COMPUTATIONAL METHODS FOR MORPHING MATTER: FROM EMBRYOS TO SELF-FOLDING ARCHITECTURE

Anvitha Sudhakar

A DISSERTATION PRESENTED TO THE FACULTY OF PRINCETON UNIVERSITY IN CANDIDACY FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

RECOMMENDED FOR ACCEPTANCE BY THE DEPARTMENT OF MECHANICAL AND AEROSPACE ENGINEERING

Adviser: Prof. Andrej Kosmrlj

November 2025

Abstract

Morphogenesis, the process by which living organisms achieve remarkable shape transformations through coordinated growth and deformation, represents a fundamental challenge at the intersection of biology, physics, and engineering. From the folding of embryonic tissues during development to the deployment of engineered structures inspired by biological systems, understanding how materials change shape in response to internal and external stresses is crucial for advancing both our knowledge of biological development and our ability to create adaptive engineering systems.

This dissertation develops comprehensive computational frameworks for modeling morphing systems, with the primary objective of understanding the underlying mechanisms that control emerging shapes. The work addresses critical gaps in our ability to predict growth-induced shape changes across multiple scales and material systems.

The research establishes robust numerical methods based on Finite Element Method and energy minimization principles, creating computational infrastructure capable of simulating large deformations in complex geometries with heterogeneous growth fields. A key contribution is the development of predictive design tools for self-folding systems, integrating physics-based simulation with machine learning approaches to enable both forward prediction of folding outcomes and inverse design of material parameters for target shapes.

The framework is demonstrated through three major applications: First, epithelial morphogenesis in lizard lung development is modeled, revealing mechanical principles governing honeycomb-like faveolar architecture formation. Second, large-scale self-folding systems operating under gravitational constraints are analyzed, establishing design methodologies for practical engineering applications. Third, the theoretical framework is extended to incorporate visco-elastic material behavior, addressing limitations of purely elastic models for biological tissues through energy-rate minimization formulations that capture both elastic stored energy and viscous dissipation.

The computational methods developed capture experimentally observed cellular patterns and tissue deformations, providing validated tools for linking mechanical behavior to chemical and biological factors. This work advances understanding of specific morphing systems while establishing robust computational frameworks that can be extended for future research in morphing matter, bridging biological morphogenesis and engineered adaptive structures.

Acknowledgements

This dissertation carries T#3490 in the records of the Department of Mechanical and Aerospace Engineering.

Contents

Chapter 1

Introduction

1.1 Motivation

Living organisms achieve remarkable shape transformations, in a process called morphogenesis, through coordinated growth and deformation processes. The morphogenesis of biological structures emerges from complex interactions between mechanical forces, material properties, and growth patterns. In developing embryos, tissues fold, stretch, and reorganize to form functional organs through processes that remain incompletely understood. Similarly, modern engineering seeks to create adaptive structures that can autonomously reconfigure their geometry, promising applications from deployable space structures to reconfigurable infrastructure. From the folding of embryonic tissues during development to the deployment of engineered structures inspired by biological systems, understanding how materials change shape in response to internal and external stresses represents a challenge at the intersection of biology, physics, and engineering. Despite these diverse contexts, both biological and engineered morphing systems share fundamental physical principles that govern their behavior. This dissertation develops computational methods to predict development of shape in both biological and engineered systems, addressing critical gaps in our ability to model growth-induced shape change across scales.

1.2 Research Objectives

This dissertation develops computational frameworks for morphing systems with the primary objective of understanding the underlying mechanisms that control the emerging shape. The following objectives are key to achieve our primary objective and also to enable ongoing research:

1. **Develop robust numerical methods for growth mechanics**: Create a finite element workflow and solver algorithms, capable of simulating large deformations in complex geometries with heterogeneous growth fields.

2. **Establish predictive design tools for self-folding systems**: Build detailed simulations of folding processes along with data-driven methods from thousands of simulation results, enabling both forward prediction of folding outcomes and inverse design of material parameters to achieve target shapes.

3. **Extend morphoelastic theory to viscoelastic materials**: Incorporate time-dependent material responses into growth mechanics, while maintaining the applicability to large deformations, to determine transient and final morphologies in biological tissues.

1.3 Dissertation Overview

This dissertation employs computational approaches based on Finite Element Method and energy minimization principles to model morphing across biological and engineered systems. The dissertation is organized as follows:

Computational Framework for Morphing Matter

Chapter 2 establishes the theoretical and numerical foundations for simulating growth-induced shape change. We develop procedures for mesh generation suitable for complex morphing geometries, formulate deformation as an energy minimization problem that assumes elastic material response, and present adaptive numerical algorithms for solving the resulting nonlinear systems. This framework provides the computational infrastructure for all subsequent chapters.

Stress-Ball Morphogenesis

Chapter 3 demonstrates the application of our computational methods to understand epithelial morphogenesis in the lizard lung [1]. Through simulations incorporating smooth muscles and luminal pressure, we reveal the mechanical principles that sculpt the honeycomb-like architecture of faveolar lungs. The computational model is built to predict experimental observations and we identify critical parameters governing epithelial corrugation formation.

Predictive Design of Large-Scale Self-Folding

Chapter 4 addresses the challenge of designing self-folding systems that function reliably under gravity. We develop a comprehensive framework integrating physics-based simulation and machine learning to predict folding outcomes and identify optimal design parameters. This work establishes forward design capabilities and pathways for inverse design, both impactful to engineer self-folding structures at practical scales.

Viscoelastic Morphing

Chapter 5 extends the morphoelastic framework to incorporate viscoelastic material behavior, addressing the limitations of purely elastic models for biological tissues. We develop energy-rate minimization formulations that capture both elastic stored energy and viscous dissipation, validated through know creep behavior of different viscoelastic materials. Simulations using this framework captured epithelial morphogenesis guided by experimentally observed cellular patterns, useful to prove the link between tissue deformation and chemical factors.

This dissertation advances our understanding of specific morphing systems, while developing robust computational frameworks that can be extended for future research in morphing matter.

Chapter 2

Computational methods for morphing matter

This chapter develops the computational methods needed to simulate growth-induced shape change in biological and engineered structures. We address four key components required for such simulations. First, we establish procedures for creating computational domains and generating meshes suitable for complex morphing geometries (Section 2.1). Second, we formulate the deformation governed by growth as an energy minimization problem (Section 2.2). Third, we discretize these continuum equations using finite element methods adapted for large deformations (Section 2.3). Fourth, we present two numerical algorithms to solve the resulting nonlinear systems: a modified Newton's method for standard cases and an arc-length method for problems involving instabilities (Section 2.4). These methods provide the computational foundation for all simulations of morphing presented in this dissertation.

2.1 Setting up the Computational Domain

2.1.1 Domain Geometry and Initialization

Accurate numerical simulation of mechanical systems begins with a precise representation of the physical domain. This section outlines the systematic approach to creating a domain geometry that captures the essential geometric and topological characteristics of the system under investigation.

Creating the computational domain begins with defining a geometry that captures the initial reference configuration of the system being modeled. Typically, this initial reference configuration is stress-free, establishing a baseline from which all subsequent shape changes or deformations are accurately measured.

Mesh dimensionality

An essential consideration in setting up the domain is identifying the appropriate spatial dimensionality and mesh dimensionality: spatial dimensionality captures how the structure can physically deform, while mesh dimensionality reflects how the structure is topologically represented. For example, a growing tendril is modeled with a one-dimensional mesh deforming in three-dimensional space in Figure 2.1. The computational problems addressed in this thesis specifically involve fully three-dimensional meshes deforming in three-dimensional space, requiring 3D spatial and mesh dimensionalities.

Construction of mesh geometry

Constructing a geometry suitable for subsequent meshing requires a structured approach to ensure the representation of a continuous domain, free from irregularities or gaps that could compromise continuum assumptions. Our approach to creating a 3D mesh-able geometry in the *Gmsh* software involves the following steps, sketched for an example domain in Figure 2.2:

1. **Identifying Key Points:**

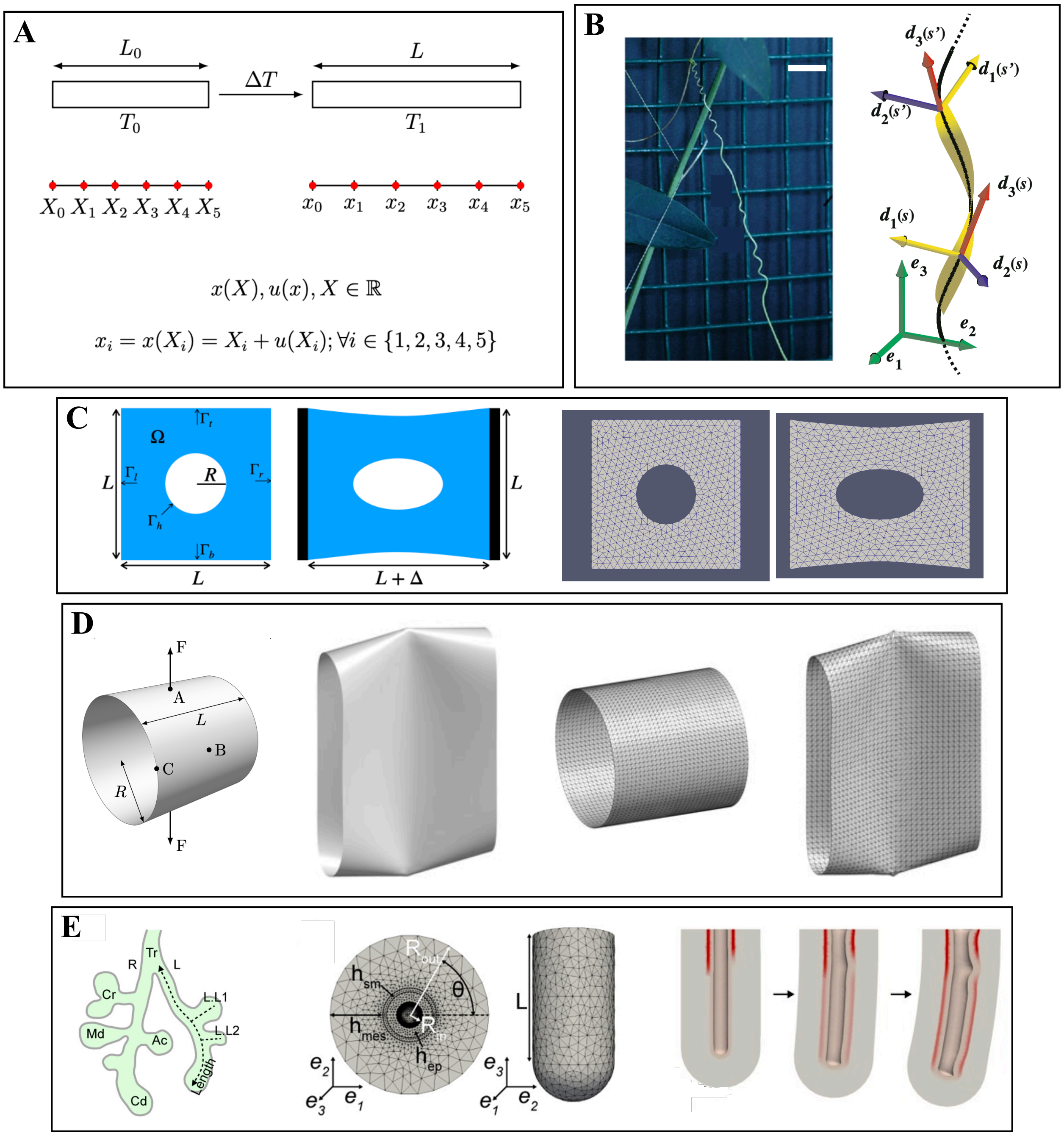


Figure 2.1: **(A)** A metal rod expanding along its length due to heating, represented by a 1D mesh with a 1D displacement field. **(B)** A tendril of the genus Passiflora (passion flowers) with perversions (left)[2], modeled using a spatial curve with a 1D mesh representing material points and 3D field representing the current configuration (right) [3]. **(C)** Deformation of a 2D elastic material with a circular hole which is clamped at the left and right boundaries and extended (left), represented by 2D meshes with a 2D displacement field (right). **(D)** A thin cylinder stretched by forces as shown to a buckled state (left), modeled using a 2D mesh and a 3D displacement field (right)[4]. **(E)** Sketch of a mouse lung (left) and a 3D mesh (middle) used to model the budding of a branch (right) [5].

Important spatial reference points within the domain, often situated along boundaries or interfaces, are initially marked. These points serve as anchors or vertices for subsequent geometry construction.

2. **Connecting Points with Smooth Curves:**

   Identified points are connected using smooth plane curves, typically straight lines or arcs, ensuring geometric coherence and continuity along domain edges.

3. **Defining Curve Loops:**

   Multiple lines or curves are combined to form closed curve loops. These loops must be fully connected, clearly defining enclosed regions without gaps or intersections.

4. **Creating Surfaces from Curve Loops:**

   Enclosed two-dimensional regions defined by connected curve loops are transformed into continuous surfaces. Each surface must be carefully constructed to remain fully connected, ensuring compatibility with meshing procedures and subsequent analysis.

5. **Constructing Surface Loops:**

   Surfaces are then organized into surface loops, wherein multiple surfaces form closed and fully connected boundaries. This defines the 3D volume boundary.

6. **Defining 3D Volumes from Surface Loops:**

   The final step involves defining a three-dimensional volume from closed surface loops. This fully enclosed region defines the computational domain, representing the initial stress-free configuration, ready for meshing.

By systematically following this approach, the resulting domain geometry is robust, fully connected, and well-suited for meshing and subsequent numerical analyses involving complex shape changes.

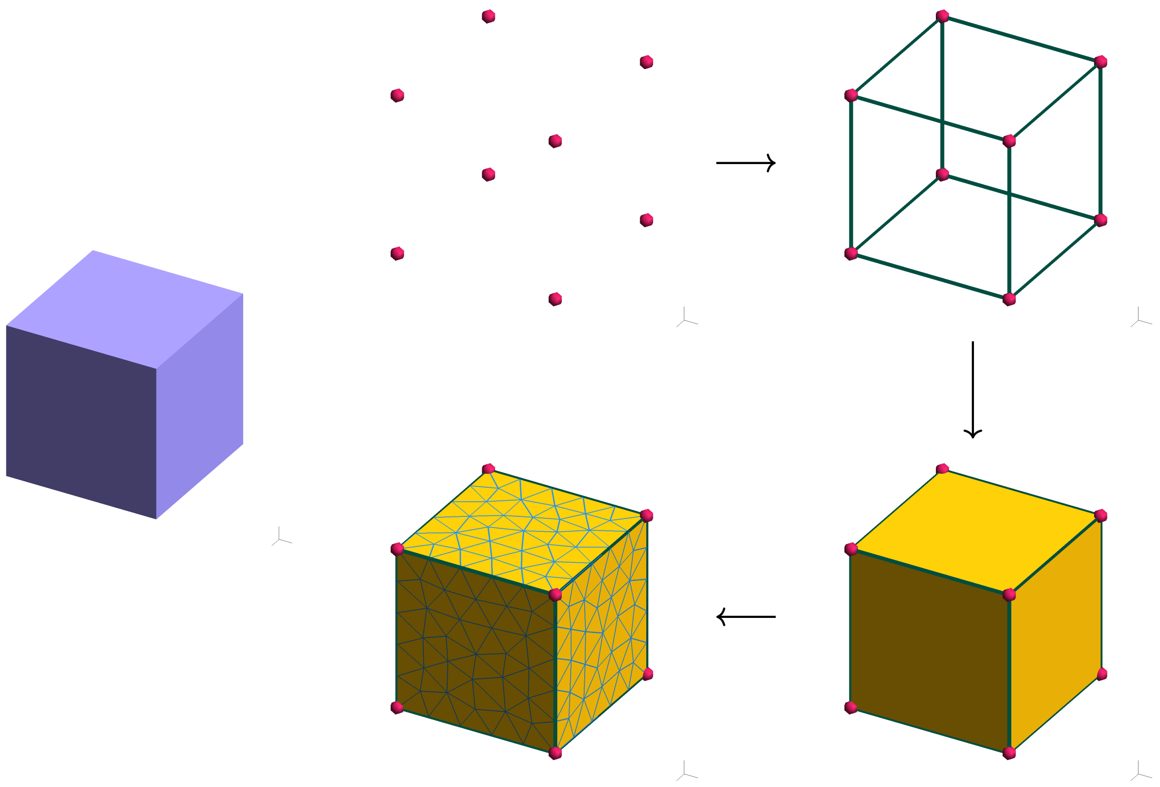


Figure 2.2: **Left**: An example body, cube, from which a domain is to be created. **Right**: Workflow of constructing the mesh geometry. Starting with a collection of relevant points, the points are connected using curves, closed curve loops form the surfaces, and surface loops define the final volume. The final volume is meshed in 3D to have a computational domain representing the body.

