps	relative error (ϵ) of E	$\|\|\mathbf{U}^{\text{err}}\|\|$
10^{-16}	83386	2.327263e-16	6.36422e-11
10^{-15}	73022	4.072709e-15	6.36405e-10
10^{-14}	62658	4.049437e-13	6.36388e-09
10^{-13}	52294	4.049681e-11	6.36371e-08
10^{-12}	41930	4.049473e-09	6.36355e-07

Table 6.7: A table showing the number of time steps required to reach the required stopping tolerance τ with the final calculated relative error ϵ of E, and $\alpha = 100$. The norm $||\mathbf{U}^{\text{err}}||$ is also given.

the solution vector obtained using time stepping and solution vector obtained using direct LU, where we write this norm as

$$||\mathbf{U}^{\text{err}}|| = \sqrt{\sum_i |\mathbf{U}_i^{\text{TS}} - \mathbf{U}_i^{\text{LU}}|^2}. \tag{6.42}$$

This matches our expectations that a smaller stopping tolerance, τ, results in smaller values for the relative error, ϵ, and norm, $||\mathbf{U}^{\text{err}}||$, in addition to an increase in the number of required time steps.

Applying this procedure to the underconstrained stretch-only case, we are able to obtain predictions for the total network energy (that we would be unable to obtain through standard static solving). Table 6.8 shows these predictions for varied tolerance, demonstrating equal estimates for the range of stopping tolerances considered, accurate up to the number of significant figures presented here.

τ	steps	E
10^{-16}	83386	1.1107207e-10
10^{-15}	73022	1.1107207e-10
10^{-14}	62658	1.1107207e-10
10^{-13}	52294	1.1107207e-10
10^{-12}	41930	1.1107207e-10

Table 6.8: A table showing the prediction of the total energy, E, of a stretch-only λ-network, obtained through time stepping, for varied stopping tolerance, τ, and fixed $\alpha = 100$.

Summary

Building on the 2D network modelling covered in Chapter 5, we have extended the discussion of our model into 3D networks. We began with a description of the additional challenges introduced when modelling 3D architectures, and proposed two different approaches for cross-linking fibres. We then motivated and validated our modelled 3D networks with a collection of toy problems. The primary focus of 3D work has been layered disordered networks. Our

investigation began with an outline of sheet network specific cross-linking, before moving to an analysis of the mechanical properties of such networks. As part of this analysis we have investigated the effects of material anisotropy and presented evidence for an existing relationship between yz and xz oriented networks. Using our work on the energy scaling to support our argument, we have concluded with a discussion of the observed trends. Following the work on layered random networks, we have presented some preliminary results on fully 3D random networks, and concluded with some proposed methods that could be utilised in future work to aid in the solving of systems resulting from problematic network geometries.

Chapter 7

Discussion

7.1 Further Work

Here we specify some of the open directions in which further work could be undertaken if the project were to be extended beyond the submission of this thesis. Some of the suggestions given here would require considerable time and research, while some others are more immediate possibilities that have been omitted from the thesis due to the time allowances of the standard PhD period. Discussion can be broadly split into application focused work, performance focused work and source code dissemination.

7.1.1 Application Focused

Beyond the results presented in Chapter 6, there is additional investigation that could be carried out on generalised fibre networks in relation to network geometry and anisotropy. Taking needlepunched nonwoven fabrics as an initial application, work could then extend to a deeper look at properties of fully 3D networks such as the Mikado networks explored in Section 6.4.

Of the three sheet cross-linking strategies discussed in Section 6.3.2, we only considered the map based approach. It follows that a next step would be to consider modifications of the other two approaches discussed and investigate the effects of anisotropy on the resulting network architectures (noting that unmodified implementations of the other cross-linking strategies would result in fibres perpendicular to the sheets that would not contribute to the mechanics).

Towards the end of the project, contact was established with the University of Leeds Nonwovens Research Group, as well as with a related spin off company NIRI (Nonwovens Innovation & Research Institute)[64, 65]. Both parties expressed an interest in the model we have developed, and the Nonwovens Research Group offered to provide experimental data to be used as parameters for our model if a future collaboration were to be formed, but discussions did not progress beyond this point due to time constraints. Such a collaboration could involve using the model to predict the mechanical properties of materials that are under development and as

a predictive tool for design of new materials.

Our model allows us to predict the response of a fibrous network under an imposed linear shear. Network response to shear can be used to determine the softness of the material it represents [75], which can accelerate the development of new materials by quickly determining in simulations which types of material are likely to have the desired mechanical properties. Additional work could also involve modifying the numerical model and code to predict the response of a given network to macroscopically imposed uniaxial strain.

The isotropy of nonwoven materials is affected by the manufacturing process and in practice most real-world nonwoven materials are anisotropic (see Section 1.1.1). While we were able to investigate the overall network scale anisotropy by considering different shear directions, intrasheet fibre network anisotropy could be investigated in further work through modification of the fibre orientation distribution (similar to the work of Missel et al.[60] on 2D random fibre networks).

Section 6.4 discusses some preliminary work on predicting the mechanical properties of random 3D Mikado networks. Extensions to this work could include a deeper investigation of the bending to stretching crossover, a quantitative comparison of cross-linking via connective fibres and cross-linking via node coupling, embedding networks close to the bending rigidity threshold (or stretch-only networks close to the isostatic threshold) in an elastic medium, and increasing network dimensions to reach representative volumes of fibres.

Finally a longer term extension to the project could look at expanding the existing linear model to a nonlinear regime. Related work has shown that fibrous semiflexible materials can exhibit interesting nonlinear behaviours such as strain stiffening [72, 80, 89]. Thus a nonlinear extension to the model would allow for such behaviours to be reproduced and predicted. Solving this system numerically could involve a Newton-Raphson type solver, in which we would repeatedly solve a linear problem and update the node positions, re-linearising after each repetition. We would use parts of our existing model to solve the linear system and update the node positions, but the assembly of the global Hessian would need to be extended to account for fibre prestress.

7.1.2 Performance Focused

Performance focused further work can be motivated by the lack of models accounting for a representative volume of fully 3D randomly oriented fibres. To get an idea of the expected run time of a representative volume of randomly oriented fibres in 3D, we observe that the largest solved system resulting from a 10 sheet network with 500 fibres per sheet had $\approx 3.5 \times 10^5$ degrees of freedom and took between 24-48 hours to solve the system directly for all 3 shear directions. Further work on the optimisation of the 3D linear solver would initially focus on preconditioning as per the 2D results presented in Section 5.4.2. In this work, we would probably use 3D Mikado network systems to compare the performance of implemented block preconditioners against existing solvers, as they are more suitable as a baseline geometry due

to their maximal symmetry that allows for a clearer overview of the mechanics. Solve times could be recorded using PETSc tools and existing model functionality, and through further customisation model bottlenecks could be identified. Beyond the preconditioning work, there would be additional opportunity to improve the performance through parallelisation and to continue the investigation of how specific geometric features of networks affect the condition number of the resulting global Hessian. As mentioned previously, condition numbers can be calculated using the PETSc implementation of singular value decomposition (SVD). In addition, PETSc also supports parallelisation and interfaces with other libraries to reduce the friction of moving from a serial code base. However, if this work were to be done, some additional thought would also need to be given to how the physical network can be partitioned, and how to translate this into the numerical model.

7.1.3 Supporting the Community

An ongoing motivation for this project has been to develop our predictive model into an open source piece of software that can be used as a tool by researchers from related disciplines to provide insight into their own fibre network architectures. While the code is publicly available [35], and has been used to successfully collect the results discussed in prior chapters, there is further development that could be done to increase the accessibility and efficiency of the software in its current form. Current documentation could be expanded, command line options could be extended to include more of the existing inner code features, efficiency could be improved through code refactoring and by interfacing with additional PETSc functionality. In particular, preconditioning code previously implemented in MATLAB could be ported and extended, and the implementation of the global matrix assembly could be parallelised using the PETSc interface with ParMETIS. With regards to testing, regression tests could be added to, unit and integration tests could be introduced, and hand validation work could be automated as part of the build process. Further automation could include shorter tests to be automatically run as part of the version control system when changes are made, and longer tests to run weekly on the Leeds Advanced Research Computing system.

Aside from the C code that contains the numerical model, there is also the C++ code, responsible for generating and visualising the majority of the networks discussed in prior chapters. This code was initially developed by Jake Horsfield and can be found at [33], whereas the modifications made by the candidate can be found in a forked repository at [34]. Further development would involve some code restructuring to remove the strong dependencies that exist between the `box` , `fibre` , and `node` objects. These changes would be primarily to improve further code maintainance and testing. Additional work would involve merging standalone lattice network generation routines (written in Python) into the C++ code. This would give users of the model an extended choice of possible network geometries that they could use as a starting point for more specific network architectures.