Marking sub-domains and boundaries

Additionally, specific geometric elements within the domain can be tagged to delineate sub-domains with distinct material properties (Figure 2.3D), designate surface or volume segments for applying different forces, or mark boundary sections to impose specific boundary conditions. This tagging enhances model accuracy by enabling precise specification of simulation conditions and a detailed representation of underlying physical phenomena.

2.1.2 Meshing Strategies

After establishing the computational domain geometry, the domain is discretized into smaller elements through a process known as meshing. Different strategies to partition the computational domain are tailored to specific computational requirements [6]:

- **Structured Meshes:** Structured meshes employ regular patterns with uniform or systematically varying element sizes (Figure 2.3 A) [7]. These meshes offer ease of implementation and high computational efficiency but may inadequately represent complex or regions with curved boundaries, as the regularity of the mesh may not be able to conform to the shape of the domain.

- **Unstructured Meshes:** Unstructured meshes provide greater flexibility to accurately capturing complex geometries since it's possible to easily fit the elements of these meshes within irregular and non-orthogonal boundaries (Figure 2.3 B). However, they typically require sophisticated mesh-generation algorithms and can increase computational complexity.

- **Adaptive Meshing (Adaptive Refinement):** Adaptive meshing dynamically adjusts element size and density according to local solution gradients or geometric complexity (Figure 2.3 C). This method optimizes computational resources by enhancing accuracy without substantially increasing computational cost.

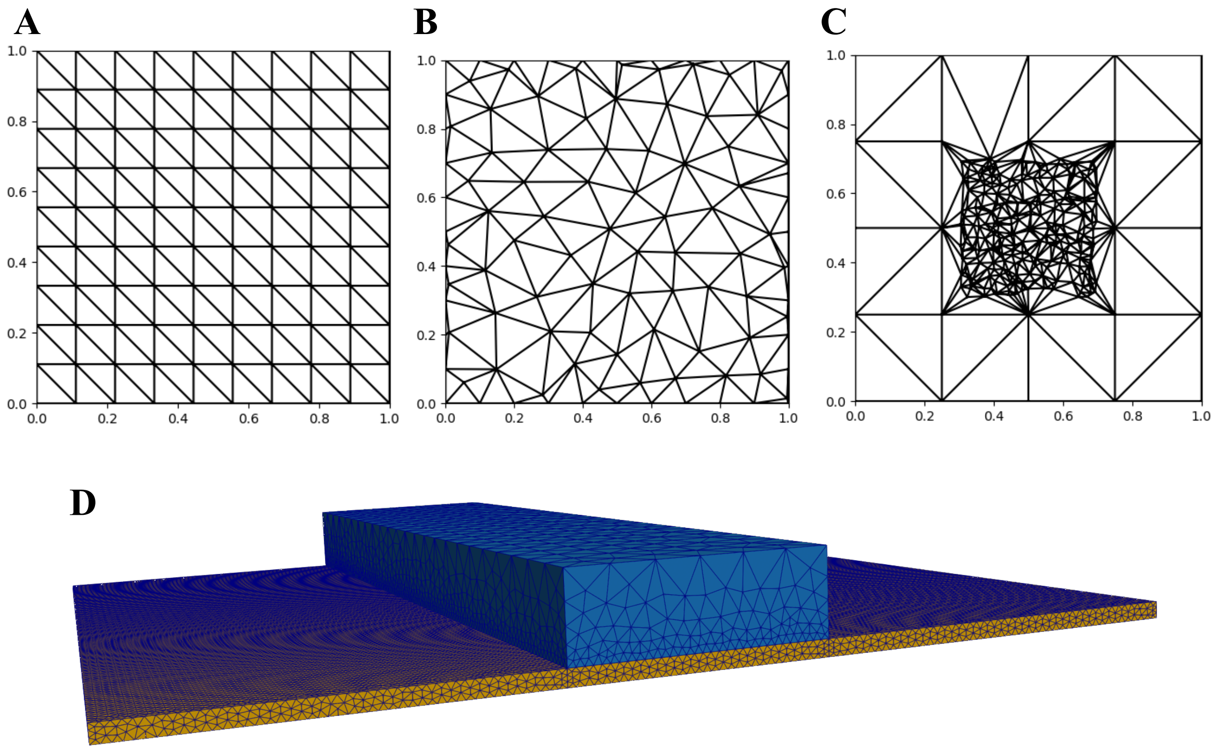


Figure 2.3: **(A)** Structured 2D mesh on a unit square. **(B)** Unstructured 2D mesh on a unit square. **(C)** Adaptive mesh prescribed with refinement of mesh size toward the center of the unit square. **(D)** A structure made of different materials marked in blue and yellow, distinguished in the computational domain using mesh markers and meshed on *Gmsh*. The mesh size is prescribed as function of the thickness of the material to have multiple elements across the thickness to comply with continuum assumptions. Adaptive meshing ensures smooth transition of mesh sizes across the shared material boundary and meshing algorithms ensure connectivity is maintained at the shared boundary.

2.1.3 3D Meshing Using *Gmsh*

In this thesis, *Gmsh* software is utilized to generate unstructured meshes from predefined domain geometries, with specific modifications detailed later in this section [8].

***Gmsh*'s pipeline for conformal meshing**

Gmsh generates finite element meshes by representing model geometries through tessellations composed of geometric elements such as lines, triangles, quadrangles, tetrahedra, prisms, hexahedra, and pyramids. These elements intersect strictly at faces, edges, or nodes, forming a conformal mesh. *Gmsh*'s meshing modules offer algorithms specifically designed to generate these conformal meshes. *Gmsh* ensures mesh conformity using a bottom-up meshing process. Curves are discretized first, surfaces are then meshed based on these discretized curves, and finally, these meshed surfaces serve as a boundary for meshing volumes. Throughout this process, the mesh of any entity is constrained only by the mesh of its boundary. In three-dimensional contexts, triangles discretizing surfaces must form tetrahedral faces if the surface defines a volume boundary. This approach inherently guarantees mesh conformity, especially in scenarios involving multiple volumes sharing common surfaces.

The meshes used in this thesis are unstructured, comprising lines, triangles, and tetrahedra created via *Gmsh*'s 2D and 3D meshing algorithms, thus producing conformal three-dimensional meshed domains.

2.1.4 Meshing Practices

This subsection outlines specific practices adopted while meshing to ensure numerical stability and physical accuracy in deformation simulations. Poor meshing, due to skewed elements or inadequate discretization, can cause numerical errors, instabilities, and higher computational cost. Thus, attention to mesh quality, which includes criteria such as element aspect ratios, uniformity, and smoothness, is essential [9]. We create and use the following practices in all simulations to ensure mesh quality that improves accuracy, stability, and efficiency.

Meshing strategy

Unstructured meshes suit complex or asymmetric geometries, yielding elements with better aspect ratios. Adaptive meshing, although beneficial, requires a feedback mechanism from simulation outcomes to iteratively refine the mesh. Such refinement can be computationally intensive, necessitating careful evaluation of its necessity and potential benefits. In practice, we use unstructured meshes with tunable mesh size parameters specified before simulation.

Continuum considerations for meshing

To satisfy continuum assumptions, inherent to the modeling in Section 2.2, mesh elements must be much smaller than the domain's characteristic dimensions, essentially the smallest dimension (usually material thickness). However, smaller elements increase computational loads, highlighting a practical trade-off between mesh resolution and computational efficiency. Typically, ensuring more than two mesh elements span the smallest domain dimension minimizes boundary effects, a condition enforced by setting mesh sizes relative to each sub-domain's smallest dimension (Figure 2.3 D).

Optimizing mesh resolution

Overall mesh resolution is a factor that directly impacts simulation accuracy and is best determined through convergence studies. These studies involve performing simulations with increasingly finer meshes—typically halving the element size at each step—until additional refinements result in minimal changes in outcomes. The mesh at this convergence point is then selected for numerical simulations.

This meshing workflow balances computational efficiency, numerical accuracy, and continuum assumptions.

2.2 Modeling deformation during growth

Having established the computational domain and spatial discretization strategies, we now turn to the physical modeling framework that governs deformation. The mesh structures developed in the previous section provide the domain upon which we will formulate the continuum mechanics equations to describe growth processes driving shape change.

2.2.1 Conceptual Background

Mathematical Framework for Deformation

We begin by quantifying how a body's configuration transitions from a reference (undeformed) state to a current (deformed) state using continuum mechanics. The primary mathematical tool used for describing this transformation is the deformation gradient tensor **F**, which relates infinitesimal line elements in the reference configuration to their corresponding elements in the current configuration [10]. Mathematically, this is expressed as:

$$\mathbf{F} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}} \tag{2.1}$$

where $\boldsymbol{x}$ represents spatial coordinates in the current configuration, and $\boldsymbol{X}$ denotes material coordinates in the reference configuration.

Deformation can also be expressed in terms of displacement $\boldsymbol{u}$, defined as the difference between positions in the current and reference configurations, $\boldsymbol{u} = \boldsymbol{x} - \boldsymbol{X}$. Using displacement, the deformation gradient tensor can be represented as:

$$\mathbf{F} = \mathbf{I} + \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}} \tag{2.2}$$

where **I** is the identity tensor.

Growth as a Source of Deformation

Growth-induced deformation arises due to irreversible changes within the body's geometry, such as changes in volume, shape, or both. The existence of residual stress in biological materials is closely associated to growth or to an internal reorganization of the reference configuration. As different elements of a body change in size or relative position, they create stresses even in the absence of applied loads. These transformations create residual stresses that are essential to drive shape change.

The central postulate of morphoelasticity is that the residual stresses are solely created by a local growth deformation tensor. This principle provides a mathematical framework for analyzing how growth processes induce mechanical stresses within tissues. Central to modeling such processes is the multiplicative decomposition of the deformation gradient tensor into elastic and growth components [2, 3]:

$$\mathbf{F} = \mathbf{F}_{\mathrm{e}}\mathbf{F}_{\mathrm{g}} \tag{2.3}$$

In this framework, the total deformation gradient $\mathbf{F}$ can be expressed in terms of displacement $\boldsymbol{u}(\boldsymbol{X}, t)$ as:

$$\mathbf{F}(\boldsymbol{X}, t) = \mathbf{I} + \frac{\partial \boldsymbol{u}(\boldsymbol{X}, t)}{\partial \boldsymbol{X}} \tag{2.4}$$

The growth tensor $\mathbf{F}_{\mathrm{g}}$ typically depends on both the reference configuration and time, hence $\mathbf{F}_{\mathrm{g}}(\boldsymbol{X}, t)$, reflecting how growth processes evolve spatially and temporally throughout the body. This growth tensor characterizes local growth-induced deformations, representing a virtual stress-free intermediate configuration that results purely from growth processes. The elastic tensor $\mathbf{F}_{\mathrm{e}}$, in turn, maps this virtual growth-induced configuration into the actual current configuration, accounting for elastic responses and associated residual stresses. Thus, $\mathbf{F}_{\mathrm{e}}$ is used to analyze the resulting residual stresses and strains.

The interplay between the growth-induced tensor $\mathbf{F}_{\mathrm{g}}$ and the elastic tensor $\mathbf{F}_{\mathrm{e}}$ governs the resulting deformation patterns and stress distributions. Accurate modeling thus requires capturing both growth processes that induce residual stresses and how these stresses are redistributed through elastic deformations, ultimately defining the equilibrium state.

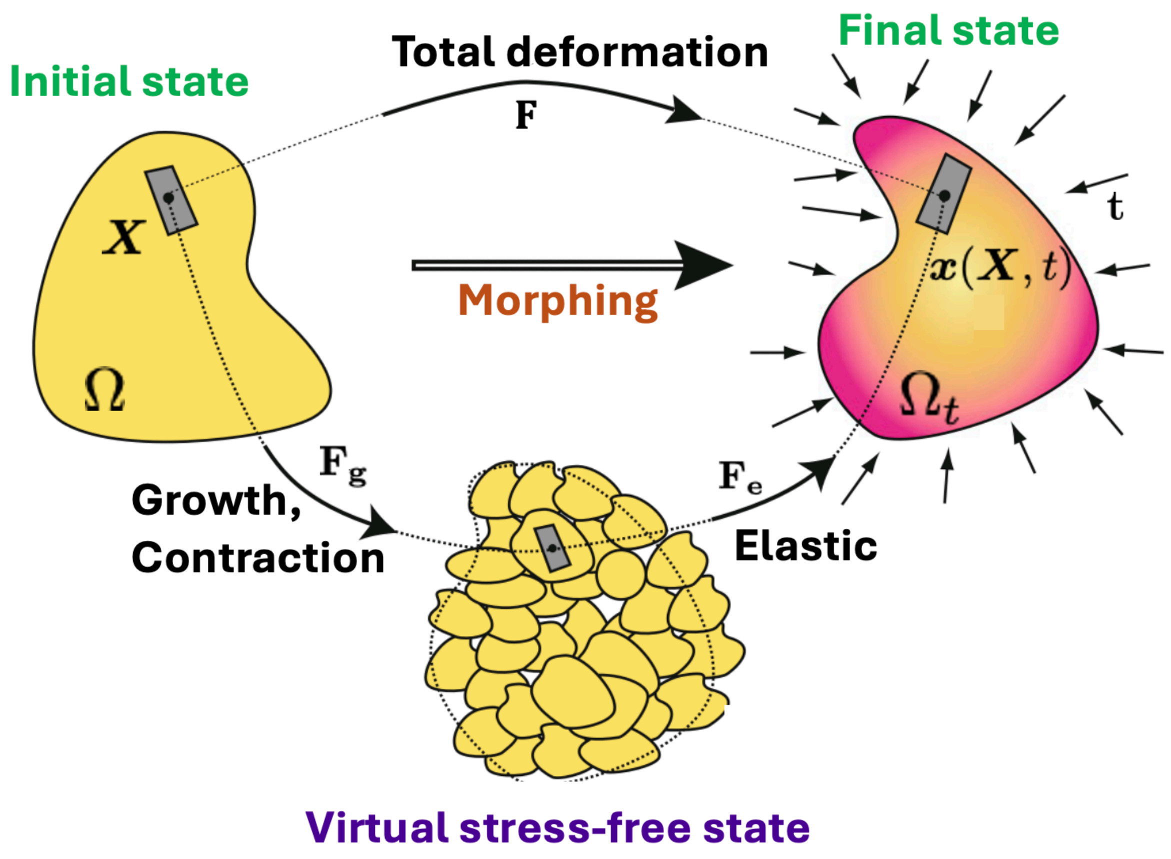


Figure 2.4: Decomposition of deformation gradient, by creating a virtual stress-free state resulting from internal growth/contraction [3].