7.2 Conclusions

Utilising existing software packages rather than developing a new piece of software offers the advantage of being able to invest more time in theoretical developments, but depending on individual research needs may offer less flexibility in model design. The results and analysis presented in this thesis were collected using a predictive numerical model that takes the form of a piece of standalone software written by the candidate (see Appendix C). The software takes as input a network in the form of a data file, assembles a global system (from the relevant contributions of a series of local systems), and then solves the system, writing out the updated network data and using the calculated node displacements to predict various mechanical properties. The model infers whether the network is coupled and uses this to determine how to assemble the global system. The software has been developed to be compatible with the PETSc library, such that solver types can be interchanged easily and decided by passing the relevant arguments at run time. The condition number of the global matrix can also be requested at run time by passing the relevant flags. This can either be done by using the PETSc implementation of SVD for small matrices, or can be approximated using estimates of the eigenvalues. All condition numbers presented that were collected with the PETSc model were calculated using SVD. Other run time features that have been developed include the option to embed a given network in an elastic medium, and the option to solve the system using (pseudo-)time stepping. The current PETSc compatible model implemented in C is the product of earlier model iterations, which include a series of Python prototypes and a MATLAB/C hybrid model. Some of the earlier results presented in Chapter 5 were collected with earlier model iterations such as these.

As mentioned in Section 7.1 above, a C++ code was used for network generation and visualisation, and was developed concurrently with the numerical C model. Modifications to the original code have been focused on the 3D development (e.g. network generators, cross-linking algorithms and node coupling), but have also included some changes to the 2D network generation, visualisation, and general code structure (e.g. periodicity support, additional network generators and energy based fibre colouring schemes). In addition to the C model and C++ network generator/visualiser, several pieces of supporting code were written, including bash scripts for automating workflows, testing and job submission, and python scripts for data analysis, toy network validation and lattice network generation.

We have provided a performance comparison of preconditioner choices for both triangular meshes (undiluted and diluted) and random 2D Mikado networks. From this performance analysis, we have found that the systems resulting from 2D networks such as these respond well to some simple choices of preconditioner. Preconditioner choices were motivated by a $x, x, \ldots, y, y, \ldots, z, z, \ldots$ ordering of the global system which exposes a clear block structure that can be exploited. While it is not known without further experimentation whether the effectiveness of the block preconditioners is immediately applicable to 3D networks, the global 3D Hessian matrices can also be ordered to reveal an analogous block structure, presenting the opportunity for investigation with little additional time investment. Furthermore, the simple

design of the preconditioners

$$P_{Db} = \begin{bmatrix} H_{xx} & O \\ O & H_{yy} \end{bmatrix}, \quad P_{Ab} = \begin{bmatrix} H_{xx} & H_{xy} \\ O & H_{yy} \end{bmatrix}, \tag{7.1}$$

depends only on the existing entries of the Hessian and does not require any additional computation to assemble [contrary to P_{AS} - see (4.22)]. This makes them low cost and relatively straightforward to add to existing custom codes written in high-level languages such as Python and MATLAB.

We have presented a fibre network model of needlepunch-inspired layered fibrous material that accounts for the stretching and bending behaviour of discretised intrasheet fibres, and the stretching behaviour of intersheet cross-linking fibres. Using this model, we investigated the effects of material anisotropy by generating predictive mechanical properties for the networks under different possible directions of imposed linear shear in 3D. This investigation led to the identification of some trends that can be described by the competition between intrasheet and intersheet energies, and to the establishment of a clear relationship between the xz and yz shear directions. We supported our findings with a scaling analysis of the network stretching energy. While it is clear that our map based intersheet cross-linking strategy is one of many possible approaches (see Section 6.3.2), it can be argued that modelling sheet to sheet bonds as elastic rods with freely rotating end points is more suitable than fixed single point bonds, since real-world needlepunched fabrics have flexible bonds created by fibre entanglements that allow some fibre movement [74, p. 403]. These findings provide a first understanding of how the microscopic mechanics of mechanically bonded layers of elastic fibres contribute to the macroscopic mechanical behaviour of the overall material. This can be used as the starting point of a collaboration with researchers from the nonwovens industry, that could include aiding the design of new materials or modifying existing materials.

While we have latterly focused on the application of layered anisotropic nonwoven fabrics, the underlying numerical model is not constrained to these respective anisotropic network architectures. As we have seen from the preliminary work on random isotropic 3D networks (see Section 6.4), the software that has been developed (along with the underlying numerical model) provides a foundation for further research and future directions. Furthermore, we have introduced and implemented two distinct approaches for cross-linking a given pair of nodes. Both the network generator and numerical model have been developed to be compatible with either approach. Having both connecting fibre cross-links and node coupled cross-links provides additional network design opportunities and motivates future work that could give a quantitative understanding of how the two approaches affect the network mechanics. Both of our cross-linking approaches use a minimum distance calculation to determine which fibres should be connected. This differs from many related works that introduce attractive forces or Monte Carlo methods to reorganise fibre positions and create binary cross-links [37, 38, 39, 40, 95]. Instead our fibres are cross-linked in their original positions and orientations similar to the

work of Astrom et al.[94], although the node coupling approach fixes the distance between node couples, such that there is no connecting fibre between points. Moreover, our cross-linking fibre approach does not connect fibres between predetermined and evenly spaced nodes, but rather between two points of true minimum distance (subject to a small tolerance). These differences distinguish our isotropic random 3D networks from existing 3D geometries, and present an opportunity to address open questions on the mechanics of such networks in future work.

Appendix A

System Assembly

A.1 Local Bending Matrix

Recall from Section 3.4 that the local bending matrix can be expressed as a matrix of sub-blocks

$$H_{ijk}^{\text{bend}} = \frac{2\kappa_{ijk}}{\ell_{ij}^2 \ell_{jk}^2 \left(\ell_{ij} + \ell_{jk}\right)} \begin{bmatrix} H_{ijk_{xx}}^{\text{bend}} & H_{ijk_{xy}}^{\text{bend}} & H_{ijk_{xz}}^{\text{bend}} \\ H_{ijk_{yx}}^{\text{bend}} & H_{ijk_{yy}}^{\text{bend}} & H_{ijk_{yz}}^{\text{bend}} \\ H_{ijk_{zx}}^{\text{bend}} & H_{ijk_{zy}}^{\text{bend}} & H_{ijk_{zz}}^{\text{bend}} \end{bmatrix}. \tag{A.1}$$

The sub-blocks of (A.1) are defined in this section, beginning with the sub-blocks for the rows corresponding to x components:

$$H_{ijk_{xx}}^{\text{bend}} = \begin{bmatrix} s_{jk_y}^2 + s_{jk_z}^2 & -s_{ik_y}s_{jk_y} - s_{ik_z}s_{jk_z} & s_{ij_y}s_{jk_y} + s_{ij_z}s_{jk_z} \\ -s_{ik_y}s_{jk_y} - s_{ik_z}s_{jk_z} & s_{ik_y}^2 + s_{ik_z}^2 & -s_{ik_y}s_{ij_y} - s_{ik_z}s_{ij_z} \\ s_{ij_y}s_{jk_y} + s_{ij_z}s_{jk_z} & -s_{ik_y}s_{ij_y} - s_{ik_z}s_{ij_z} & s_{ij_y}^2 + s_{ij_z}^2 \end{bmatrix}, \tag{A.2}$$

$$H_{ijk_{xy}}^{\text{bend}} = \begin{bmatrix} -s_{jk_x}s_{jk_y} & s_{ik_x}s_{jk_y} & -s_{ij_x}s_{jk_y} \\ s_{ik_y}s_{jk_x} & -s_{ik_x}s_{ik_y} & s_{ik_y}s_{ij_x} \\ -s_{ij_y}s_{jk_x} & s_{ik_x}s_{ij_y} & -s_{ij_x}s_{ij_y} \end{bmatrix}, \tag{A.3}$$

$$H_{ijk_{xz}}^{\text{bend}} = \begin{bmatrix} -s_{jk_x}s_{jk_z} & s_{ik_x}s_{jk_z} & -s_{ij_x}s_{jk_z} \\ s_{ik_z}s_{jk_x} & -s_{ik_x}s_{ik_z} & s_{ik_z}s_{ij_x} \\ -s_{ij_z}s_{jk_x} & s_{ik_x}s_{ij_z} & -s_{ij_x}s_{ij_z} \end{bmatrix}. \tag{A.4}$$

Then the sub-blocks for the rows corresponding to y components are:

$$H_{ijk_{yx}}^{\text{bend}} = \begin{bmatrix} -s_{jk_x}s_{jk_y} & s_{ik_y}s_{jk_x} & -s_{ij_y}s_{jk_x} \\ s_{ik_x}s_{jk_y} & -s_{ik_x}s_{ik_y} & s_{ik_x}s_{ij_y} \\ -s_{ij_x}s_{jk_y} & s_{ik_y}s_{ij_x} & -s_{ij_x}s_{ij_y} \end{bmatrix}, \tag{A.5}$$

$$H^{\text{bend}}_{ijk_{yy}} = \begin{bmatrix} s^2_{jk_x}+s^2_{jk_z} & -s_{ik_x}s_{jk_x}-s_{ik_z}s_{jk_z} & s_{ij_x}s_{jk_x}+s_{ij_z}s_{jk_z} \\ -s_{ik_x}s_{jk_x}-s_{ik_z}s_{jk_z} & s^2_{ik_x}+s^2_{ik_z} & -s_{ik_x}s_{ij_x}-s_{ik_z}s_{ij_z} \\ s_{ij_x}s_{jk_x}+s_{ij_z}s_{jk_z} & -s_{ik_x}s_{ij_x}-s_{ik_z}s_{ij_z} & s^2_{ij_x}+s^2_{ij_z} \end{bmatrix}, \tag{A.6}$$

$$H^{\text{bend}}_{ijk_{yz}} = \begin{bmatrix} -s_{jk_y}s_{jk_z} & s_{ik_y}s_{jk_z} & -s_{ij_y}s_{jk_z} \\ s_{ik_z}s_{jk_y} & -s_{ik_y}s_{ik_z} & s_{ik_z}s_{ij_y} \\ -s_{ij_z}s_{jk_y} & s_{ik_y}s_{ij_z} & -s_{ij_y}s_{ij_z} \end{bmatrix}. \tag{A.7}$$

Finally the sub-blocks for the rows corresponding to z components are:

$$H^{\text{bend}}_{ijk_{zx}} = \begin{bmatrix} -s_{jk_x}s_{jk_z} & s_{ik_z}s_{jk_x} & -s_{ij_z}s_{jk_x} \\ s_{ik_x}s_{jk_z} & -s_{ik_x}s_{ik_z} & s_{ik_x}s_{ij_z} \\ -s_{ij_x}s_{jk_z} & s_{ik_z}s_{ij_x} & -s_{ij_x}s_{ij_z} \end{bmatrix}, \tag{A.8}$$

$$H^{\text{bend}}_{ijk_{zy}} = \begin{bmatrix} -s_{jk_y}s_{jk_z} & s_{ik_z}s_{jk_y} & -s_{ij_z}s_{jk_y} \\ s_{ik_y}s_{jk_z} & -s_{ik_y}s_{ik_z} & s_{ik_y}s_{ij_z} \\ -s_{ij_y}s_{jk_z} & s_{ik_z}s_{ij_y} & -s_{ij_y}s_{ij_z} \end{bmatrix}, \tag{A.9}$$

$$H^{\text{bend}}_{ijk_{zz}} = \begin{bmatrix} s^2_{jk_x}+s^2_{jk_y} & -s_{ik_x}s_{jk_x}-s_{ik_y}s_{jk_y} & s_{ij_x}s_{jk_x}+s_{ij_y}s_{jk_y} \\ -s_{ik_x}s_{jk_x}-s_{ik_y}s_{jk_y} & s^2_{ik_x}+s^2_{ik_y} & -s_{ik_x}s_{ij_x}-s_{ik_y}s_{ij_y} \\ s_{ij_x}s_{jk_x}+s_{ij_y}s_{jk_y} & -s_{ik_x}s_{ij_x}-s_{ik_y}s_{ij_y} & s^2_{ij_x}+s^2_{ij_y} \end{bmatrix}. \tag{A.10}$$

A.2 Local Bending RHS Vector

Recall from Section 3.4 that the local bending RHS vector was expressed in the shorthand form

$$\mathbf{B}^{\text{bend}}_{ijk} = \frac{2\kappa_{ijk}}{\ell^2_{ij}\ell^2_{jk}\left(\ell_{ij}+\ell_{jk}\right)} \begin{bmatrix} s_{jk_y}\beta_1^{i_x} + s_{jk_z}\beta_2^{i_x} \\ -s_{ik_y}\beta_1^{j_x} - s_{ik_z}\beta_2^{j_x} \\ s_{ij_y}\beta_1^{k_x} + s_{ij_z}\beta_2^{k_x} \\ -s_{jk_x}\beta_1^{i_y} + s_{jk_z}\beta_2^{i_y} \\ s_{ik_x}\beta_1^{j_y} - s_{ik_z}\beta_2^{j_y} \\ -s_{ij_x}\beta_1^{k_y} + s_{ij_z}\beta_2^{k_y} \\ -s_{jk_x}\beta_1^{i_z} - s_{jk_y}\beta_2^{i_z} \\ s_{ik_x}\beta_1^{j_z} + s_{ik_y}\beta_2^{j_z} \\ -s_{ij_x}\beta_1^{k_z} - s_{ij_y}\beta_2^{k_z} \end{bmatrix}. \tag{A.11}$$

The shorthand notation used in (A.11) is defined in this section. For the rows corresponding to x components we write

$$\begin{aligned}
\beta_1^{i_x} &= -s_{ij_x}u_{jk_y} + s_{ij_y}u_{jk_x} + s_{jk_x}u_{ij_y} - s_{jk_y}u_{ij_x}, \\
\beta_2^{i_x} &= -s_{ij_x}u_{jk_z} + s_{ij_z}u_{jk_x} + s_{jk_x}u_{ij_z} - s_{jk_z}u_{ij_x}, \\
\beta_1^{j_x} &= -s_{ij_x}u_{jk_y} + s_{ij_y}u_{jk_x} + s_{jk_x}u_{ij_y} - s_{jk_y}u_{ij_x}, \\
\beta_2^{j_x} &= -s_{ij_x}u_{jk_z} + s_{ij_z}u_{jk_x} + s_{jk_x}u_{ij_z} - s_{jk_z}u_{ij_x}, \\
\beta_1^{k_x} &= -s_{ij_x}u_{jk_y} + s_{ij_y}u_{jk_x} + s_{jk_x}u_{ij_y} - s_{jk_y}u_{ij_x}, \\
\beta_2^{k_x} &= -s_{ij_x}u_{jk_z} + s_{ij_z}u_{jk_x} + s_{jk_x}u_{ij_z} - s_{jk_z}u_{ij_x}.
\end{aligned} \tag{A.12}$$

For the rows corresponding to y components we write

$$\begin{aligned}
\beta_1^{i_y} &= -s_{ij_x}u_{jk_y} + s_{ij_y}u_{jk_x} + s_{jk_x}u_{ij_y} - s_{jk_y}u_{ij_x}, \\
\beta_2^{i_y} &= -s_{ij_y}u_{jk_z} + s_{ij_z}u_{jk_y} + s_{jk_y}u_{ij_z} - s_{jk_z}u_{ij_y}, \\
\beta_1^{j_y} &= -s_{ij_x}u_{jk_y} + s_{ij_y}u_{jk_x} + s_{jk_x}u_{ij_y} - s_{jk_y}u_{ij_x}, \\
\beta_2^{j_y} &= -s_{ij_y}u_{jk_z} + s_{ij_z}u_{jk_y} + s_{jk_y}u_{ij_z} - s_{jk_z}u_{ij_y}, \\
\beta_1^{k_y} &= -s_{ij_x}u_{jk_y} + s_{ij_y}u_{jk_x} + s_{jk_x}u_{ij_y} - s_{jk_y}u_{ij_x}, \\
\beta_2^{k_y} &= -s_{ij_y}u_{jk_z} + s_{ij_z}u_{jk_y} + s_{jk_y}u_{ij_z} - s_{jk_z}u_{ij_y}.
\end{aligned} \tag{A.13}$$

For the rows corresponding to z components we write

$$\begin{aligned}
\beta_1^{i_z} &= -s_{ij_x}u_{jk_z} + s_{ij_z}u_{jk_x} + s_{jk_x}u_{ij_z} - s_{jk_z}u_{ij_x}, \\
\beta_2^{i_z} &= -s_{ij_y}u_{jk_z} + s_{ij_z}u_{jk_y} + s_{jk_y}u_{ij_z} - s_{jk_z}u_{ij_y}, \\
\beta_1^{j_z} &= -s_{ij_x}u_{jk_z} + s_{ij_z}u_{jk_x} + s_{jk_x}u_{ij_z} - s_{jk_z}u_{ij_x}, \\
\beta_2^{j_z} &= -s_{ij_y}u_{jk_z} + s_{ij_z}u_{jk_y} + s_{jk_y}u_{ij_z} - s_{jk_z}u_{ij_y}, \\
\beta_1^{k_z} &= -s_{ij_x}u_{jk_z} + s_{ij_z}u_{jk_x} + s_{jk_x}u_{ij_z} - s_{jk_z}u_{ij_x}, \\
\beta_2^{k_z} &= -s_{ij_y}u_{jk_z} + s_{ij_z}u_{jk_y} + s_{jk_y}u_{ij_z} - s_{jk_z}u_{ij_y}.
\end{aligned} \tag{A.14}$$

A.3 Example Network Sparsity Calculation

To calculate the CSR compatible sparsity pattern of the complete Hessian matrix represented by the network seen in Figure A.1, we begin by identifying the fibres associated with each

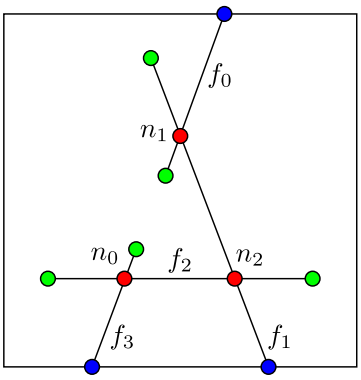


Figure A.1: An example network for calculating the sparsity pattern.