Energy Minimization Methods

Energy minimization methods provide a rigorous framework for predicting deformation states resulting from growth. Deformation configurations correspond to states where the total potential energy is minimal. The total energy can be expressed in its primary form as:

$$\Pi(\boldsymbol{u}) = \int_{\Omega} \Psi(\mathbf{F}_{\mathrm{e}}, \boldsymbol{X})\, dV - W_{\mathrm{ext}}(\boldsymbol{u}) \tag{2.5}$$

Where $\Psi(\mathbf{F}_{\mathrm{e}}, \boldsymbol{X})$ denotes the elastic strain energy density that depends on both the elastic deformation $\mathbf{F}_{\mathrm{e}}$ and the material position $\boldsymbol{X}$ in the reference configuration. This explicit dependence on position acknowledges that material properties can vary spatially throughout the body, affecting how strain energy is stored at different locations. This heterogeneity is particularly important in biological tissues, where composition and microstructure often vary regionally. The term $W_{\mathrm{ext}}(\boldsymbol{u})$ represents the external virtual work. From Equation (2.3), $\mathbf{F}_{\mathrm{e}}$ depends on the growth and total deformation so it can be expressed as:

$$\mathbf{F}_{\mathrm{e}} = \mathbf{F} \cdot \mathbf{F}_{\mathrm{g}}^{-1} \tag{2.6}$$

Expanding this formulation to account for different sources of external work, we have:

$$\Pi(\boldsymbol{u}) = \int_{\Omega} \Psi(\mathbf{F}_{\mathrm{e}}, \boldsymbol{X})\, dV - \int_{\Omega} \boldsymbol{b} \cdot \boldsymbol{u}\, dV - \int_{\partial\Omega} \boldsymbol{t} \cdot \boldsymbol{u}\, dA - W_{\mathrm{c}}(\boldsymbol{u}) \tag{2.7}$$

Here, the second and third terms represent work done by body forces $\boldsymbol{b}$ and surface tractions $\boldsymbol{t}$, respectively, while $W_{\mathrm{c}}(\boldsymbol{u})$ accounts for virtual work from possible constraints. The domain Ω refers to the reference configuration, and $\partial\Omega$ is its boundary.

Equilibrium deformation configurations satisfy the condition:

$$\delta\Pi(\boldsymbol{u}) = 0 \tag{2.8}$$

This condition mathematically defines stable equilibrium states, allowing precise prediction and analysis of complex deformation behaviors induced by growth.

This conceptual and mathematical framework enables systematic modeling of growth-induced deformation. It supports detailed exploration and accurate prediction of shape evolution

Deformation in growing systems arises from the interplay of various factors that can be broadly categorized as active drivers of shape change and passive mechanical responses. The following sections systematically examines these factors and their mathematical representations within the modeling framework established.

2.2.2 Active Drivers of Deformation

Active factors directly drive the deformation process, either through intrinsic biological mechanisms or external mechanical influences. These factors drive deformation patterns observed in growing systems.

1. **Internal Growth and Contraction**

   Internal growth and contraction mechanisms represent the primary drivers of morphogenesis in biological systems. We model these mechanisms using the growth tensor $\mathbf{F}_\mathrm{g}$, which encodes local expansion, contraction, or reorientation of tissue [11]. Specifying $\mathbf{F}_\mathrm{g}$ through appropriate constitutive relations is an important choice in modeling deformation (Section 2.2.4).

2. **External Forces**

   External forces applied to the boundary or throughout the volume of a growing tissue can significantly influence deformation patterns.

   (a) **Traction forces** act on the boundary of the body and contribute to the deformation energy through the work term:

   $$W_{\text{trac}}(\boldsymbol{u}) = \int_{\partial\Omega_t} \boldsymbol{t} \cdot \boldsymbol{u} \, dA \tag{2.9}$$

   where $\boldsymbol{t}$ represents the prescribed traction vector, and $\partial\Omega_t$ denotes the portion of the boundary where tractions are applied.

(b) **Pressure forces** represent a special case of traction where the force acts normal to the surface with magnitude proportional to the pressure:

$$\boldsymbol{t} = -p\boldsymbol{n} \tag{2.10}$$

where p denotes the pressure magnitude, and $\boldsymbol{n}$ is the outward unit normal to the current surface. Pressure forces are particularly relevant in modeling fluid-filled structures such as plant cells or vascular tissues.

2.2.3 Passive Mechanical Factors

Passive factors encompass the mechanical conditions and material properties that influence how a body responds to applied forces or growth-induced deformations. These factors do not actively drive deformation but rather determine how the system accommodates imposed changes.

1. **Boundary Conditions**

   Boundary conditions constitute a fundamental aspect of deformation modeling, as they define the constraints imposed at the boundaries of the domain. In the context of continuum mechanics, boundary conditions typically manifest in two primary forms [12]:

   (a) **Dirichlet (essential) boundary conditions** prescribe displacements on portions of the boundary, mathematically expressed as:

   $$\boldsymbol{u} = \bar{\boldsymbol{u}} \quad \text{on} \quad \partial\Omega_D \tag{2.11}$$

   where $\bar{\boldsymbol{u}}$ represents the prescribed displacement and $\partial\Omega_D$ denotes the portion of the boundary where displacement is specified.

   (b) **Neumann (natural) boundary conditions** prescribe tractions on the boundary, represented as:

$$\mathbf{P}\boldsymbol{N} = \bar{\boldsymbol{t}} \quad \text{on} \quad \partial\Omega_N \tag{2.12}$$

where $\mathbf{P}$ is the first Piola-Kirchhoff stress tensor, $\boldsymbol{N}$ is the outward unit normal vector to the boundary in the reference configuration, and $\bar{\boldsymbol{t}}$ represents the prescribed traction on the Neumann boundary $\partial\Omega_N$.

The partitioning of the boundary satisfies $\partial\Omega = \partial\Omega_D \cup \partial\Omega_N$ and $\partial\Omega_D \cap \partial\Omega_N = \emptyset$, ensuring a well-posed boundary value problem. Appropriate boundary conditions are essential for capturing physical constraints in growth-driven deformation.

2. **Kinematic Constraints**

Kinematic constraints impose restrictions on the permissible deformations of a body, effectively reducing the degrees of freedom available to the system. Two particularly relevant classes of constraints in our computational models include:

(a) **Rigid body constraints** prevent specific modes of deformation while allowing others. Common rigid body constraints include:

- *Translation constraints*, which restrict displacement of specific points in specific directions. We apply these for constraining the global translation of the body, and the constraint can be expressed as:

$$C_{\text{trans, i}}(\boldsymbol{u}) = \int_\Omega \boldsymbol{u}(\boldsymbol{X}) \cdot \boldsymbol{e}_i \, dV = 0 \tag{2.13}$$

where $\boldsymbol{e}_i$ is a unit vector in the constrained direction. In the total energy formulation, this constraint contributes:

$$\mathcal{L}_{\text{trans}}(\boldsymbol{u}, \lambda_{\text{trans, i}}) = \Pi(\boldsymbol{u}) + \lambda_{\text{trans, i}} \left[\int_\Omega \boldsymbol{u}(\boldsymbol{X}) \cdot \boldsymbol{e}_i \, dV \right] \tag{2.14}$$

- *Rotation constraints*, which prevent rotational motion around specified axes. For the entire body, a constraint preventing rotation around axis $\boldsymbol{e}_i$ can be formulated as:

$$C_{\text{rot}}(\boldsymbol{u}) = \int_\Omega [\boldsymbol{u}(\boldsymbol{X}) \times (\boldsymbol{X} - \boldsymbol{X}_c)] \cdot \boldsymbol{e}_i \, dV = 0 \tag{2.15}$$

where $\boldsymbol{X}_c$ is the center of mass. In the variational form, this becomes:

$$\mathcal{L}_{\text{rot,i}}(\boldsymbol{u}, \lambda_{\text{rot,i}}) = \Pi(\boldsymbol{u}) + \lambda_{\text{rot,i}} \left\{ \int_\Omega [\boldsymbol{u}(\boldsymbol{X}) \times (\boldsymbol{X} - \boldsymbol{X}_c)] \cdot \boldsymbol{e}_i \, dV \right\} \tag{2.16}$$

In cases where global translation is already constrained (i.e., the center of mass remains fixed), the rotational constraint can be simplified using 2.13. This leads to the following constraint formulation:

$$C^{\text{fixed}}_{\text{rot,i}}(\boldsymbol{u}) = \int_\Omega \boldsymbol{X} \times \boldsymbol{u} \cdot \boldsymbol{e}_i \, dV = 0 \tag{2.17}$$

The corresponding variational form becomes:

$$\mathcal{L}^{\text{fixed}}_{\text{rot,i}}(\boldsymbol{u}, \lambda_{\text{rot,i}}) = \Pi(\boldsymbol{u}) - \lambda_{\text{rot,i}} \int_\Omega \boldsymbol{X} \times \boldsymbol{u} \cdot \boldsymbol{e}_i \, dV \tag{2.18}$$

This formulation is particularly efficient in numerical implementations as it directly enforces the conservation of angular momentum while accounting for the pre-existing translational constraints.

More generally, rigid body constraints can be mathematically implemented through Lagrange multipliers in the variational formulation:

$$\mathcal{L}_c(\boldsymbol{u}, \boldsymbol{\lambda}) = \Pi(\boldsymbol{u}) + \sum_{i=1}^{m} \lambda_i C_i(\boldsymbol{u}) \tag{2.19}$$

where λ_i represents the Lagrange multiplier associated with constraint $C_i(\boldsymbol{u})$, and m is the number of constraints.

(b) **Contact constraints** model interactions with external surfaces, such as the ground or adjacent structures. These constraints are typically formulated as inequality constraints:

$$d_{\text{gap}}(\boldsymbol{X}, \boldsymbol{u}) \geq 0 \tag{2.20}$$

where d_{gap} represents the gap function measuring the proximity to the contact surface. The constraint becomes active when $d_{\text{gap}} = 0$, at which point additional terms enter the virtual work functional to account for contact forces.

In numerical implementations, these constraints often necessitate specialized solution strategies, such as the augmented Lagrangian method or penalty formulations, to ensure accurate satisfaction of the constraint conditions.

3. **Gravitational Effects**

Gravity is modeled as a body force distributed throughout the material volume. The virtual work contribution due to gravitational effects can be expressed as:

$$W_{\text{grav}}(\boldsymbol{u}) = \int_{\Omega} \rho(\boldsymbol{X}) \boldsymbol{g} \cdot \boldsymbol{u} \, dV \tag{2.21}$$

where $\rho(\boldsymbol{X})$ represents the position-dependent mass density and $\boldsymbol{g}$ denotes the gravitational acceleration vector. In terrestrial biological systems, gravitational effects can significantly influence growth patterns and resulting morphologies, particularly in plant systems where gravitropism—directional growth in response to gravity—plays a crucial role in developmental processes.

4. **Elastic Response**

The elastic response of a material determines how it accommodates deformations and distributes internal stresses. This response is characterized by constitutive models that relate the deformation state to the stored elastic energy density $\Psi(\mathbf{F}_{\text{e}}, \boldsymbol{X})$.

Several constitutive models of varying complexity are employed in growth mechanics:

(a) **Linear elastic models** assume a linear relationship between stress and strain, valid for small deformations [13]:

$$\Psi(\mathbf{F}_\mathrm{e}) = \frac{1}{2}\lambda(\operatorname{tr}\mathbf{E}_\mathrm{e})^2 + \mu\operatorname{tr}(\mathbf{E}_\mathrm{e}^2) \tag{2.22}$$

where $\mathbf{E}_\mathrm{e} = \frac{1}{2}(\mathbf{F}_\mathrm{e}^T\mathbf{F}_\mathrm{e} - \mathbf{I})$ is the elastic Green-Lagrange strain tensor, and λ and μ are the Lamé parameters characterizing the material's elastic properties. The first Lamé parameter λ quantifies the material's resistance to deformation and relates to the bulk modulus K through $\lambda = K - \frac{2\mu}{3}$. The second Lamé parameter μ, also known as the shear modulus G, characterizes the material's resistance to shear deformation. These parameters can be expressed in terms of the more commonly used engineering constants, Young's modulus E and Poisson's ratio ν, through the following relations:

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \quad \mu = \frac{E}{2(1+\nu)} \tag{2.23}$$

For infinitesimal deformations, this strain tensor can be linearized by expanding $\mathbf{F}_\mathrm{e} = \mathbf{I} + \nabla\boldsymbol{u}$, yielding [13]:

$$\mathbf{E}_\mathrm{e} \approx \frac{1}{2}(\nabla\boldsymbol{u}_\mathrm{e} + \nabla\boldsymbol{u}^T) = \boldsymbol{\varepsilon}_\mathrm{e} \tag{2.24}$$

where $\boldsymbol{\varepsilon}_\mathrm{e}$ is the infinitesimal strain tensor and $\nabla\boldsymbol{u}_\mathrm{e}$ represents the elastic displacement gradient.

(b) **Neo-Hookean models** extend to the nonlinear regime, providing a more accurate representation for materials undergoing moderate deformations [14, 15]:

$$\Psi(\mathbf{F}_\mathrm{e}) = \frac{\mu}{2}(I_1 - 3 - 2\ln J_\mathrm{e}) + \frac{\lambda}{2}(\ln J_\mathrm{e})^2 \tag{2.25}$$

where $I_1 = \operatorname{tr}(\mathbf{F}_\mathrm{e}^T\mathbf{F}_\mathrm{e})$ is the first invariant of the elastic right Cauchy-Green deformation tensor, and $J_\mathrm{e} = \det(\mathbf{F}_\mathrm{e})$ quantifies volumetric changes in the elastic deformation.

(c) **Anisotropic elastic models** incorporate directional variations in material properties, essential for tissues with fiber architectures:

$$\Psi(\mathbf{F}_\mathrm{e}) = \Psi_\text{iso}(\mathbf{F}_\mathrm{e}) + \sum_{i=1}^{n} \Psi^i_\text{aniso}(\mathbf{F}_\mathrm{e}, \boldsymbol{a}_i) \tag{2.26}$$

where Ψ_iso represents the isotropic contribution, Ψ^i_aniso captures the anisotropic response along preferred direction $\boldsymbol{a}_i$, and n denotes the number of fiber families. Here, the material response changes with change in material properties at the location and the direction of any fibre bundles present.

The selection of the constitutive model depends on both the material's properties and the deformation magnitude. Under continuum mechanics, finite strain theory, and elastic conditions, Neo-Hookean constitutive models are commonly employed due to their ability to capture nonlinear material responses while maintaining mathematical tractability. For these reasons, we also use Neo-Hookean constitutive models for all the elastic response modeling in the following chapters.