internal node

$$\underset{\texttt{count}}{\begin{bmatrix}2\\2\\2\end{bmatrix}}, \quad \underset{\texttt{rowp}}{\begin{bmatrix}0\\2\\4\\6\end{bmatrix}}, \quad \underset{\texttt{col}}{\begin{bmatrix}\mathrm{ID}(f_2)\\ \mathrm{ID}(f_3)\\ \mathrm{ID}(f_1)\\ \mathrm{ID}(f_2)\\ \mathrm{ID}(f_0)\\ \mathrm{ID}(f_1)\end{bmatrix}}, \quad \underset{\texttt{val}}{\begin{bmatrix}\mathrm{value}(0,\mathrm{ID}(f_2))\\ \mathrm{value}(0,\mathrm{ID}(f_3))\\ \mathrm{value}(1,\mathrm{ID}(f_1))\\ \mathrm{value}(1,\mathrm{ID}(f_2))\\ \mathrm{value}(2,\mathrm{ID}(f_0))\\ \mathrm{value}(2,\mathrm{ID}(f_1))\end{bmatrix}}, \tag{A.15}$$

and then continue by calculating the respective node associations

$$\underset{\texttt{count}}{\begin{bmatrix}2\\2\\3\end{bmatrix}}, \quad \underset{\texttt{rowp}}{\begin{bmatrix}0\\2\\4\\7\end{bmatrix}}, \quad \underset{\texttt{col}}{\begin{bmatrix}\mathrm{ID}(n_0)\\ \mathrm{ID}(n_2)\\ \mathrm{ID}(n_1)\\ \mathrm{ID}(n_2)\\ \mathrm{ID}(n_0)\\ \mathrm{ID}(n_1)\\ \mathrm{ID}(n_2)\end{bmatrix}}, \quad \underset{\texttt{val}}{\begin{bmatrix}\mathrm{value}(0,\mathrm{ID}(n_0))\\ \mathrm{value}(0,\mathrm{ID}(n_2))\\ \mathrm{value}(1,\mathrm{ID}(n_1))\\ \mathrm{value}(1,\mathrm{ID}(n_2))\\ \mathrm{value}(2,\mathrm{ID}(n_0))\\ \mathrm{value}(2,\mathrm{ID}(n_1))\\ \mathrm{value}(2,\mathrm{ID}(n_2))\end{bmatrix}}. \tag{A.16}$$

The sparsity calculated thus far only represents the sparsity of a single block of the Hessian. To calculate the full sparsity whilst preserving the CSR format, further work needs to be done to copy the indices successfully.

Since each internal node ID stored in the `col` array corresponds to a column index of a specific row, it is important to also update the values of the node IDs that correspond to the central H_{py} and right H_{pz} blocks, for $p = \{x, y, z\}$. So for row 0 of the sparse matrix obtained from the example given in Figure A.1, the indices of the H_{xy} block should be updated to

$$\mathrm{ID}(n_0) + N_{int}, \quad \mathrm{ID}(n_2) + N_{int}, \tag{A.17}$$

and the indices of the H_{xz} block should be updated to

$$\mathrm{ID}(n_0) + 2N_{int}, \quad \mathrm{ID}(n_2) + 2N_{int}. \tag{A.18}$$

Applying this procedure to all vector entries the complete sparsity pattern can be given as

$$\begin{array}{cccc} \texttt{count} & \texttt{rowp} & \texttt{col} & \texttt{val} \\ \begin{bmatrix} 2 \\ 2 \\ 3 \\ 2 \\ 2 \\ 3 \\ 2 \\ 2 \\ 3 \end{bmatrix}, & \begin{bmatrix} 0 \\ 6 \\ 12 \\ 21 \\ 27 \\ 33 \\ 42 \\ 48 \\ 54 \\ 63 \end{bmatrix}, & \begin{bmatrix} \mathrm{ID}(n_0) \\ \mathrm{ID}(n_2) \\ \mathrm{ID}(n_0) + N_{int} \\ \mathrm{ID}(n_2) + N_{int} \\ \mathrm{ID}(n_0) + 2N_{int} \\ \mathrm{ID}(n_2) + 2N_{int} \\ \mathrm{ID}(n_1) \\ \mathrm{ID}(n_2) \\ \mathrm{ID}(n_1) + N_{int} \\ \mathrm{ID}(n_2) + N_{int} \\ \mathrm{ID}(n_1) + 2N_{int} \\ \mathrm{ID}(n_2) + 2N_{int} \\ \mathrm{ID}(n_0) \\ \mathrm{ID}(n_1) \\ \mathrm{ID}(n_2) \\ \mathrm{ID}(n_0) + N_{int} \\ \mathrm{ID}(n_1) + N_{int} \\ \mathrm{ID}(n_2) + N_{int} \\ \mathrm{ID}(n_0) + 2N_{int} \\ \mathrm{ID}(n_1) + 2N_{int} \\ \mathrm{ID}(n_2) + 2N_{int} \\ \vdots \end{bmatrix}, & \begin{bmatrix} \mathrm{value}(0, \mathrm{ID}(n_0)) \\ \mathrm{value}(0, \mathrm{ID}(n_2)) \\ \mathrm{value}(0, \mathrm{ID}(n_0) + N_{int}) \\ \mathrm{value}(0, \mathrm{ID}(n_2) + N_{int}) \\ \mathrm{value}(0, \mathrm{ID}(n_0) + 2N_{int}) \\ \mathrm{value}(0, \mathrm{ID}(n_2) + 2N_{int}) \\ \mathrm{value}(1, \mathrm{ID}(n_1)) \\ \mathrm{value}(1, \mathrm{ID}(n_2)) \\ \mathrm{value}(1, \mathrm{ID}(n_1) + N_{int}) \\ \mathrm{value}(1, \mathrm{ID}(n_2) + N_{int}) \\ \mathrm{value}(1, \mathrm{ID}(n_1) + 2N_{int}) \\ \mathrm{value}(1, \mathrm{ID}(n_2) + 2N_{int}) \\ \mathrm{value}(2, \mathrm{ID}(n_0)) \\ \mathrm{value}(2, \mathrm{ID}(n_1)) \\ \mathrm{value}(2, \mathrm{ID}(n_2)) \\ \mathrm{value}(2, \mathrm{ID}(n_0) + N_{int}) \\ \mathrm{value}(2, \mathrm{ID}(n_1) + N_{int}) \\ \mathrm{value}(2, \mathrm{ID}(n_2) + N_{int}) \\ \mathrm{value}(2, \mathrm{ID}(n_0) + 2N_{int}) \\ \mathrm{value}(2, \mathrm{ID}(n_1) + 2N_{int}) \\ \mathrm{value}(2, \mathrm{ID}(n_2) + 2N_{int}) \\ \vdots \end{bmatrix}, \end{array} \tag{A.19}$$

where for brevity `col` and `val` only show entries corresponding to the first 3 rows of the matrix.

After successfully copying the sparsity pattern to each of the Hessians 9 sub-blocks, every slice represents a row spanning the entirety of the global Hessian, such that every 3 consecutive slices of the final `col` array represents 3 consecutive sub-blocks of the Hessian, and every internal node ID contained in the respective slice corresponds to a column index of that respective row of the global Hessian that contains a nonzero element.

Appendix B

Additional Working

B.1 Affine Shear Modulus for a Triangular Mesh

Consider a 2D periodic network of fibres of height H, built from a regular mesh of triangles with sides length ℓ_t [see Figure B.1 *(left)*]. Taking the affine assumption implies that if we

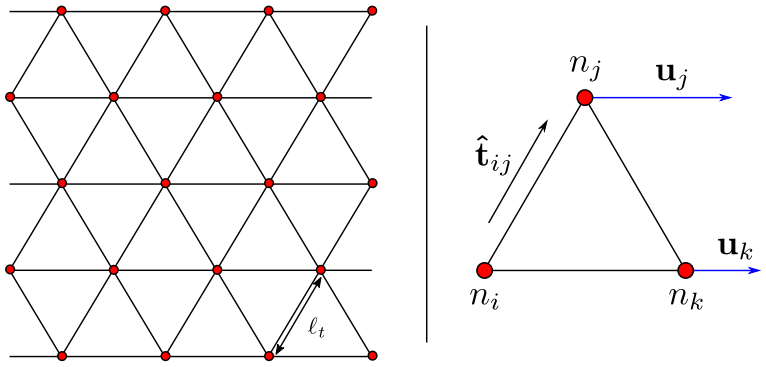


Figure B.1: A segment of a 2D periodic network made up of regular triangles with sides length ℓ_t *(left)* and a schematic of one of these triangles, with a relative shear directed with blue arrows *(right)*.

apply a shear to the upper boundary of the network, such that any node on this boundary is displaced in the x direction by γH, then all other nodes in the network are displaced uniformly in the x direction. We begin by considering a single triangle of the network [see Figure B.1 *(right)*], defined by the position vectors $\mathbf{s}_i, \mathbf{s}_j, \mathbf{s}_k$, of the nodes n_i, n_j, and n_k respectively. The displacements of these nodes are

$$\mathbf{u}_i = (\gamma s_{i_y}, 0)^T,\ \mathbf{u}_j = (\gamma s_{j_y}, 0)^T,\ \mathbf{u}_k = (\gamma s_{k_y}, 0)^T \quad \text{with } s_{i_y} = s_{k_y},$$

respectively. Relative to node n_i, the node n_j has a x displacement of $\gamma(s_{j_y} - s_{i_y}) = \frac{\sqrt{3}}{2}\ell_t\gamma$, where $\frac{\sqrt{3}}{2}\ell_t$ is the height of the triangle. We can see this from the difference in displacements

$$\mathbf{u}_j - \mathbf{u}_i = \begin{bmatrix} \gamma(s_{j_y} - s_{i_y}) \\ 0 \end{bmatrix} = \begin{bmatrix} \gamma\frac{\sqrt{3}}{2}\ell_t \\ 0 \end{bmatrix}. \tag{B.1}$$

Calculating the tangent vector $\hat{\mathbf{t}}_{ij}$

$$\hat{\mathbf{t}}_{ij} = \frac{\Delta\mathbf{s}_{ij}}{\ell_{ij}} = \frac{1}{\ell_t}\begin{bmatrix} \frac{1}{2}\ell_t \\ \frac{\sqrt{3}}{2}\ell_t \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 \\ \sqrt{3} \end{bmatrix}. \tag{B.2}$$

Taking the segment between nodes n_i and n_j as a simple Hookean spring, we can use (3.34) to write the energy of the segment as

$$E_{ij} = \frac{k_t}{2}\left[(\mathbf{u}_j - \mathbf{u}_i)\cdot\hat{\mathbf{t}}_{ij}\right]^2 = \frac{k_t}{2}\left[\frac{\sqrt{3}}{4}\gamma\ell_t\right]^2 = \frac{k_t}{2}\frac{3}{16}\gamma^2\ell_t^2, \tag{B.3}$$

where k_t is the same constant for every spring (since the mesh is regular). Through symmetry, it can be shown that $E_{jk} = E_{ij}$ and $E_{ik} = 0$ follows from $s_{i_y} = s_{k_y}$.

To find the total network energy, we must sum together the energy of each spring of the network. To do this we must determine how to count the springs. Here we argue that since each horizontal spring has zero energy and the mesh is tessellated, we can count the left spring of every triangle in the network. Thus

$$E = \frac{N_t}{2}k_t\gamma^2\ell_t^2\frac{3}{16},$$

for a network with N_t triangles. Thus the area of the network is $A = N_t\frac{\sqrt{3}}{4}\ell_t^2$, since there are N_t triangles and the area of each can be shown to be $\frac{\sqrt{3}}{4}\ell_t^2$. Therefore, referring to (5.34) we find

$$G^{\text{aff}} = \frac{2E}{A\gamma^2}, = \frac{2\frac{N_t}{2}k_t\gamma^2\ell_t^2\frac{3}{16}}{N_t\frac{\sqrt{3}}{4}\ell_t^2\gamma^2} = k_t\frac{\sqrt{3}}{4}, \tag{B.4}$$

as required.

B.2 Linear Extension/Compression

We begin by considering the fibre segment with end points $\mathbf{s}_i$ and $\mathbf{s}_j$. The initial end-to-end vector can be written in terms of the undeformed (resting) length of the segment and the unit tangent vector as

$$\ell_{ij}\hat{\mathbf{t}}_{ij}. \tag{B.5}$$

Then the new end-to-end vector can be written as (B.5) summed with the overall displacement vector, $\Delta\mathbf{u}_{ij}$,

$$\ell_{ij}\hat{\mathbf{t}}_{ij} + \Delta\mathbf{u}_{ij}, \tag{B.6}$$

where the overall displacement vector is written as the difference of the two end point displacement vectors $\Delta\mathbf{u}_{ij} = \mathbf{u}_j - \mathbf{u}_i$.

To derive an expression for the linearised extension/compression, we start by writing the square of the length of the new end-to-end vector. In this way we are able to express its scalar product counterpart and expand

$$\begin{aligned}
\left|\ell_{ij}\hat{\mathbf{t}}_{ij} + \Delta\mathbf{u}_{ij}\right|^2 &= (\ell_{ij}\hat{\mathbf{t}}_{ij} + \Delta\mathbf{u}_{ij}) \cdot (\ell_{ij}\hat{\mathbf{t}}_{ij} + \Delta\mathbf{u}_{ij}), && \text{(B.7)}\\
&= \ell_{ij}^2(\hat{\mathbf{t}}_{ij} \cdot \hat{\mathbf{t}}_{ij}) + 2\ell_{ij}(\Delta\mathbf{u}_{ij} \cdot \hat{\mathbf{t}}_{ij}) + \mathcal{O}(\mathbf{u}^2), && \text{(B.8)}\\
&\approx \ell_{ij}^2 \left\{1 + \frac{2(\Delta\mathbf{u}_{ij} \cdot \hat{\mathbf{t}}_{ij})}{\ell_{ij}}\right\}, && \text{(B.9)}
\end{aligned}$$

where our approximation comes from dropping the nonlinear term. Now we continue by removing the square on the left hand side and approximating the square root on the right hand side

$$\begin{aligned}
\left|\ell_{ij}\hat{\mathbf{t}}_{ij} + \Delta\mathbf{u}_{ij}\right| &= \left[\ell_{ij}^2 \left\{1 + \frac{2(\Delta\mathbf{u}_{ij} \cdot \hat{\mathbf{t}}_{ij})}{\ell_{ij}}\right\}\right]^{1/2}, && \text{(B.10)}\\
&\approx \quad \ell_{ij} \left\{1 + \frac{(\Delta\mathbf{u}_{ij} \cdot \hat{\mathbf{t}}_{ij})}{\ell_{ij}}\right\}, && \text{(B.11)}\\
\ell_{ij} + \delta\ell \quad &\approx \quad \ell_{ij} + \Delta\mathbf{u}_{ij} \cdot \hat{\mathbf{t}}_{ij}. && \text{(B.12)}
\end{aligned}$$

Cancelling ℓ_{ij}, we are left with the expression for linear extension/compression

$$\delta\ell \approx \Delta\mathbf{u}_{ij} \cdot \hat{\mathbf{t}}_{ij}. \tag{B.13}$$

B.3 Dangling Node Updating

The procedure for updating the positions of dangling nodes can be split into two cases: the case where the fibre has only one internal/boundary (i.e. non dangling) node [see Figure B.2 *(left)*], and the case where the fibre has more than one internal/boundary node [see Figure B.2 *(right)*]. Beginning with the former case, dangling nodes n_p, n_q can be updated using the central internal node, n_i, of the fibre. Specifically, if

$$\mathbf{s}_i' = \mathbf{s}_i + \mathbf{u}_i,$$

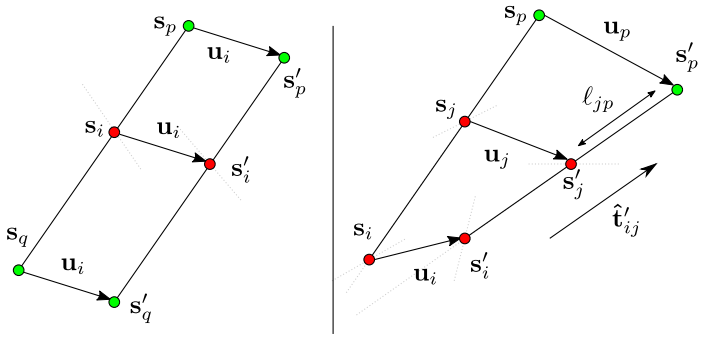


Figure B.2: Dangling nodes that are part of an unconstrained fibre can be updated using the midpoint of the central node *(left)*. Other dangling nodes are updated using the closest two internal/boundary nodes.

is the updated position vector of node n_i, it follows that

$$\mathbf{s}'_p = \mathbf{s}_p + \mathbf{u}_p, \tag{B.14}$$

$$\mathbf{s}'_q = \mathbf{s}_q + \mathbf{u}_q, \tag{B.15}$$

where $\mathbf{u}_p = \mathbf{u}_q = \mathbf{u}_i$. In the latter case, dangling node n_p can be updated using the tangent vector of the updated positions of the two closest nodes n_i, n_j, along the fibre. To elaborate, assume

$$\mathbf{s}'_i = \mathbf{s}_i + \mathbf{u}_i, \tag{B.16}$$

$$\mathbf{s}'_j = \mathbf{s}_j + \mathbf{u}_j, \tag{B.17}$$

then we can define

$$\hat{\mathbf{t}}'_{ij} = \frac{\Delta \mathbf{s}'_{ij}}{\ell'_{ij}} = \frac{\mathbf{s}'_j - \mathbf{s}'_i}{|\mathbf{s}'_j - \mathbf{s}'_i|}. \tag{B.18}$$

Then position vector of n_p can be updated to

$$\mathbf{s}'_p = \mathbf{s}'_j + \ell_{jp}\hat{\mathbf{t}}'_{ij}, \tag{B.19}$$

where ℓ_{jp} is the length of the undisplaced segment between nodes n_j and n_p.

B.4 Cross-link Density of a 2D Network

To derive the cross-link density of a 2D network, we begin with the simple case of two fibres ($N = 2$). Consider two fibres of the same length ℓ, deposited onto a plane with random position and orientation. If we denote the relative angle between the fibre pair as θ, then the region

in which the two fibre midpoints lie (see Figure B.3) defines the region in which the cross-link that connects the fibres can lie. The area of this region seen in Figure B.3 is

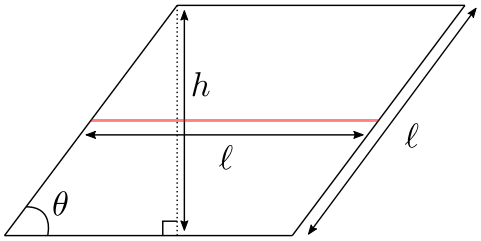


Figure B.3: Given the relative angle θ between a pair of fibres of length ℓ, the area in which a cross-link must lie is a parallelogram found by considering fibre midpoints. For the red fibre seen here, the midpoint of a second fibre must lie somewhere in the parallelogram for the pair to intersect.

$$A_\times = \ell h = \ell^2 |\sin(\theta)|. \tag{B.20}$$

Averaging over θ gives

$$\langle A_\times \rangle = \frac{1}{\pi/2}\ell^2 \int_0^{\pi/2} d\theta \sin\theta = \frac{2\ell^2}{\pi}[-\cos\theta]_0^{\pi/2} = \frac{2\ell^2}{\pi}, \tag{B.21}$$

where range $[0, \pi/2]$ is used to handle the modulus. Therefore, the probability of a randomly-deposited fibre being cross-linked with one other fibre in a plane of area A is

$$p_\times = \frac{A_\times}{A} = \frac{2\ell^2}{A\pi}. \tag{B.22}$$

We note that here boundaries have not yet been considered, so this is only valid for ℓ less than the smallest box dimension for a periodic system, or an infinite box.