2.2.4 Constitutive models for growth

The growth field locally at each material point is specified by $\mathbf{F_g}$ and the interplay of growth fields between neighboring material patches creates residual stress that drives deformation (Figure 2.5 C). We now outline key constitutive approaches used to define the growth tensor $\mathbf{F_g}$ [3, 16, 17]:

1. **Isotropic growth models** describe uniform expansion or contraction in all directions (Figure 2.5 A):

$$\mathbf{F}_\mathrm{g} = (1+g)\mathbf{I} \tag{2.27}$$

where the scalar parameter $g > 0$ indicates growth, $g < 0$ indicates contraction, and $g = 0$ represents no growth. In matrix form, the isotropic growth tensor can be represented as:

$$\mathbf{F}_{\mathrm{g}} = \begin{cases} \begin{bmatrix} 1+g & 0 \\ 0 & 1+g \end{bmatrix} & \text{for 2D systems} \\ \\ \begin{bmatrix} 1+g & 0 & 0 \\ 0 & 1+g & 0 \\ 0 & 0 & 1+g \end{bmatrix} & \text{for 3D systems} \end{cases} \tag{2.28}$$

2. **Anisotropic growth models** capture directionally dependent growth processes (Figure 2.5 B):

$$\mathbf{F}_{\mathrm{g}} = \mathbf{I} + g_1 \boldsymbol{e}_1 \otimes \boldsymbol{e}_1 + g_2 \boldsymbol{e}_2 \otimes \boldsymbol{e}_2 + g_3 \boldsymbol{e}_3 \otimes \boldsymbol{e}_3 \tag{2.29}$$

where g_i represents the growth factor along principal direction $\boldsymbol{e}_i$. This formulation is particularly relevant for tissues with oriented cellular structures or fiber architectures that influence growth directionality. In matrix form, this anisotropic growth tensor can be represented as:

$$\mathbf{F}_{\mathrm{g}} = \begin{cases} \begin{bmatrix} 1+g_1 & 0 \\ 0 & 1+g_2 \end{bmatrix} & \text{for 2D systems} \\ \\ \begin{bmatrix} 1+g_1 & 0 & 0 \\ 0 & 1+g_2 & 0 \\ 0 & 0 & 1+g_3 \end{bmatrix} & \text{for 3D systems} \end{cases} \tag{2.30}$$

When the principal directions do not align with the coordinate axes, a more general representation is required:

$$\mathbf{F}_{\mathrm{g}} = \mathbf{Q}\,\mathrm{diag}(g_1, g_2, g_3)\,\mathbf{Q}^T \tag{2.31}$$

where **Q** is the orthogonal tensor that rotates the coordinate system to align with the principal directions, and diag(g_1, g_2, g_3) is the diagonal tensor containing the principal growth factors.

3. **Growth evolution models** describe the temporal evolution of growth processes:

$$\dot{\mathbf{F}}_{\mathrm{g}} = \mathcal{K}(\boldsymbol{\sigma}, \nabla\boldsymbol{\sigma}, ...)\mathbf{F}_{\mathrm{g}} \tag{2.32}$$

where $\dot{\mathbf{F}}_{\mathrm{g}}$ denotes the material time derivative of the growth tensor, and $\mathcal{K}$ is a tensor-valued function that depends on mechanobiological factors such as stress $\boldsymbol{\sigma}$ or its gradient $\nabla\boldsymbol{\sigma}$. This approach enables modeling of mechanosensitive growth responses where tissue growth responds dynamically to the local mechanical environment.

4. **Chemomechanical models** incorporate the influence of morphogen concentrations or other biochemical signals on growth processes:

$$\dot{\mathbf{F}}_{\mathrm{g}} = \mathbf{f}(c_1, c_2, ..., c_n, \boldsymbol{X}, t) \tag{2.33}$$

where c_i represents the concentration of the i-th morphogen, coupling chemical signaling with mechanical deformation. This approach is particularly relevant for developmental processes where tissue growth is regulated by diffusing morphogens.

The specification of appropriate growth kinematics and their driving factors constitutes a central challenge in morphomechanical modeling, requiring integration of biological insights with mechanical principles. In this chapter and for the purposes of modeling growth in the thesis, we

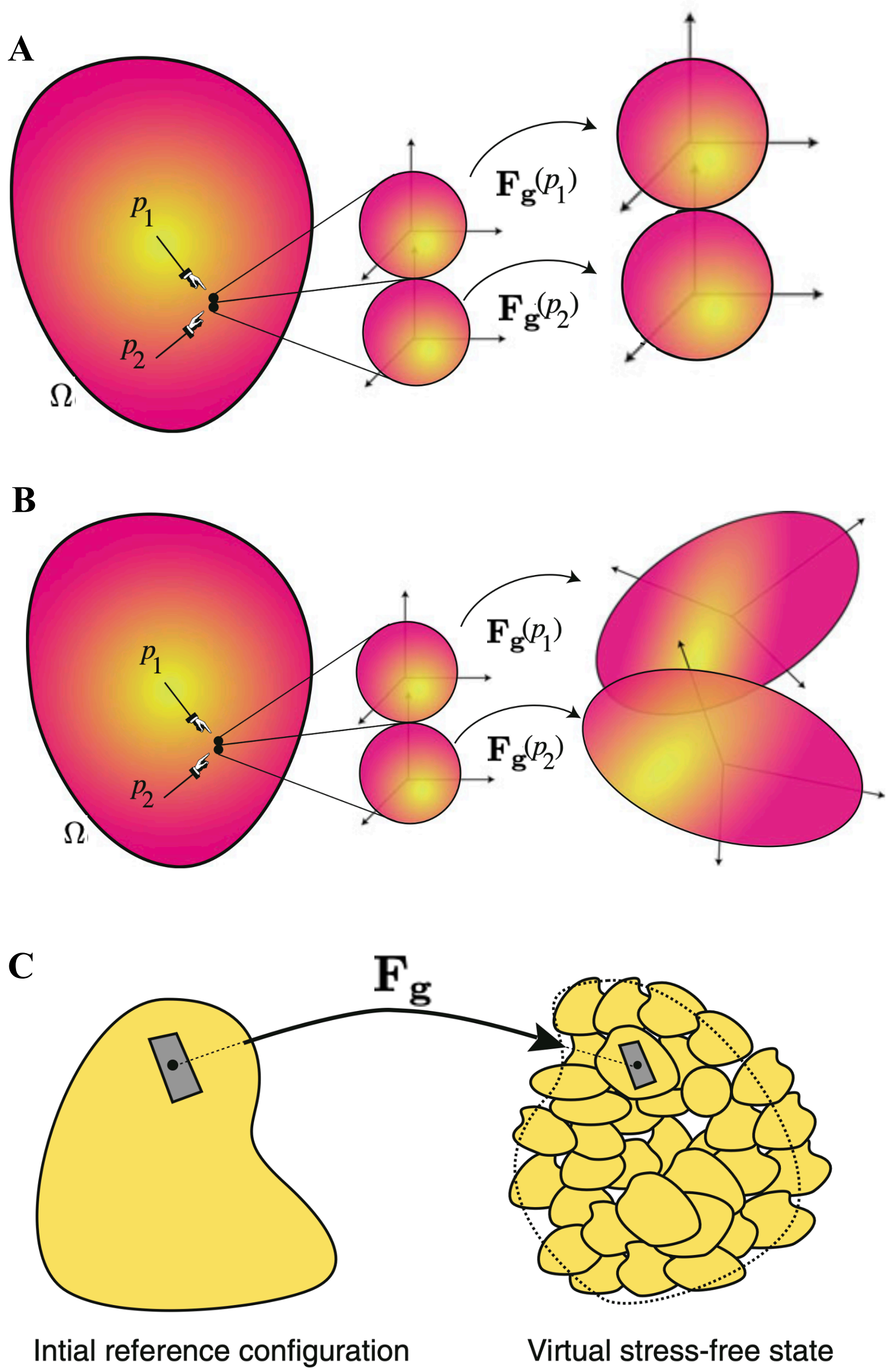


Figure 2.5: **(A-B)** Schematic showing how isotropic and anisotropic growth transforms 2 neighboring material patches, $\boldsymbol{X} = p_1, \boldsymbol{X} = p_2$ in the domain Ω. The patches of spherical volumes maintain their shape in isotropic growth but transform into ellipsoids through anisotropic growth, the sizes along the axes are dependent on growth strains in different directions. **(C)** Neighboring patches transform as a result of internal growth through $\mathbf{F_g}$ to result in a virtual stress-free state where transformed material patches can overlap and there's no residual stress [3].

consider growth to be prescribed as isotropic or anisotropic with time-varying inputs determining the specific values.

2.2.5 Integrated Perspective

The factors influencing deformation in growing systems do not operate in isolation but rather form an interconnected network of mechanical influences. The resulting morphologies emerge from the complex interplay between passive mechanical responses and active growth processes, mediated by the specific material properties and boundary conditions characterizing the system. This integrated perspective is mathematically captured in the overall energy functional:

$$\Pi(\boldsymbol{u}, \boldsymbol{\lambda}) = \int_{\Omega} \Psi(\mathbf{F}_{\mathrm{e}}(\boldsymbol{u}, \mathbf{F}_{\mathrm{g}}), \boldsymbol{X})\, dV - W_{\mathrm{ext}}(\boldsymbol{u}) - W_{\mathrm{c}}(\boldsymbol{u}) \tag{2.34}$$

where the elastic deformation gradient $\mathbf{F}_{\mathrm{e}}$ depends on both the displacement field $\boldsymbol{u}$ and the growth tensor $\mathbf{F}_{\mathrm{g}}$, establishing a fundamental coupling between growth kinematics and elastic deformation [18]. This framework connects mechanics and morphogenesis, enabling shape prediction from physical and biological inputs.

2.3 Finite Element Formulation for Growth Mechanics

The mathematical framework established in the preceding sections provides a continuum description of growth-induced deformation. To solve practical problems within this framework, we must transition from the continuous formulation to a discretized representation suitable for numerical implementation. The finite element method (FEM) offers a systematic approach for this discretization, enabling the solution of complex boundary value problems with mechanics characterized by nonlinear material behavior, heterogeneous growth fields, and complex geometries [19, 20]. This section presents a rigorous finite element formulation for growth-induced deformation problems.

2.3.1 Posing the Variational Problem

The variational form of the energy minimization problem, as described in Section 2.2, seeks the displacement field $\boldsymbol{u}$ that minimizes the total potential energy functional:

$$\Pi(\boldsymbol{u}) = \int_{\Omega} \Psi(\mathbf{F}_{\mathrm{e}}(\boldsymbol{u}, \mathbf{F}_{\mathrm{g}}), \boldsymbol{X})\, dV - W_{\mathrm{ext}}(\boldsymbol{u}) - W_{\mathrm{c}}(\boldsymbol{u}) \tag{2.35}$$

where Ψ represents the strain energy density function, $\mathbf{F}_{\mathrm{e}}$ is the elastic deformation gradient derived from the total deformation gradient $\mathbf{F}$ and the prescribed growth tensor $\mathbf{F}_{\mathrm{g}}$, while W_{ext} and W_{c} denote the work done by external forces and contact forces, respectively. The necessary condition for minimization requires that the first variation vanishes:

$$\delta\Pi(\boldsymbol{u}) = 0 \tag{2.36}$$

The first variation $\delta\Pi(\boldsymbol{u})$ represents the directional derivative of the functional Π along an arbitrary perturbation $\delta\boldsymbol{u}$ and is defined as:

$$\delta\Pi(\boldsymbol{u}) = \frac{d}{d\epsilon}\Pi(\boldsymbol{u} + \epsilon\delta\boldsymbol{u})\bigg|_{\epsilon=0} \tag{2.37}$$

This condition in Equation (2.36) is analogous to setting the derivative equal to zero in classical optimization and constitutes a necessary condition for the displacement field $\boldsymbol{u}$ to minimize the total potential energy. Physically, it corresponds to the principle of virtual work, stating that at equilibrium, the virtual work done by all forces must vanish for any virtual displacement.

Handling Lagrange Constraints

In growth mechanics problems, various constraints may be necessary to enforce physical conditions such as preservation of rigid body motion or maintenance of specific geometric conditions. When such constraints are enforced through Lagrange multipliers, as discussed in Section 2.2.3, the function being minimized is no longer the potential energy $\Pi(\boldsymbol{u})$ alone, but rather the augmented functional:

$$\mathcal{L}(\boldsymbol{u},\boldsymbol{\lambda}) = \Pi(\boldsymbol{u}) + \sum_{i=1}^{m} \lambda_i C_i(\boldsymbol{u}) \tag{2.38}$$

where $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, ..., \lambda_m) \in \mathbb{R}^m$ represents the vector of Lagrange multipliers associated with the m constraint functions $C_i(\boldsymbol{u}) : \mathcal{U} \to \mathbb{R}$, each requiring $C_i(\boldsymbol{u}) = 0$. In this case, the stationarity condition requires that the first variations with respect to both $\boldsymbol{u}$ and $\boldsymbol{\lambda}$ vanish, leading to the system:

$$\delta_{\boldsymbol{u}}\mathcal{L}(\boldsymbol{u},\boldsymbol{\lambda}) = 0 \quad \text{and} \quad \delta_{\boldsymbol{\lambda}}\mathcal{L}(\boldsymbol{u},\boldsymbol{\lambda}) = 0 \tag{2.39}$$

The second condition enforces the constraints $C_i(\boldsymbol{u}) = 0$ for $i = 1, 2, ..., m$, while the first yields the equilibrium equations modified by the constraint forces.

In the subsequent numerical approximation using the Finite Element Method, we will initially present the formulation in terms of the displacement field $\boldsymbol{u}$ as the unknown field that minimizes Equation (2.35). For implementations involving Lagrange constraints, we extend this formulation by replacing $\boldsymbol{u}$ with the augmented unknown function $\boldsymbol{s} = (\boldsymbol{u},\boldsymbol{\lambda}) \in \mathcal{U} \times \mathbb{R}^m$ that minimizes Equation (2.38), notably increasing the dimensionality of the solution space.

2.3.2 Approximating on basis functions

In the finite element method, we approximate the continuous displacement field $\boldsymbol{u}(\boldsymbol{X}) \in \mathcal{U}$ using a finite-dimensional function space spanned by a set of basis functions or shape functions $\{\boldsymbol{U}_I(\boldsymbol{X})\}_{I=1}^{N}$, where N is the number of global degrees of freedom [21]. This approximation takes the form:

$$\boldsymbol{u}(\boldsymbol{X}) \approx \boldsymbol{u}^{\mathrm{h}}(\boldsymbol{X}) = \sum_{I=1}^{N} v_I \boldsymbol{U}_I(\boldsymbol{X}) \tag{2.40}$$

where $v_I \in \mathbb{R}$ represents the coefficient associated with the I-th global degree of freedom, and $\boldsymbol{U}_I(\boldsymbol{X})$ is the corresponding shape function. The shape functions satisfy the partition of unity property and possess the Kronecker delta property at the mesh nodes, ensuring interpolation of the nodal values.

Choosing linear basis functions

The choice of basis or shape functions determines the interpolation order and consequently affects both the accuracy and computational efficiency of the finite element solution. While higher-order elements can provide improved convergence rates for smooth solutions, we employ linear shape functions (Figure 2.6 A-B, Figure 2.7 D) in our simulations for several compelling reasons:

1. Linear elements reduce the number of degrees of freedom (Figure 2.6 D-F, Figure 2.7 B-C), enhancing computational efficiency for large-scale simulations characteristic of the growth mechanics problems being solved.

2. The fine meshing required to capture geometric complexities in the concerned biological or synthetic structures typically satisfies the continuum assumption, mitigating accuracy losses from linear interpolation.