For $N > 2$, recall that the total *fibre density* for N fibres is

$$\rho = \frac{N\ell}{A}. \tag{B.23}$$

In terms of (B.23), the expected number of cross-links per fibre for large N is therefore

$$Np_\times = \frac{2N\ell^2}{A\pi} = \frac{2}{\pi}\rho\ell, \tag{B.24}$$

[or $(N-1)p_\times$ for finite N]. Thus the total number of cross-links is

$$\frac{N}{2}\left(\frac{2}{\pi}\rho\ell\right) = \frac{1}{\pi}N\rho\ell = \frac{N^2\ell^2}{\pi A}, \tag{B.25}$$

and the total number of cross-links per unit area, $\langle n_\times \rangle$, is (B.25) divided by A, which is

$$\langle n_\times \rangle = \frac{N^2\ell^2}{\pi A^2} = \frac{\rho^2}{\pi}. \tag{B.26}$$

Appendix C

Code Overview

C.1 Numerical Model Overview

A high-level overview of the components involved in the numerical model implementation is given in Figure C.1. The code was designed to be as modular as far as possible, to reduce the friction of dependency between functions and allow for easier code maintenance. For further

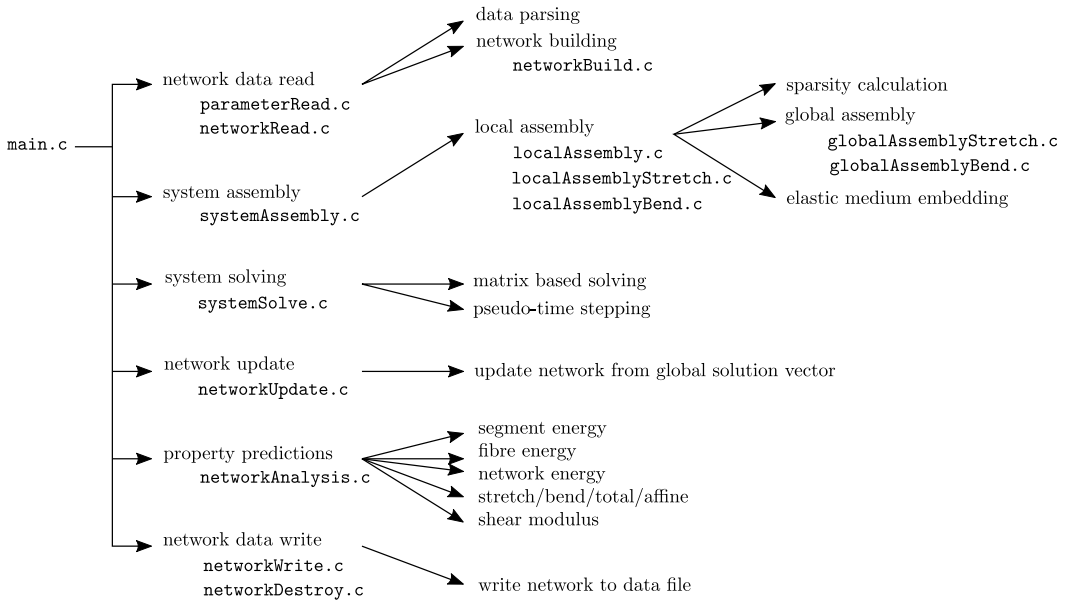
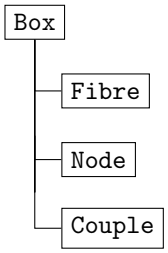


Figure C.1: An overview of numerical model implementation. A main driver function collects together the functionality of the modular components.

details of the code we refer the reader to [35]. In addition for details of the modified network generator/visualiser C++ code refer to [34], or to [33] for the original code.

C.2 Data Structure

As previously discussed in Section 5.1.6, the model assumes that networks are structured in a specific format within the input network data file. After the data is read into the numerical model, the core data types are organised as:



The network generator/visualiser code maintains the same relationship between these data types, but implements the data types as C++ objects rather than C structures.

C.3 Workflow

The general workflow for using the codebase has been summarised in Figure C.2. Depending

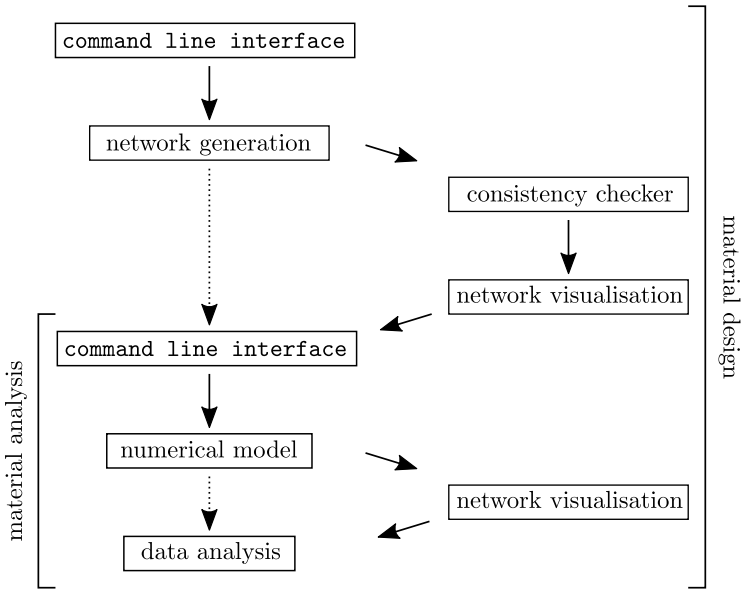


Figure C.2: A diagram to show the high-level workflow for developed software, broken into the network generator and numerical model with optional tools such as network visualisation on the right.

on the individual needs of a user, the typical workflow may differ. A user looking to analyse a specific material may pass their custom network data to the numerical model to generate

mechanical property predictions about their material. This could be followed up with further data analysis on the numerical model output to produce further insight. A user looking to design a new material or manipulate an existing one may have a more in-depth workflow that also involves using our code for network generation. Combined with the use of the numerical model, this could be an iterative procedure of using the output of the numerical model to guide further modifications to the material at the network generation stage. Both of these proposed workflows could be supplemented with the use of the network visualisation code, and network generation can be supplemented with some python scripts that perform basic consistency checks between nodes and fibres.

C.4 Code Usage

The model is compiled using `make`, which results in the executable `model`. Each network data file must be passed to the model via a parameter file, which can be generated automatically using:

```
make parfiles
```

This command will look in each of the existing subdirectories `subdir` of `data/dat/subdir` and build parameter files for all of the network data files with a filename ending in `_in.dat`. To run the model use:

```
mpiexec -n 1 ./model data/par/parameterFile.par [optional_args]
```

This command runs the model in serial, for the network corresponding with the relevant parameter file found at `data/par/parameterFile.par`.

PETSc library compatibility grants access to a variety of useful command line flags, depending on the inclusion of the respective methods that the flags allow interfacing with. Here we list some of the flags most relevant to our numerical model implementation and refer the interested reader to the PETSc documentation [5].

`-pc_type`: Assigns the preconditioner method. For direct solvers this should be set to `lu`, and for calculating the condition number of the matrix this should be set to `svd`.

`-pc_factor_mat_solver_type`: Assigns the solver for matrix factorisation. By default this is the PETSc sparse LU factorisation, but other choices include `superlu` and `mumps`.

`-pc_svd_monitor`: Prints information about the condition number and extreme singular values. Should be called with `-pc_type svd`.

In addition to the command line flags available through PETSc, there are a variety of other optional flags custom to our implementation. We list these below.

`-gamma`: Assigns the macroscopic shear strain γ applied to the upper boundary of the network. Set to 0.01 by default.

`-y_mod`: Assigns the Youngs modulus E^f of the network fibres. Set to 1.0 by default.

`-use_ksp`: Specifies whether or not to solve the system using a matrix based direct or iterative method. True by default.

`-use_ts`: Specifies whether or not to solve the system using (pseudo)-time stepping. False by default.

`-ts_restart`: Specifies whether to restart the (pseudo)-time stepping method from a new initial guess of the **U** vector, or whether to read in a previously stored estimate. Reads from the file named `vector.dat` by default.

`-max_steps`: Assigns the maximum number of steps permitted for the (pseudo)-time stepping method. Set to 1×10^7 by default.

`-alpha`: Assigns the step size for the (pseudo)-time stepping method. Set to 0.1 by default.

`-f_tol`: Assigns the stopping tolerance for the (pseudo)-time stepping method. Set to 1×10^{-12} by default.

`-use_em`: Specifies whether or not to embed the network in an elastic background medium. False by default.

`-k`: Assigns the elastic spring constant, λ_{EM}, associated with the elastic background medium. Set to 1×10^{-5} by default.

Bibliography

[1] T. Allen, T. D. Allen, and G. Cowling. *The cell: A very short introduction.* Oxford University Press, 2011.

[2] R. Ananthakrishnan, J. Guck, F. Wottawah, S. Schinkinger, B. Lincoln, M. Romeyke, T. Moon, and J. Käs. Quantifying the contribution of actin networks to the elastic strength of fibroblasts. *Journal of Theoretical Biology*, 242(2):502–516, Sept. 2006.

[3] M. Bai, A. R. Missel, W. S. Klug, and A. J. Levine. The mechanics and affine–nonaffine transition in polydisperse semiflexible networks. *Soft Matter*, 7(3):907–914, 2011.

[4] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, A. Dener, V. Eijkhout, W. D. Gropp, D. Karpeyev, D. Kaushik, M. G. Knepley, D. A. May, L. C. McInnes, R. T. Mills, T. Munson, K. Rupp, P. Sanan, B. F. Smith, S. Zampini, H. Zhang, and H. Zhang. PETSc Web page. `https://www.mcs.anl.gov/petsc`, 2019.

[5] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, A. Dener, V. Eijkhout, W. D. Gropp, D. Karpeyev, D. Kaushik, M. G. Knepley, D. A. May, L. C. McInnes, R. T. Mills, T. Munson, K. Rupp, P. Sanan, B. F. Smith, S. Zampini, H. Zhang, and H. Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 3.13, Argonne National Laboratory, 2020.

[6] M. Bathe, C. Heussinger, M. M. Claessens, A. R. Bausch, and E. Frey. Cytoskeletal Bundle Mechanics. *Biophysical Journal*, 94(8):2955–2964, Apr. 2008.

[7] J. R. Blundell and E. M. Terentjev. Stretching Semiflexible Filaments and Their Networks. *Macromolecules*, 42(14):5388–5394, July 2009.

[8] J. R. Blundell and E. M. Terentjev. Semiflexible filaments subject to arbitrary interactions: a Metropolis Monte Carlo approach. *Soft Matter*, 7(8):3967, 2011.

[9] C. Broedersz and F. MacKintosh. Modeling semiflexible polymer networks. *Reviews of Modern Physics*, 86(3):995–1036, July 2014.

[10] C. P. Broedersz, M. Depken, N. Y. Yao, M. R. Pollak, D. A. Weitz, and F. C. MacKintosh. Cross-Link-Governed Dynamics of Biopolymer Networks. *Physical Review Letters*, 105(23):238101, Nov. 2010.

[11] C. P. Broedersz, X. Mao, T. C. Lubensky, and F. C. MacKintosh. Criticality and isostaticity in fibre networks. *Nature Physics*, 7(12):983–988, 2011.

[12] C. P. Broedersz, M. Sheinman, and F. C. MacKintosh. Filament-Length-Controlled Elasticity in 3D Fiber Networks. *Physical Review Letters*, 108(7):078102, Feb. 2012.

[13] C. P. Broedersz, C. Storm, and F. C. MacKintosh. Nonlinear Elasticity of Composite Networks of Stiff Biopolymers with Flexible Linkers. *Physical Review Letters*, 101(11):118103, Sept. 2008.

[14] G. A. Buxton and N. Clarke. "Bending to Stretching" Transition in Disordered Networks. *Physical Review Letters*, 98(23):238103, June 2007.

[15] C. Calladine. Buckminster Fuller's "Tensegrity" structures and Clerk Maxwell's rules for the construction of stiff frames. *International Journal of Solids and Structures*, 14(2):161–172, 1978.

[16] M. M. A. E. Claessens, M. Bathe, E. Frey, and A. R. Bausch. Actin-binding proteins sensitively mediate F-actin bundle stiffness. *Nature Materials*, 5(9):748–753, Sept. 2006.

[17] M. Doi. *Soft matter physics*. Oxford University Press, 2013.

[18] M. Doi, S. F. Edwards, and S. F. Edwards. *The theory of polymer dynamics*, volume 73. oxford university press, 1988.

[19] D. H. Eberly. *3D game engine design: a practical approach to real-time computer graphics*. CRC Press, 2006.

[20] H. C. Elman, D. J. Silvester, and A. J. Wathen. *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*. Oxford University Press, USA, 2014.

[21] W. Ford. *Numerical linear algebra with applications: Using MATLAB*. Academic Press, 2014.

[22] M. L. Gardel. Elastic Behavior of Cross-Linked and Bundled Actin Networks. *Science*, 304(5675):1301–1305, May 2004.

[23] G. Gompper, P. Lettinga, J. Elgeti, C. Fahlke, J. K. Dhont, S. Förster, D. Fedosov, and A. Offenhäusser. Physics of life. Technical report, Zelluläre Biophysik, 2018.

[24] D. A. Head. Private communication, 2020.

[25] D. A. Head, A. J. Levine, and F. C. MacKintosh. Deformation of Cross-Linked Semiflexible Polymer Networks. *Physical Review Letters*, 91(10):108102, Sept. 2003.

[26] D. A. Head, A. J. Levine, and F. C. MacKintosh. Distinct regimes of elastic response and deformation modes of cross-linked cytoskeletal and semiflexible polymer networks. *Physical Review E*, 68(6):061907, Dec. 2003.

[27] D. A. Head, F. C. MacKintosh, and A. J. Levine. Nonuniversality of elastic exponents in random bond-bending networks. *Physical Review E*, 68(2):025101, Aug. 2003.

[28] H. Herrmann, H. Bär, L. Kreplak, S. V. Strelkov, and U. Aebi. Intermediate filaments: from cell architecture to nanomechanics. *Nature reviews Molecular cell biology*, 8(7):562–573, 2007.

[29] C. Heussinger, M. Bathe, and E. Frey. Statistical Mechanics of Semiflexible Bundles of Wormlike Polymer Chains. *Physical Review Letters*, 99(4):048101, July 2007.

[30] C. Heussinger and E. Frey. Role of architecture in the elastic response of semiflexible polymer and fiber networks. *Physical Review E*, 75(1):011917, Jan. 2007.

[31] C. Heussinger, B. Schaefer, and E. Frey. Nonaffine rubber elasticity for stiff polymer networks. *Physical Review E*, 76(3):031906, Sept. 2007.

[32] C. Heussinger, F. Schüller, and E. Frey. Statics and dynamics of the wormlike bundle model. *Physical Review E*, 81(2):021904, Feb. 2010.

[33] J. Horsfield. Storage and visualisation of fibre networks. `https://gitlab.com/M-R-Houghton/storage-and-visualisation/-/tree/continuedDevelopment`, 2016.

[34] M. Houghton. An elastic fibre network generator & visualiser. `https://gitlab.com/M-R-Houghton/storage-and-visualisation/-/tree/continuedDevelopment`, 2018–2020.

[35] M. Houghton. A numerical model for elastic fibre networks. `https://github.com/M-R-Houghton/petsc_model`, 2018–2020.

[36] M. Houghton, D. Head, and M. Walkley. A numerical model for random fibre networks. In *International Conference on Numerical Methods and Applications*, pages 408–415. Springer, 2018.

[37] E. M. Huisman and T. C. Lubensky. Internal Stresses, Normal Modes, and Nonaffinity in Three-Dimensional Biopolymer Networks. *Physical Review Letters*, 106(8):088301, Feb. 2011.

[38] E. M. Huisman, C. Storm, and G. T. Barkema. Monte Carlo study of multiply crosslinked semiflexible polymer networks. *Physical Review E*, 78(5):051801, Nov. 2008.

[39] E. M. Huisman, C. Storm, and G. T. Barkema. Frequency-dependent stiffening of semiflexible networks: A dynamical nonaffine to affine transition. *Physical Review E*, 82(6):061902, Dec. 2010.

[40] E. M. Huisman, T. van Dillen, P. R. Onck, and E. Van der Giessen. Three-Dimensional Cross-Linked F-Actin Networks: Relation between Network Architecture and Mechanical Behavior. *Physical Review Letters*, 99(20):208103, Nov. 2007.

[41] I. M. Hutten. *Handbook of nonwoven filter media*. Elsevier, 2007.

[42] H. Isambert and A. Maggs. Dynamics and rheology of actin solutions. *Macromolecules*, 29(3):1036–1040, 1996.

[43] H. Ishikawa, R. Bischoff, and H. Holtzer. Mitosis and intermediate-sized filaments in developing skeletal muscle. *The Journal of cell biology*, 38(3):538–555, 1968.

[44] J. Käs, H. Strey, and E. Sackmann. Direct imaging of reptation for semiflexible actin filaments. *Nature*, 368(6468):226–229, 1994.

[45] K. Kasza, C. Broedersz, G. Koenderink, Y. Lin, W. Messner, E. Millman, F. Nakamura, T. Stossel, F. MacKintosh, and D. Weitz. Actin filament length tunes elasticity of flexibly cross-linked actin networks. *Biophysical journal*, 99(4):1091–1100, 2010.