3. For linear finite elements, there is a direct correspondence between the nodal displacement values $\boldsymbol{u}(\boldsymbol{X})$ and the degrees of freedom coefficients v_I introduced in Equation (2.40) (Figures 2.6 C, 2.7 E). This relationship can be expressed explicitly for each component as:

$$v_{ni+j+1} = \boldsymbol{u}(\boldsymbol{X}_i)_j, \quad i = 0, 1, \ldots, n_{\text{nodes}} - 1, \quad j = 0, 1, \ldots, n - 1 \tag{2.41}$$

where $\boldsymbol{u}(\boldsymbol{X}_i)_j$ represents the j-th component of the displacement vector at node i, and v_{ni+j+1} is the corresponding entry in the global coefficient vector $\boldsymbol{v} \in \mathbb{R}^N$. The relationship between the number of global degrees of freedom N, the dimensionality of the displacement field n (typically $n = 2$ for planar problems and $n = 3$ for spatial problems),

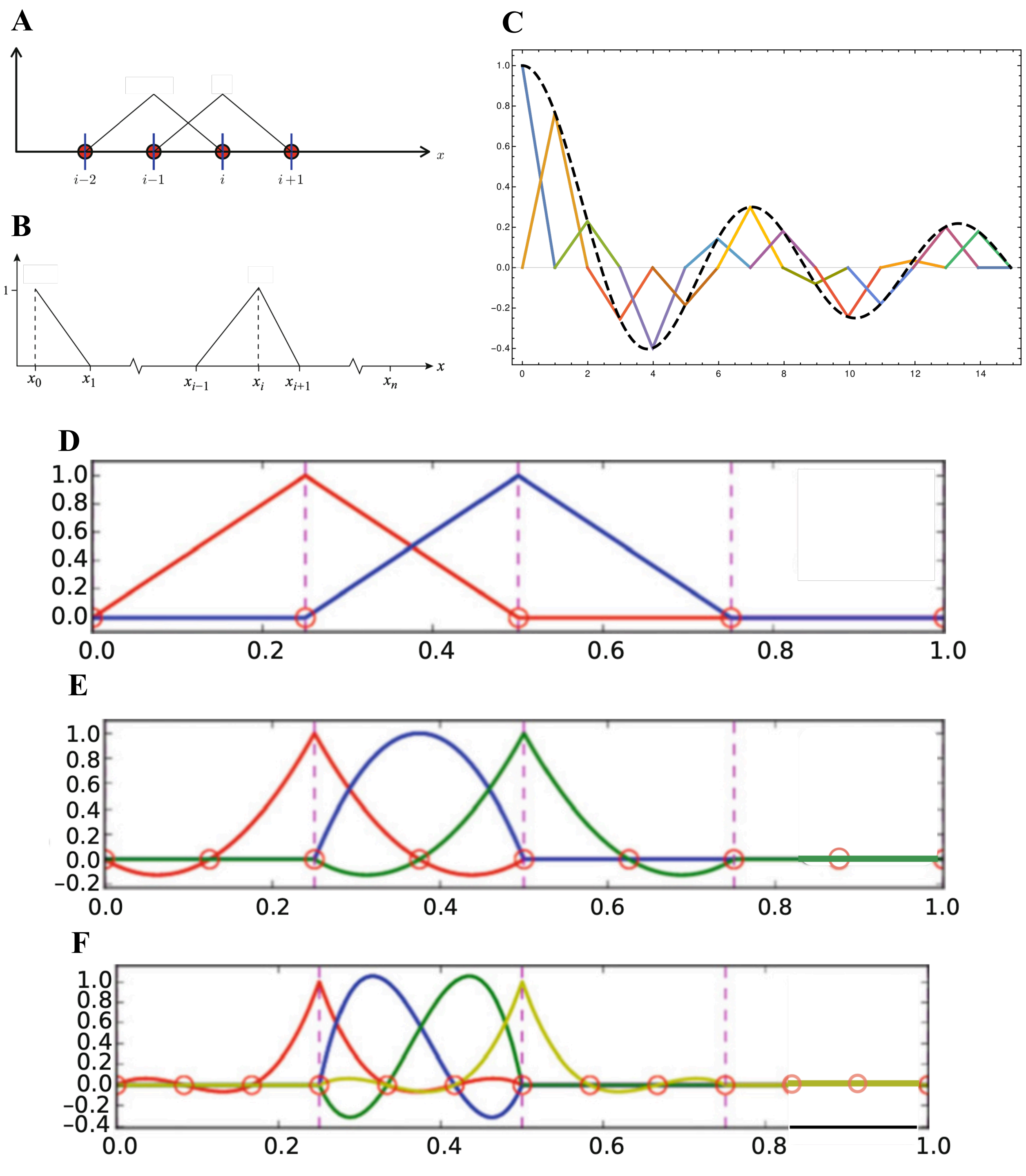


Figure 2.6: **(A, B)** Linear 1D shape functions, hat function, are the basis to approximate a 1D function. Neighboring hat functions and hat function at the boundary is shown [21, 20]. **(C)** A linear function approximated using 1D linear shape functions. **(D-F)** Comparing linear(D), quadratic(E), and cubic(F) 1D shape functions in the same domain. Note the increase in the number of shape functions, and their corresponding coefficients, from 2, 3, 4 to represent a function in the same interval of [0.25, 0.50] [20].

and the number of nodes n_{nodes} is given by:

$$N = n \times n_{\text{nodes}} \tag{2.42}$$

This one-to-one mapping eliminates complex transformations between the physical and computational representations of the displacement field, which can become computationally intensive because of the large dimension of these representations. Thus, choosing linear elements simplifies implementation, especially for nonlinear materials which have more transformations between displacement representations.

2.3.3 Computing deformation

In the finite element formulation of growth mechanics, a fundamental consideration is how the element-level deformation gradient relates to the global deformation field. Unlike nodal quantities such as displacements, the deformation gradient is not directly assembled through conventional assembly operators. Instead, its treatment requires careful consideration of the relationship between local and global kinematic quantities, particularly in the context of the multiplicative decomposition characteristic of growth mechanics.

Deformation in the element

The continuous domain Ω is discretized into a collection of non-overlapping finite elements Ω^e, such that $\Omega = \cup_{e=1}^{n_{\text{elem}}} \Omega^e$, each contributing to the global solution. Within this discretized framework, the global displacement field $\boldsymbol{u}(\boldsymbol{X})$ is represented through the interpolation of nodal values using global shape functions as given in Equation (2.40). These global shape functions $\boldsymbol{U}_I(\boldsymbol{X})$ are constructed by appropriately combining the element shape functions through the connectivity mapping between local and global degrees of freedom.
At the element level, the displacement field within an element Ω^e is expressed using local shape functions:

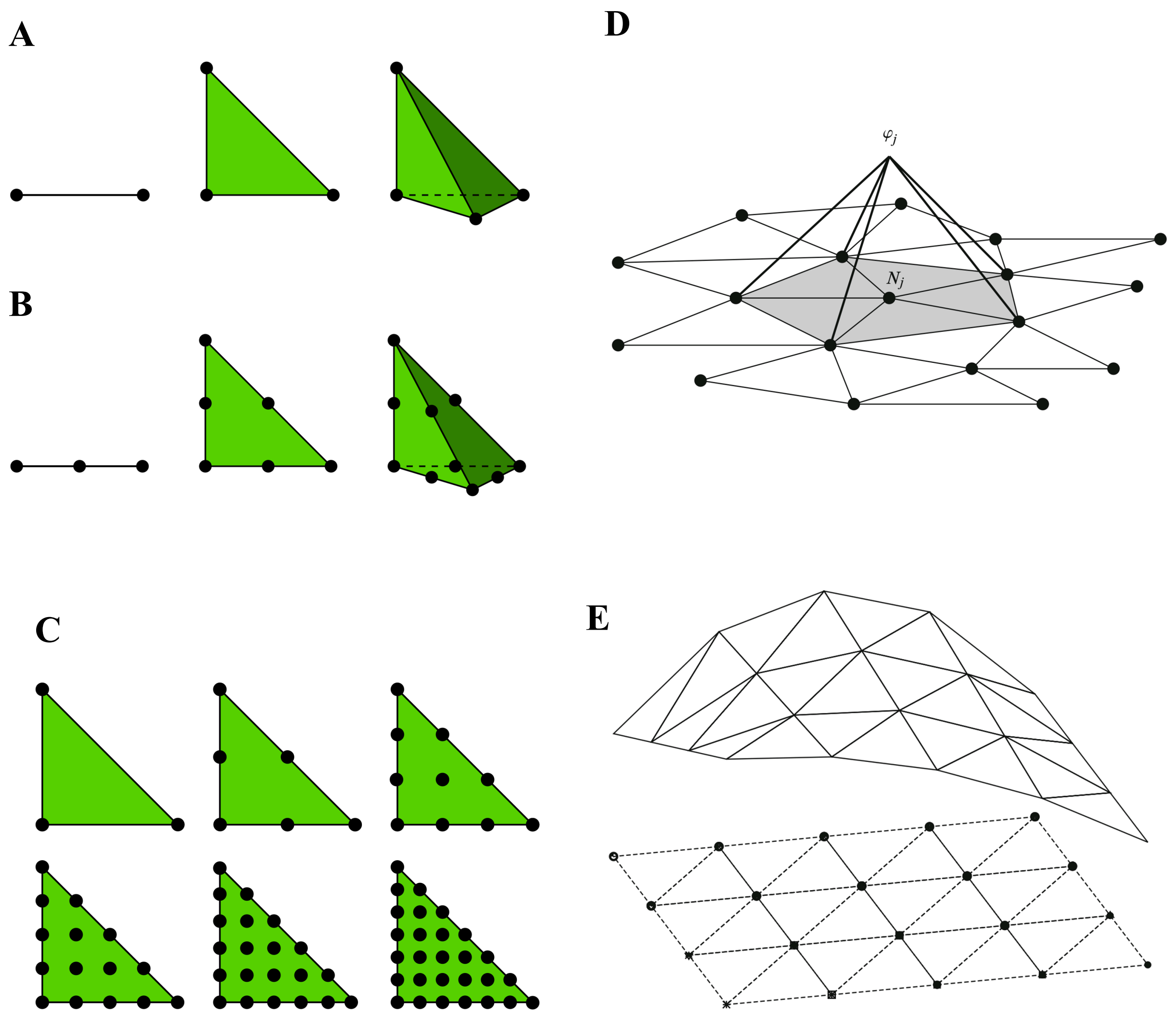


Figure 2.7: **(A)** Linear elements or first-order elements in 1D, 2D, and 3D on whose nodes shape functions are defined [22]. **(B)** Quadratic(2nd order) elements in 1D, 2D, and 3D on whose nodes shape functions are defined [22]. **(C)** 2D elements of order p = 1, 2, 3, 4, 5, 6 on whose nodes the shape functions for 2D meshes are defined [22]. **(D)** Linear shape function defined on 2D finite element i.e 2D hate function [21]. **(E)** Example of a scalar function approximated using linear 2D shape functions on a 2D mesh [20].

$$\boldsymbol{u}^e(\boldsymbol{X}) = \sum_{a=1}^{n_e} v_a^{\mathrm{e}} \boldsymbol{U}_a^{\mathrm{e}}(\boldsymbol{X}) \tag{2.43}$$

where n_e is the number of nodes per element, $\boldsymbol{U}_a^{\mathrm{e}}(\boldsymbol{X})$ are the element shape functions, and v_a^{e} are the coefficients or nodal displacements associated with the element. These element nodal displacements correspond to entries in the global displacement vector through the element connectivity mapping, which defines the relationship between local node indices a and global node indices I.

The deformation gradient at a material point $\boldsymbol{X}$ within element Ω^e is computed by taking the gradient of the displacement field and adding the identity tensor [23]:

$$\mathbf{F}(\boldsymbol{X}) = \mathbf{I} + \nabla \boldsymbol{u}(\boldsymbol{X}) = \mathbf{I} + \sum_{a=1}^{n_e} v_a^{\mathrm{e}} \nabla \boldsymbol{U}_a^{\mathrm{e}}(\boldsymbol{X}) \tag{2.44}$$

where $\nabla \boldsymbol{U}_a^{\mathrm{e}}(\boldsymbol{X}) = \frac{\partial \boldsymbol{U}_a^{\mathrm{e}}(\boldsymbol{X})}{\partial \boldsymbol{X}}$ represents the gradient of the shape function with respect to the reference coordinates. This expression reveals that the deformation gradient is evaluated pointwise based on the gradient of the shape functions and the nodal displacement values. The deformation gradient thus represents a derived field quantity rather than a primary unknown in the finite element formulation.

Assembling deformation

Our choice of linear shape elements for their advantages in Section 2.3.2 has the additional advantage of making the assembled deformation tensor a more sparse matrix when implemented numerically (Figure 2.8). The higher quality sparse matrices, arising from ordering degrees of freedom corresponding to the displacement of nodes, leads to computationally faster operations when compared to lower quality sparse matrices from irregular ordering of degrees of freedom.

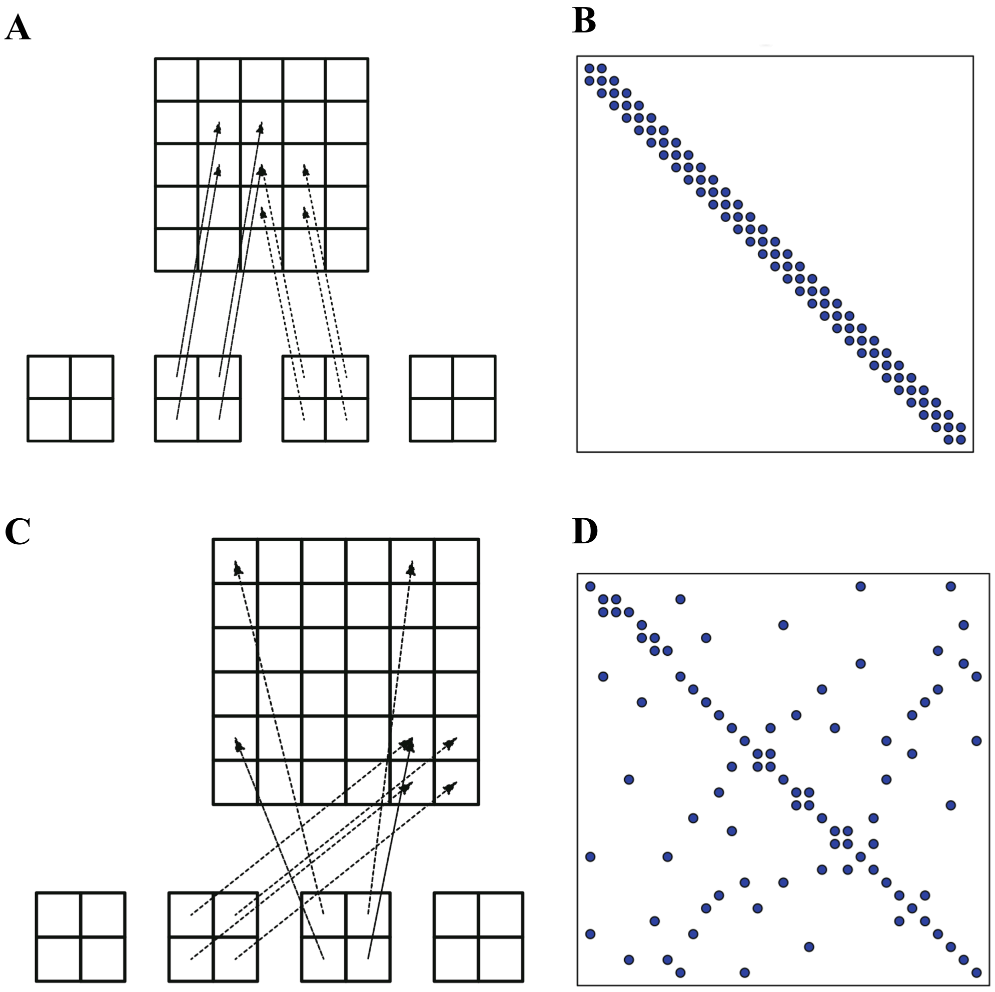


Figure 2.8: **(A)** Illustration of matrix assembly for regularly numbered degrees of freedom mapped in order to node displacements. **(B)** Matrix sparsity pattern for regularly numbered degrees of freedom. **(C)** Illustration of matrix assembly for irregularly numbered degrees of freedom not directly mapped to node displacements. **(D)** Matrix sparsity pattern for irregularly numbered degrees of freedom.[20]

Continuity of deformation fields

Global consistency of the deformation gradient across element boundaries is maintained through two key mechanisms :

1. **Displacement Continuity**: The conforming nature of the finite element basis functions ensures that the displacement field $\boldsymbol{u}(\boldsymbol{X})$ is continuous across element boundaries (for standard C^0 elements). This continuity is critical for physical consistency of the deformation field.

2. **Weak Continuity of Derivatives**: While the gradients of the shape functions, and consequently the deformation gradient, may exhibit discontinuities across element boundaries, the variational formulation ensures that the effects of these discontinuities are appropriately integrated into the global system, resulting in physically consistent solutions.

Calculating elastic deformation gradient

In the context of growth mechanics, the computation of the deformation gradient is further nuanced by the multiplicative decomposition:

$$\mathbf{F}(\boldsymbol{X}) = \mathbf{F}_{\mathrm{e}}(\boldsymbol{X})\mathbf{F}_{\mathrm{g}}(\boldsymbol{X}) \tag{2.45}$$

where $\mathbf{F}_{\mathrm{g}}(\boldsymbol{X})$ represents the prescribed growth tensor that may vary spatially according to biological or physical growth laws. The elastic deformation gradient $\mathbf{F}_{\mathrm{e}}(\boldsymbol{X}) = \mathbf{F}(\boldsymbol{X}) \cdot \mathbf{F}_{\mathrm{g}}^{-1}(\boldsymbol{X})$ is then used to compute the stress tensor and strain energy density at each integration point within the elements according to the constitutive model selected to construct the total potential energy as described in Section 2.2.

The evaluation of the growth tensor $\mathbf{F}_{\mathrm{g}}(\boldsymbol{X})$ at integration points requires careful consideration, particularly when the growth field exhibits spatial heterogeneity or temporal evolution. In our implementation, the growth tensor is specified through analytical functions or discrete field data that are interpolated to the integration points. This approach allows for a flexible representation of complex growth patterns while maintaining computational tractability.

The pointwise evaluation of the deformation gradient, with global consistency in the finite element formulation, along with the multiplicative decomposition of deformation is essential for constructing the total potential energy function, given in Equation (2.35), across the discretized domain.

2.3.4 Nonlinear equations for equilibrium

Substituting the approximation in Equation (2.40) into the total potential energy functional Equation in (2.35) and recognizing that the variations $\delta \boldsymbol{u}$ can be expressed in terms of variations in the coefficients δv_I, we obtain the approximate form of the function to be minimized [22]:

$$\Pi(\boldsymbol{u}) \approx \Pi(\boldsymbol{u}^h) = \Pi\left(\sum_{I=1}^{N} v_I \boldsymbol{U}_I(\boldsymbol{X})\right) \tag{2.46}$$

The global vector of coefficients $\boldsymbol{v}$ collects all degrees of freedom:

$$\boldsymbol{v} = (v_1, v_2, ..., v_N), \boldsymbol{v} \in \mathbb{R}^N \tag{2.47}$$

The stationarity condition for the discrete problem becomes:

$$\delta\Pi(\boldsymbol{u}) \approx \delta\Pi(\boldsymbol{u}^h) = 0 \implies \frac{d}{d\epsilon}\Pi(\boldsymbol{u}^h + \epsilon\delta\boldsymbol{u}^h)\bigg|_{\epsilon=0} = 0 \tag{2.48}$$

$$\implies \frac{d}{d\epsilon}\Pi\left(\sum_{I=1}^{N} \boldsymbol{U}_I(\boldsymbol{X})(v_I + \epsilon\delta v_I)\right)\Bigg|_{\epsilon=0} = \sum_{I=1}^{N} \delta v_I \cdot R_I(\boldsymbol{v}) = 0 \tag{2.49}$$

where $R_I(\boldsymbol{v})$ represents the residual vector associated with the I-th degree of freedom. Since Equation (2.49) must hold for arbitrary variations δv_I, we obtain the nonlinear system of equations:

$$R_I(\boldsymbol{v}) = \mathbf{0}, \quad I = 1, 2, \ldots, N \tag{2.50}$$

which constitutes the discretized equilibrium condition. The residual vector R_I can be decomposed into internal and external contributions:

$$R_I(\boldsymbol{v}) = f_I^{\text{int}}(\boldsymbol{v}) - f_I^{\text{ext}}(\boldsymbol{v}) \tag{2.51}$$

where f_I^{int} represents the internal force vector from elastic strain energy and f_I^{ext} denotes the external force vector resulting from applied loads and boundary conditions, both associated with the corresponding degree of freedom I.

Through this finite element formulation, we have effectively transformed the problem of solving for a continuous displacement field to one of determining a finite set of coefficients $\boldsymbol{v} = \{v_1, v_2, ..., v_N\}$. The nonlinear system defined by Equation (2.50) will be solved using the numerical algorithms detailed in Section 2.4. Once the solution $\boldsymbol{v}$ is obtained, the continuous displacement field is constructed through:

$$\boldsymbol{u}_{\text{solution}}(\boldsymbol{X}) = \sum_{I=1}^{N} v_I \boldsymbol{U}_I(\boldsymbol{X}) \tag{2.52}$$

For implementations involving Lagrange constraints, the unknown function vector $\boldsymbol{s} = (\boldsymbol{u}, \boldsymbol{\lambda})$ is approximated as $\boldsymbol{s}^h(\boldsymbol{v}, \boldsymbol{\lambda})$, where each Lagrange multiplier $\lambda_j \in \mathbb{R}$ contributes an additional degree of freedom to the system.

2.4 Numerical Solvers for Growth-Induced Deformation

The finite element formulation presented in Section 2.3 transforms the continuous problem of growth-induced deformation into a nonlinear system of algebraic equations. The current section details the specialized numerical strategies developed to solve this discrete system efficiently and robustly, with emphasis on algorithms adapted to handle the unique challenges posed by evolving growth fields and geometric instabilities. Our nonlinear solvers track equilibrium states under evolving growth fields and handle instabilities and geometric nonlinearities.

2.4.1 General Problem Formulation

To obtain the equilibrium deformation state at each stage of growth evolution, we must solve a nonlinear system of equations for the unknown coefficients v_I introduced in Section 2.3:

$$R_I(v_1, v_2, \ldots, v_N) = 0, \quad I = 1, 2, \ldots, N \tag{2.53}$$

or in vector form:

$$\boldsymbol{R}(\boldsymbol{v}) = \boldsymbol{0}, \quad \boldsymbol{v} \in \mathbb{R}^N \tag{2.54}$$

where N represents the total number of degrees of freedom in the assembled system. Once determined, these coefficients enable the reconstruction of the continuous displacement field through interpolation with the basis of shape functions:

$$\boldsymbol{u}_{\text{solution}}(\boldsymbol{X}, e) = \sum_{I=1}^{N} v_I \boldsymbol{U}_I(\boldsymbol{X}) \tag{2.55}$$

The parameter $e \in [0, 1]$ represents a non-dimensional evolution parameter that quantifies the progression of shape change (Figure 2.9), where $e = 0$ corresponds to the initial undeformed state and $e = 1$ represents the final deformed state.

We prescribe driving factors through the evolution parameter e. For example, growth strain $g(\boldsymbol{X})$ at any intermediate state is given by:

$$g(\boldsymbol{X}) = e \cdot g^{\text{final}}(\boldsymbol{X}) \tag{2.56}$$

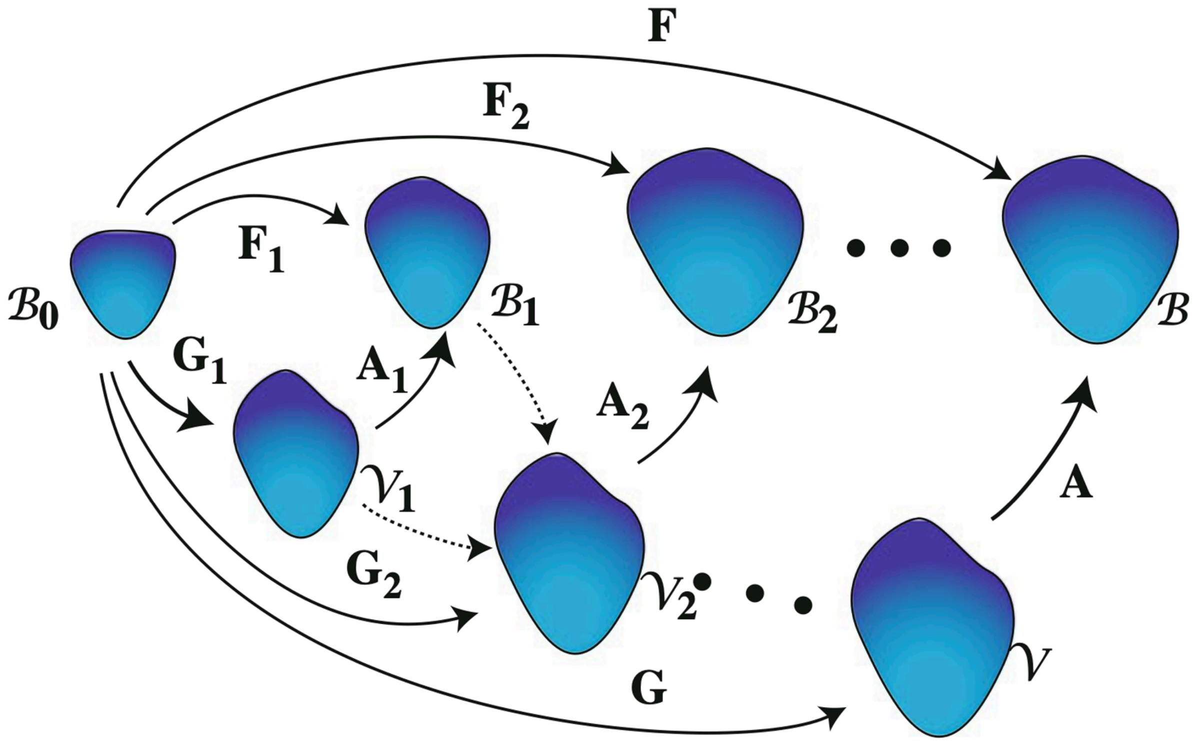


Figure 2.9: The evolution of shape modelled step-wise using deformation decomposition. Here, the growth tensor $\mathbf{F_g}$ changes with the evolution parameter such that $\mathbf{G_i} = \mathbf{F_g}(\boldsymbol{X}, e_i)$ where e_i the different values of evolution parameter. Each of the intermediate states at different values of e is represented by $\mathcal{B}_i$ and the corresponding stress-free virtual states is $\mathcal{V}_i$. Consequently, the resulting deformation at each of the intermediate states is $\mathbf{F}_i = \mathbf{F}(\boldsymbol{X}, e_i)$.[3]

where $g^{\text{final}}(\boldsymbol{X})$ denotes the prescribed growth strain at the final state. We assume the reference configuration has no initial growth strain. Similarly, for problems involving evolving external forces such as pressure or traction, we parameterize the pressure (which can be replaced by external forces) as:

$$p = p^{\text{init}} + e \cdot (p^{\text{final}} - p^{\text{init}}) \tag{2.57}$$

where p^{init} and p^{final} represent the prescribed pressures in the initial and final states, respectively. If there exist multiple driving factors of deformation their change can be captured by the evolution parameter either simultaneously or as sequential evolution steps, based on what aligns with the observations of the morphing system. In the following derivations, we assume only a single driving factor, internal growth, and its corresponding evolution parameter exist, but the derived methods can be extended to other/mutliple evolving driving factors.

This section presents two sophisticated numerical algorithms developed by us for specifically modeling growth-induced large deformations. Both algorithms build upon established nonlinear solvers, viz. Newton's method and the arc-length method, that have been used for static load bearing structures. Using these methods, we developed algorithms to solve for the stages of shape development of morphing structures. We incorporate substantial modifications to address the particular challenges of morphing structures, including following the solution trajectory through geometric instabilities and adaptive stepping strategies. Also presented are further advances that can be optionally implemented to improve computational efficiency by reducing memory usage and compute time.

2.4.2 Modified Newton's Method

Governing System of Equations

The classical Newton's method for solving nonlinear systems iteratively refines an initial guess by linearizing the residual equations around the current solution estimate. For our problem to solve Equation (2.50) for an equilibrium state of $\boldsymbol{v}$, we seek the solution $\boldsymbol{v}^k$ at the k-th iteration that satisfies:

$$\boldsymbol{R}(\boldsymbol{v}^k) = \mathbf{0} \tag{2.58}$$

Assuming an increment $\Delta\boldsymbol{v}^k$ such that:

$$\boldsymbol{R}(\boldsymbol{v}^{k-1} + \Delta\boldsymbol{v}^k) = \mathbf{0}, \quad \text{where} \quad \Delta\boldsymbol{v}^k = \boldsymbol{v}^k - \boldsymbol{v}^{k-1} \tag{2.59}$$

We expand the residual around $\boldsymbol{v}^{k-1}$ using Taylor series and truncate after the linear term:

$$\boldsymbol{R}(\boldsymbol{v}^{k-1}) + \nabla\boldsymbol{R}(\boldsymbol{v}^{k-1}) \cdot \Delta\boldsymbol{v}^k + \mathcal{O}((\Delta\boldsymbol{v}^k)^2) \approx \mathbf{0} \tag{2.60}$$

Introducing the tangent stiffness matrix $\mathbf{K}(\boldsymbol{v}^{k-1}) = \nabla\boldsymbol{R}(\boldsymbol{v}^{k-1})$, results in:

$$\boldsymbol{R}(\boldsymbol{v}^{k-1}) + \mathbf{K}(\boldsymbol{v}^{k-1}) \cdot \Delta\boldsymbol{v}^k \approx \mathbf{0} \tag{2.61}$$

Solving for the increment yields:

$$\Delta\boldsymbol{v}^k = -\mathbf{K}^{-1}(\boldsymbol{v}^{k-1})\boldsymbol{R}(\boldsymbol{v}^{k-1}) \tag{2.62}$$

where $\mathbf{K} \in \mathbb{R}^{N\times N}$ and $\Delta\boldsymbol{v}^k, \boldsymbol{R} \in \mathbb{R}^N$. If $\boldsymbol{R}(\boldsymbol{v}^{k-1} + \Delta\boldsymbol{v}^k) \approx \mathbf{0}$ within a specified tolerance $\epsilon > 0$, $\boldsymbol{v}^k = \boldsymbol{v}^{k-1} + \Delta\boldsymbol{v}^k$ is accepted as the converged solution (Figure 2.10). Otherwise, we proceed to the next iteration, solving for $\Delta\boldsymbol{v}^{k+1}$ to satisfy $\boldsymbol{R}(\boldsymbol{v}^k + \Delta\boldsymbol{v}^{k+1}) = \mathbf{0}$.