[46] T. Kim, W. Hwang, and R. D. Kamm. Computational Analysis of a Cross-linked Actin-like Network. *Experimental Mechanics*, 49(1):91–104, Feb. 2009.

[47] T. Kim, W. Hwang, H. Lee, and R. D. Kamm. Computational Analysis of Viscoelastic Properties of Crosslinked Actin Networks. *PLoS Computational Biology*, 5(7):e1000439, July 2009.

[48] T. P. J. Knowles and M. J. Buehler. Nanomechanics of functional and pathological amyloid materials. *Nature Nanotechnology*, 6(8):469–479, 2011.

[49] O. Kratky and G. Porod. Röntgenuntersuchung gelöster fadenmoleküle. *Recueil des Travaux Chimiques des Pays-Bas*, 68(12):1106–1122, 1949.

[50] S. Köster, D. A. Weitz, R. D. Goldman, U. Aebi, and H. Herrmann. Intermediate filament mechanics in vitro and in the cell: from coiled coils to filaments, fibers and networks. *Current Opinion in Cell Biology*, 32:82–91, Feb. 2015.

[51] L. D. Landau, A. Kosevich, L. P. Pitaevskii, and E. M. Lifshitz. Theory of elasticity. 1986.

[52] M. Latva-Kokko, J. Mäkinen, and J. Timonen. Rigidity transition in two-dimensional random fiber networks. *Physical Review E*, 63(4):046113, Mar. 2001.

[53] M. Latva-Kokko and J. Timonen. Rigidity of random networks of stiff fibers in the low-density limit. *Physical Review E*, 64(6):066117, Nov. 2001.

[54] B. Laufer. The early history of felt. *American Anthropologist*, 32(1):1–18, 1930.

[55] X. S. Li, J. W. Demmel, J. R. Gilbert, L. Grigori, M. Shao, and I. Yamazaki. Superlu users' guide. *Lawrence Berkeley National Laboratory*, 1999.

[56] Y.-C. Lin, N. Y. Yao, C. P. Broedersz, H. Herrmann, F. C. MacKintosh, and D. A. Weitz. Origins of Elasticity in Intermediate Filament Networks. *Physical Review Letters*, 104(5):058101, Feb. 2010.

[57] H. Lodish, A. Berk, S. L. Zipursky, P. Matsudaira, D. Baltimore, and J. Darnell. Molecular cell biology 4th edition. *National Center for Biotechnology Information, Bookshelf*, 2000.

[58] F. C. MacKintosh, J. Käs, and P. A. Janmey. Elasticity of Semiflexible Biopolymer Networks. *Physical Review Letters*, 75(24):4425–4428, Dec. 1995.

[59] MATLAB Documentation. MathWorks Help Center. `https://uk.mathworks.com/help/matlab/ref/mldivide.html`, 2020.

[60] A. R. Missel, M. Bai, W. S. Klug, and A. J. Levine. Affine-nonaffine transition in networks of nematically ordered semiflexible polymers. *Physical Review E*, 82(4):041907, Oct. 2010.

[61] D. C. Morse. Tube diameter in tightly entangled solutions of semiflexible polymers. *Physical Review E*, 63(3):031502, 2001.

[62] Y. Mulla, F. MacKintosh, and G. H. Koenderink. Origin of Slow Stress Relaxation in the Cytoskeleton. *Physical Review Letters*, 122(21):218102, May 2019.

[63] P. Müller and J. Kierfeld. Wrinkling of Random and Regular Semiflexible Polymer Networks. *Physical Review Letters*, 112(9):094303, Mar. 2014.

[64] Nonwovens Innovation & Research Institute. NIRI Ltd. `http://www.nonwovens-innovation.com`, 2020.

[65] Nonwovens Research Group. University of Leeds. `https://nonwovens.leeds.ac.uk`, 2020.

[66] P. R. Onck, T. Koeman, T. van Dillen, and E. van der Giessen. Alternative Explanation of Stiffening in Cross-Linked Semiflexible Networks. *Physical Review Letters*, 95(17):178102, Oct. 2005.

[67] C. M. O'Connor, J. U. Adams, and J. Fairman. Essentials of cell biology. *Cambridge, MA: NPG Education*, 1:54, 2010.

[68] E. K. Paluch. Actin cortex mechanics and cell shape control in migration and division. *Biophysical Journal*, 108(2):4a, 2015.

[69] G. Patrick. *Organic chemistry: a very short introduction*. Oxford University Press, 2017.

[70] R. H. Pritchard, Y. Y. S. Huang, and E. M. Terentjev. Mechanics of biological networks: from the cell cytoskeleton to connective tissue. *Soft matter*, 10(12):1864–1884, 2014.

[71] A. Quarteroni, R. Sacco, and F. Saleri. *Numerical mathematics*, volume 37. Springer Science & Business Media, 2010.

[72] R. Rens, M. Vahabi, A. Licup, F. MacKintosh, and A. Sharma. Nonlinear mechanics of athermal branched biopolymer networks. *The Journal of Physical Chemistry B*, 120(26):5831–5841, 2016.

[73] M. Rubinstein, R. H. Colby, et al. *Polymer physics*, volume 23. Oxford university press New York, 2003.

[74] S. J. Russell. *Handbook of nonwovens*. Woodhead Publishing, 2006.

[75] S. J. Russell. Private communication, 2019.

[76] Y. Saad. *Iterative methods for sparse linear systems*, volume 82. siam, 2003.

[77] C. Schuldt, J. Schnauß, T. Händler, M. Glaser, J. Lorenz, T. Golde, J. A. Käs, and D. M. Smith. Tuning synthetic semiflexible networks by bending stiffness. *Physical review letters*, 117(19):197801, 2016.

[78] A. M. Stein, D. A. Vader, D. A. Weitz, and L. M. Sander. The micromechanics of three-dimensional collagen-I gels. *Complexity*, 16(4):22–28, Mar. 2011.

[79] O. Stenull and T. C. Lubensky. Simple lattice model for biological gels. *arXiv:1108.4328 [cond-mat]*, Aug. 2011. arXiv: 1108.4328.

[80] C. Storm, J. J. Pastore, F. C. MacKintosh, T. C. Lubensky, and P. A. Janmey. Nonlinear elasticity in biological gels. *Nature*, 435(7039):191–194, May 2005.

[81] D. Sunday. Geometry algorithms, 2001.

[82] I. Szeverenyi, A. J. Cassidy, C. W. Chung, B. T. Lee, J. E. Common, S. C. Ogg, H. Chen, S. Y. Sim, W. L. Goh, K. W. Ng, et al. The human intermediate filament database: comprehensive information on a gene family involved in many human diseases. *Human mutation*, 29(3):351–360, 2008.

[83] M. Tassieri. Dynamics of semiflexible polymer solutions in the tightly entangled concentration regime. *Macromolecules*, 50(14):5611–5618, 2017.

[84] M. Tassieri, R. Evans, L. Barbu-Tudoran, J. Trinick, and T. Waigh. The self-assembly, elasticity, and dynamics of cardiac thin filaments. *Biophysical journal*, 94(6):2170–2178, 2008.

[85] M. Tassieri, R. M. L. Evans, L. Barbu-Tudoran, G. N. Khaname, J. Trinick, and T. A. Waigh. Dynamics of Semiflexible Polymer Solutions in the Highly Entangled Regime. *Physical Review Letters*, 101(19):198301, Nov. 2008.

[86] Y. Tsuda, H. Yasutake, A. Ishijima, and T. Yanagida. Torsional rigidity of single actin filaments and actin-actin bond breaking force under torsion measured directly by in vitro micromanipulation. *Proceedings of the National Academy of Sciences*, 93(23):12937–12942, Nov. 1996.

[87] A. F. Turbak et al. *Nonwovens: theory, process, performance, and testing*. 1993.

[88] J. M. Van Doorn, L. Lageschaar, J. Sprakel, and J. van der Gucht. Criticality and mechanical enhancement in composite fiber networks. *Physical Review E*, 95(4):042503, 2017.

[89] M. F. Vermeulen, A. Bose, C. Storm, and W. G. Ellenbroek. Geometry and the onset of rigidity in a disordered network. *Physical Review E*, 96(5):053003, 2017.

[90] J. Wilhelm and E. Frey. Radial Distribution Function of Semiflexible Polymers. *Physical Review Letters*, 77(12):2581–2584, Sept. 1996.

[91] J. Wilhelm and E. Frey. Elasticity of Stiff Polymer Networks. *Physical Review Letters*, 91(10):108103, Sept. 2003.

[92] Z. Xu, R. Paparcone, and M. J. Buehler. Alzheimer's AB(1-40) Amyloid Fibrils Feature Size-Dependent Mechanical Properties. *Biophysical Journal*, 98(10):2053–2062, May 2010.

[93] R. Yasuda, H. Miyata, and K. Kinosita Jr. Direct measurement of the torsional rigidity of single actin filaments. *Journal of molecular biology*, 263(2):227–236, 1996.

[94] J. A. Åström, P. B. S. Kumar, I. Vattulainen, and M. Karttunen. Strain hardening, avalanches, and strain softening in dense cross-linked actin networks. *Physical Review E*, 77(5):051913, May 2008.

[95] G. Žagar, P. R. Onck, and E. Van der Giessen. Elasticity of Rigidly Cross-Linked Networks of Athermal Filaments. *Macromolecules*, 44(17):7026–7033, Sept. 2011.