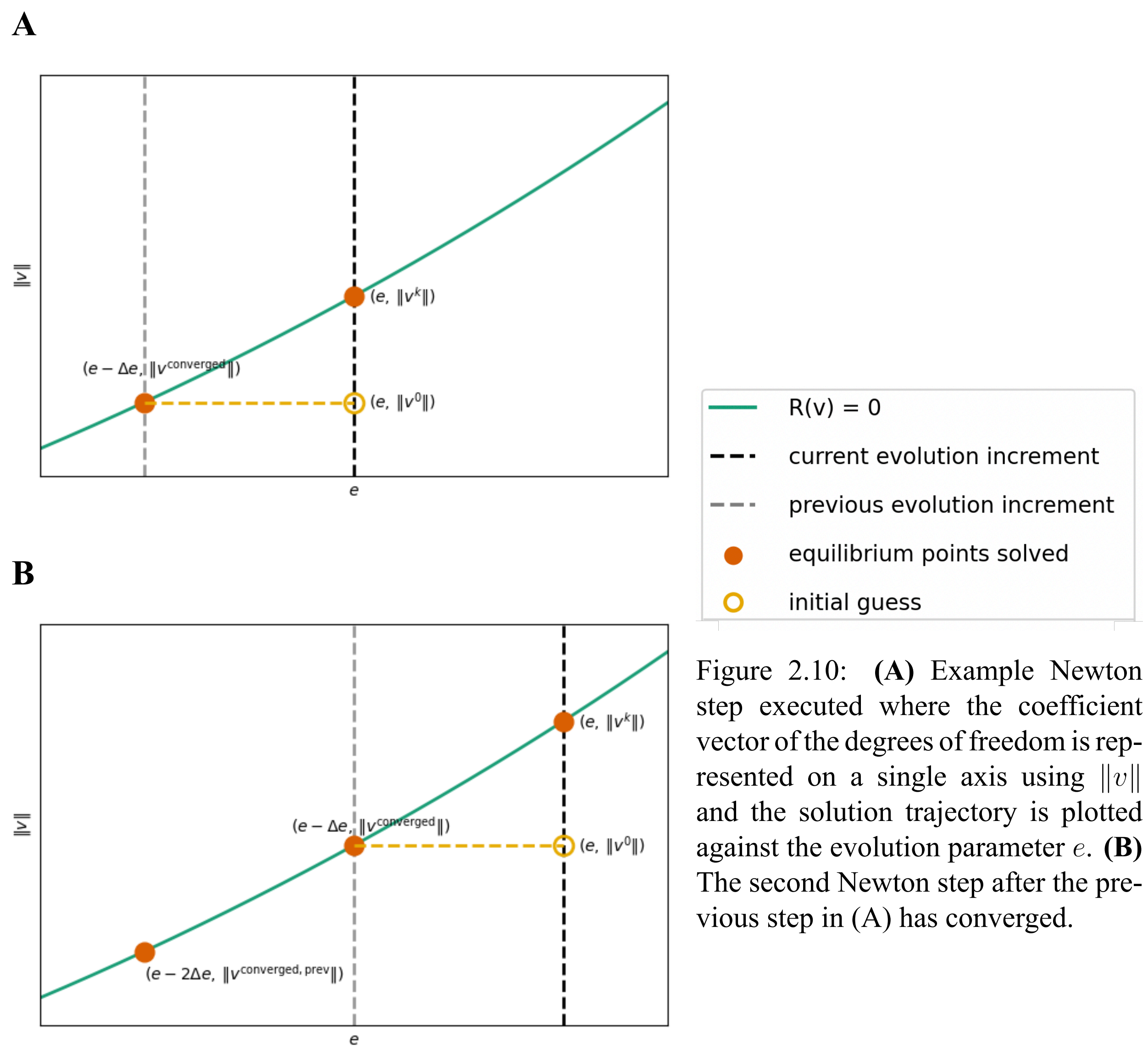


Figure 2.10: **(A)** Example Newton step executed where the coefficient vector of the degrees of freedom is represented on a single axis using $\|v\|$ and the solution trajectory is plotted against the evolution parameter e. **(B)** The second Newton step after the previous step in (A) has converged.

Solving for coefficients at equilibrium

The iterative implementation of the modified Newton's method is detailed in Algorithm 1.

Algorithm 1 Modified Newton's Method for Growth-Induced Deformation

```
1: Initialize coefficient vector $\boldsymbol{v}^0 = \boldsymbol{v}^{\text{init}}$
2: Set convergence tolerance $\epsilon$ and allowable iterations $k^{\max}$
3: Set iteration counter $k = 1$
4: while $\|\boldsymbol{R}(\boldsymbol{v}^{k-1})\| > \epsilon$ and $k \leq k^{\max}$ do
5:     Compute displacement field $\boldsymbol{u}^{k-1} = \sum_{I=1}^{N} v_I^{k-1} \boldsymbol{U}_I(\boldsymbol{X})$
6:     Compute deformation gradient $\mathbf{F}^{k-1} = \mathbf{I} + \nabla \boldsymbol{u}^{k-1}$
7:     Compute elastic deformation gradient $\mathbf{F}_e^{k-1} = \mathbf{F}^{k-1} \cdot (\mathbf{F}_g)^{-1}$
8:     Evaluate residual vector $\boldsymbol{R}(\boldsymbol{v}^{k-1})$
9:     Assemble tangent stiffness matrix $\mathbf{K}(\boldsymbol{v}^{k-1})$
10:    Solve linear system $\mathbf{K}(\boldsymbol{v}^{k-1})\Delta \boldsymbol{v}^k = -\boldsymbol{R}(\boldsymbol{v}^{k-1})$
11:    Update solution: $\boldsymbol{v}^k = \boldsymbol{v}^{k-1} + \Delta \boldsymbol{v}^k$
12:    Increment counter: $k = k + 1$
13: end while
14: if $k \leq k^{\max}$ then
15:     return Converged solution $\boldsymbol{v}^{\text{converged}} = \boldsymbol{v}^k$
16: else return Convergence failed error
17: end if
```

The computational bottleneck in this algorithm is solving the linear system in step 10. We cannot directly invert the tangent stiffness matrix **K** to implement the analytical solution for the increment given in Equation (2.62) because a direct inversion computationally the worst case scenario. So we convert it to a linear solver equation that requires either direct factorization or iterative solution of a potentially large-scale matrix system ($N \times N$). For problems with millions of degrees of freedom, specialized sparse matrix solvers and preconditioning techniques become essential for computational efficiency with the computational complexity ranging between $\mathcal{O}(N)$ and $\mathcal{O}(N^{7/3})$.

Convergence tolerance ϵ is typically set between 10^{-7} and 10^{-10} based on the balance between acceptable displacement errors and computational cost, with tighter tolerances providing more accurate solutions at the expense of increased iteration counts.

Additionally, the iteration counter k is capped at a predefined maximum $k^{\max}$, typically 50-100 iterations, to prevent uncontrolled computational runs in cases where convergence is not achievable.

Growth Evolution Strategy

To capture the complete growth-induced deformation process, we incrementally evolve the growth parameter e from 0 to 1 and solve for the equilibrium configuration at each step:

Algorithm 2 Incremental Growth Evolution with Modified Newton's Method

```
1:  Initialize displacement field: $\boldsymbol{u}(\boldsymbol{X}) = 0$
2:  Initialize evolution parameter: $e = 0$
3:  Set number of evolution increments: $n_{\text{steps}}$
4:  Compute evolution increment: $\Delta e = 1/n_{\text{steps}}$
5:  while $e \leq 1$ do
6:      Compute growth tensor $\mathbf{F}_g$ based on $g = e \cdot g^{\text{final}}$
7:      Initialize coefficient vector $\boldsymbol{v}^{\text{init}}$ from nodal displacement values of $\boldsymbol{u}(\boldsymbol{X})$
8:      Solve for equilibrium coefficients $\boldsymbol{v}^{\text{converged}}$ using Algorithm 1
9:      Update displacement field: $\boldsymbol{u}(\boldsymbol{X}) = \sum_{I=1}^{N} v_I^{\text{converged}} \boldsymbol{U}_I(\boldsymbol{X})$
10:     Store current configuration: $\boldsymbol{u}(\boldsymbol{X}, e) = \boldsymbol{u}(\boldsymbol{X})$
11:     Increment growth parameter: $e = e + \Delta e$
12: end while
```

The selection of the evolution parameter increment size Δe (or equivalently, n_{steps}) involves critical trade-offs that impact both solution accuracy and computational efficiency. Qualitatively, smaller evolution increments enhance numerical stability and convergence reliability by providing initial guesses closer to the solution. Larger evolution increments may lead to convergence difficulties or miss important qualitative changes in the deformation path. Quantitatively, computational cost depends on both the number of evolution steps (n_{steps}) and the number of Newton solver iterations. This creates a non-trivial optimization problem for choosing step size. Large evolution increments reduce n_{steps}, but increase initial residual error at each step, requiring more solver iterations per evolution step. Conversely, with smaller evolution increments, n_{steps} is higher but the initial residual error is lower, requiring fewer iterations per step. Computational cost scales with both the number of evolution steps and the average solver iterations per step, approximately proportional to $n_{\text{steps}} \times n_{\text{avg,iter}}$, where $n_{\text{avg,iter}}$ is the average number of solver iterations per evolution step. In practice, we determine an appropriate increment size through preliminary simulations, typically exploring growth evolution up to $e \approx 0.1$ with different resolution settings of Δe, to balance computational efficiency and the number of intermediate states for each morphing system.

Algorithmic Enhancements

To improve the robustness and efficiency of the growth evolution algorithm, we have implemented several key enhancements:

- **Predictive Initial Guess**

  Instead of simply using the previous converged solution as the initial guess for the next evolution increment, we leverage quantities derived from the converged solution to generate a initial prediction closer to the solution (Figure 2.11):

  $$\boldsymbol{v}^{\text{init}}(e) = \boldsymbol{v}^{\text{converged}}(e - \Delta e) - \mathbf{K}^{-1}(\boldsymbol{v}^{\text{converged}})\boldsymbol{R}(\boldsymbol{v}^{\text{converged}}, e) \tag{2.63}$$

  This implicitly includes the first Newton iteration, reducing solver iterations required for convergence.

- **Adaptive Stepping**

  To handle challenging regions of the solution path without sacrificing overall computational efficiency, we develop an adaptive evolution increment strategy shown in Algorithm 3. We implement this as a code segment within Algorithm 2, added between lines 8 and 9. This approach ensures that the simulation can navigate through numerically challenging regions by automatically reducing the evolution step increment when convergence difficulties are encountered, then restoring the original increment once stable conditions return (Figure 2.12). Reducing Δe increases the likelihood of convergence because the coefficient vector is initialized closer to the expected solution with the new e. The lower bound Δe_{min} prevents excessively small increments while providing a substantial range of adaptive refinement, typically 4-5 orders of magnitude,

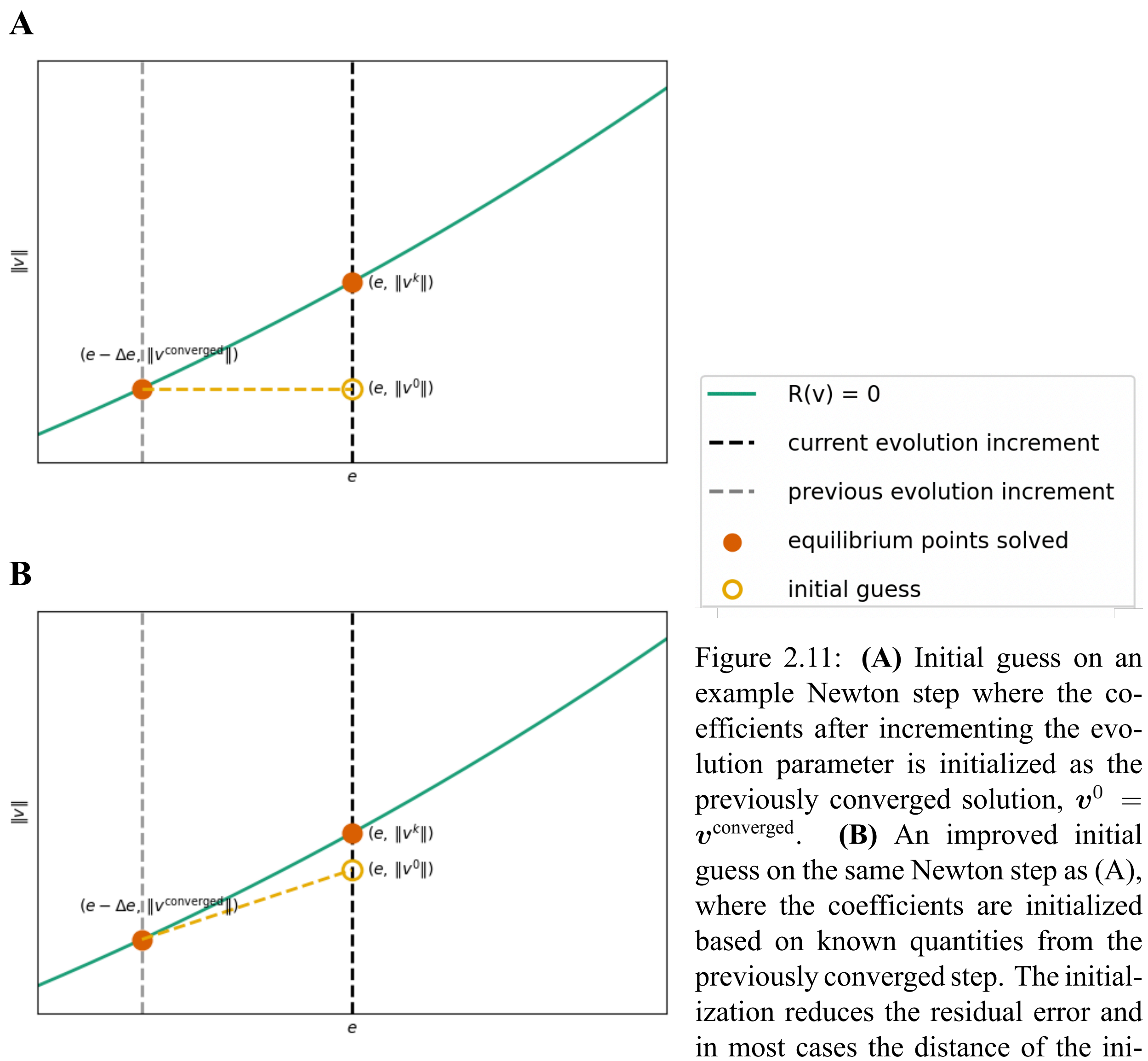

A
$\|v\|$
$(e, \|v^k\|)$
$(e - \Delta e, \|v^{\text{converged}}\|)$
$(e, \|v^0\|)$
e
B
$\|v\|$
$(e, \|v^k\|)$
$(e, \|v^0\|)$
$(e - \Delta e, \|v^{\text{converged}}\|)$
e
R(v) = 0
current evolution increment
previous evolution increment
equilibrium points solved
initial guess

Figure 2.11: **(A)** Initial guess on an example Newton step where the coefficients after incrementing the evolution parameter is initialized as the previously converged solution, $\boldsymbol{v}^0 = \boldsymbol{v}^{\text{converged}}$. **(B)** An improved initial guess on the same Newton step as (A), where the coefficients are initialized based on known quantities from the previously converged step. The initialization reduces the residual error and in most cases the distance of the initial condition from the actual solution leading to faster convergence.

$\Delta e_{\text{min}} = \Delta e \cdot 10^{-4}$ or $\Delta e_{\text{min}} = \Delta e \cdot 10^{-5}$. Retrying convergence with adaptive Δe ensures the simulation continues running even after convergence failure thus preventing manual restarts and the user updating evolution increments.

Algorithm 3 Adaptive Growth Stepping Enhancement

```
1: Set minimum evolution increment: $\Delta e_{\text{min}}$
2: **if** Newton iterations fail to converge **then**
3:     Revert to previous converged state: $e = e - \Delta e$
4:     Reduce evolution increment: $\Delta e = \Delta e/10$
5:     **if** $\Delta e \geq \Delta e_{\text{min}}$ **then**
6:         **continue** at **while** with reduced increment: $e = e + \Delta e_{\text{new}}$
7:     **else return** Failed to solve after $e$
8:     **end if**
9: **else if** Convergence achieved after previous reduction **then**
10:     Restore original increment: $\Delta e = 1/n_{\text{steps}}$
11: **end if**
```

2.4.3 Arc-Length Method for Growth-Induced deformations

Although effective, the modified Newton's method can struggle when the system undergoes snap-through or snap-back instabilities, where multiple equilibrium configurations may exist for a given growth parameter value. In such cases, the standard Newton's method may fail to trace the complete equilibrium path. To address this limitation we have implemented an algorithm, based on the arc-length method [24, 25], specifically adapted for growth mechanics problems. This algorithm automatically adjusts the evolution increments based on the changing mechanical response of the system, taking smaller steps during drastic change and larger steps during gradual change, thereby optimizing the computational efficiency by tracing the equilibrium path in lesser steps. Additionally, this improves resolution of critical shape transitions without manual tuning of step size.

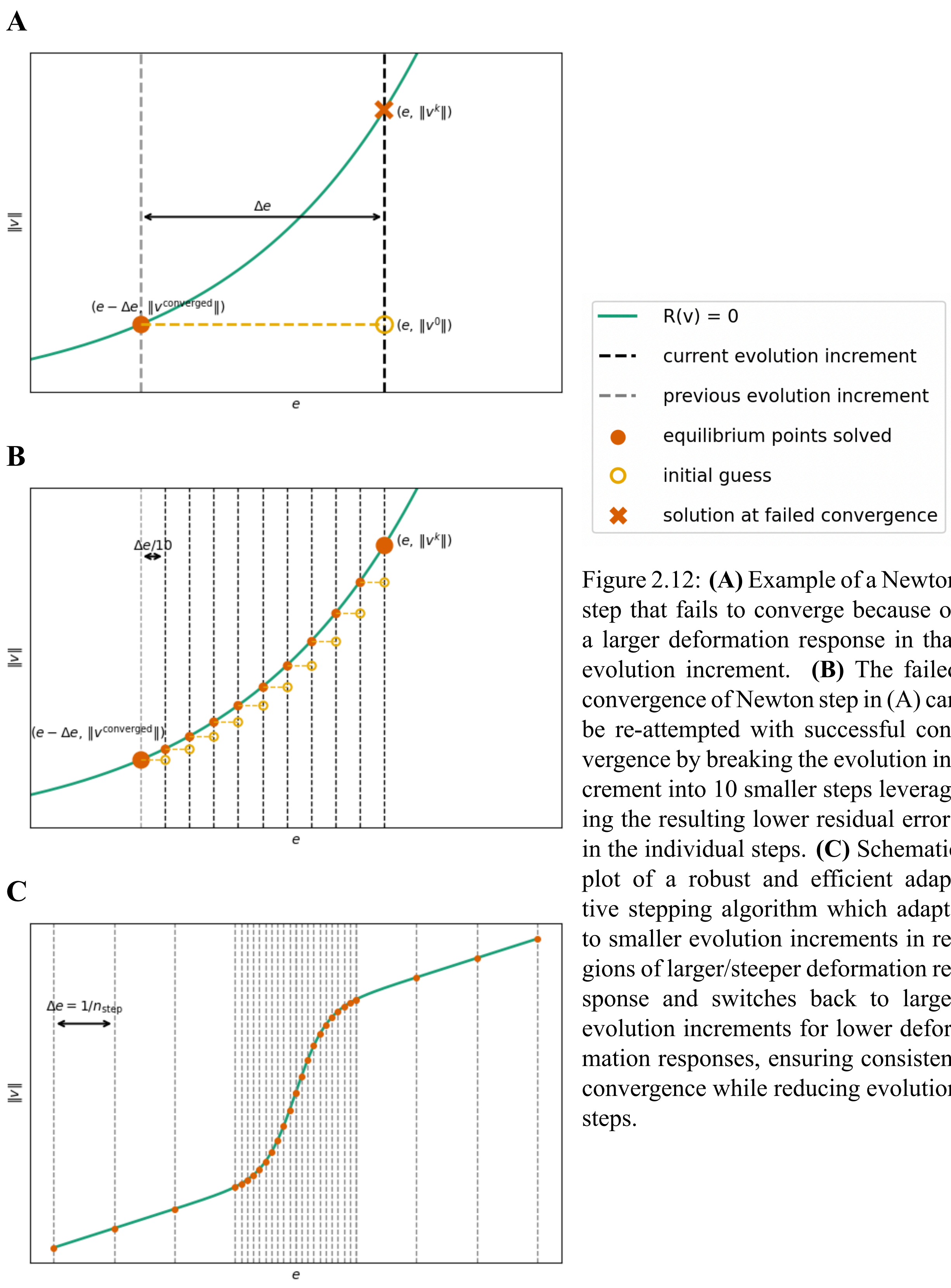


Figure 2.12: **(A)** Example of a Newton step that fails to converge because of a larger deformation response in that evolution increment. **(B)** The failed convergence of Newton step in (A) can be re-attempted with successful convergence by breaking the evolution increment into 10 smaller steps leveraging the resulting lower residual errors in the individual steps. **(C)** Schematic plot of a robust and efficient adaptive stepping algorithm which adapts to smaller evolution increments in regions of larger/steeper deformation response and switches back to larger evolution increments for lower deformation responses, ensuring consistent convergence while reducing evolution steps.

Extended System of Equations

The arc-length method treats the growth parameter e as an additional unknown rather than a prescribed value, thereby increasing the dimensionality of the solution space by one degree of freedom. To accommodate this additional unknown, we introduce a constraint equation that geometrically represents an ellipsoidal surface in the combined solution space of displacement coefficients and growth parameter:

$$f_{\text{arc}}(\boldsymbol{v}, e) = \frac{\|\boldsymbol{v} - \boldsymbol{v}^{\text{converged}}\|^2}{a_{\text{arc}}^2} + \frac{(e - e_{\text{converged}})^2}{b_{\text{arc}}^2} - 1 = 0 \tag{2.64}$$

where $\boldsymbol{v}^{\text{converged}}$ and $e_{\text{converged}}$ represent the most recently converged solution, while a_{arc} and b_{arc} are parameters controlling the relative scaling between coefficient changes and growth parameter changes.

The expanded system of equations becomes:

$$\boldsymbol{R}_{\text{arc}}(\boldsymbol{v}, e) = \begin{bmatrix} \boldsymbol{R}(\boldsymbol{v}, e) \\ f_{\text{arc}}(\boldsymbol{v}, e) \end{bmatrix} = \mathbf{0} \tag{2.65}$$

This formulation allows the solution path to progress naturally through regions where the relationship between growth and displacement is not monotonic, effectively permitting the growth parameter to decrease locally if necessary to follow the true equilibrium path.

Iterative Solution Procedure

For the k-th iteration, we seek the solution $(\boldsymbol{v}^k, e^k)$ that satisfies:

$$\boldsymbol{R}_{\text{arc}}(\boldsymbol{v}^k, e^k) = \mathbf{0} \tag{2.66}$$

Given the current approximation $(\boldsymbol{v}^{k-1}, e^{k-1})$, we define the increments:

$$\Delta\boldsymbol{v}^k = \boldsymbol{v}^k - \boldsymbol{v}^{k-1}, \quad \Delta e^k = e^k - e^{k-1} \tag{2.67}$$

Linearizing the expanded system $\boldsymbol{R}_{\text{arc}}(\boldsymbol{v}^k, e^k)$ around the current approximation $(\boldsymbol{v}^{k-1}, e^{k-1})$ using Equations (2.66), (2.67):

$$\boldsymbol{R}_{\text{arc}}(\boldsymbol{v}^{k-1}, e^{k-1}) + \nabla \boldsymbol{R}_{\text{arc}}(\boldsymbol{v}^{k-1}, e^{k-1}) \cdot \begin{bmatrix} \Delta \boldsymbol{v}^k \\ \Delta e^k \end{bmatrix} \approx \mathbf{0} \tag{2.68}$$

where the gradient $\nabla \boldsymbol{R}_{\text{arc}}$ takes the form:

$$\nabla \boldsymbol{R}_{\text{arc}} = \begin{bmatrix} \mathbf{K}(\boldsymbol{v}, e) & \frac{\partial \boldsymbol{R}}{\partial e}(\boldsymbol{v}, e) \\ \frac{\partial f_{\text{arc}}}{\partial \boldsymbol{v}}(\boldsymbol{v}, e) & \frac{\partial f_{\text{arc}}}{\partial e}(\boldsymbol{v}, e) \end{bmatrix} \tag{2.69}$$

The increments are then computed as:

$$\begin{bmatrix} \Delta \boldsymbol{v}^k \\ \Delta e^k \end{bmatrix} = -[\nabla \boldsymbol{R}_{\text{arc}}(\boldsymbol{v}^{k-1}, e^{k-1})]^{-1} \boldsymbol{R}_{\text{arc}}(\boldsymbol{v}^{k-1}, e^{k-1}) \tag{2.70}$$

The updated solution approximation becomes:

$$\boldsymbol{v}^k = \boldsymbol{v}^{k-1} + \Delta \boldsymbol{v}^k, \quad e^k = e^{k-1} + \Delta e^k \tag{2.71}$$

The iterations continue until the convergence criterion is satisfied (Figure 2.13):

$$||\boldsymbol{R}_{\text{arc}}(\boldsymbol{v}^k, e^k)|| < \epsilon \tag{2.72}$$

at which point the solution is recorded as the new converged state:

$$\boldsymbol{v}^{\text{converged}} = \boldsymbol{v}^k, \quad e^{\text{converged}} = e^k \tag{2.73}$$

where $\mathbf{K}(\boldsymbol{v}, e) = \partial \boldsymbol{R} / \partial \boldsymbol{v} \in \mathbb{R}^{N \times N}$ is the tangent stiffness matrix, $\Delta \boldsymbol{v}^k \in \mathbb{R}^N$, $\boldsymbol{R} \in \mathbb{R}^N$, $\boldsymbol{R}_{\text{arc}} \in \mathbb{R}^{N+1}$, and $\nabla \boldsymbol{R}_{\text{arc}} \in \mathbb{R}^{(N+1)\times(N+1)}$.

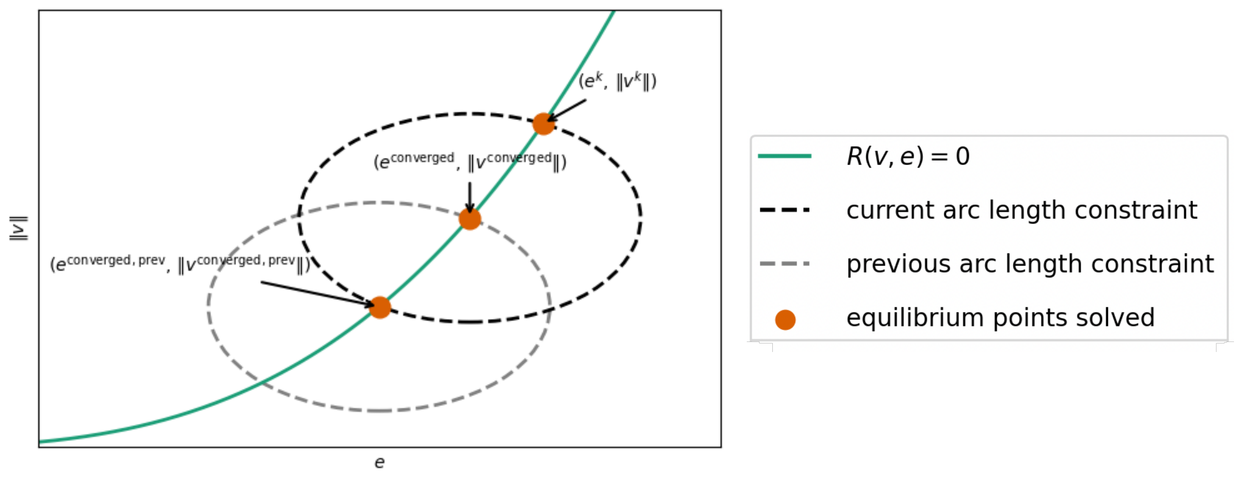

$(e^k, \|v^k\|)$
$(e^{\text{converged}}, \|v^{\text{converged}}\|)$
$(e^{\text{converged, prev}}, \|v^{\text{converged, prev}}\|)$
$\|v\|$
e
$R(v, e) = 0$
current arc length constraint
previous arc length constraint
equilibrium points solved

Figure 2.13: Example of an arc-length step showing the contours of the constraint and the actual solution trajectory, visualized by collapsing the unknown coefficient vector to a single axis of $\|\boldsymbol{v}\|$

Growth Evolution with Arc-Length step

The complete path-following algorithm for growth-induced deformation is structured as given in Algorithm 4, which automatically adapts the evolution increments based on the evolving mechanical response of the system, taking smaller steps near instabilities and larger steps in regions of gradual change, thereby optimizing the computational efficiency. Also, we set the initial guess in the arc length step as the previous converged solution $\boldsymbol{v}^0 = \boldsymbol{v}^{\text{converged}}$, $e^k = e^{\text{converged}}$.

Advanced Arc-Length Enhancements

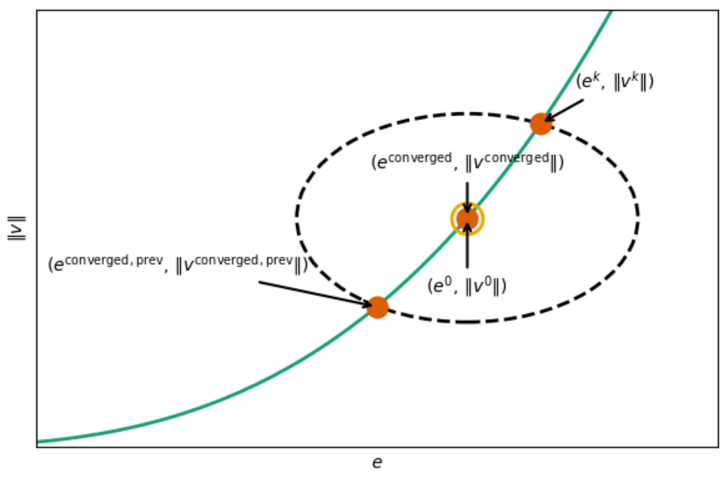
We have implemented several sophisticated enhancements to the basic arc-length method to further improve its performance for growth mechanics problems:

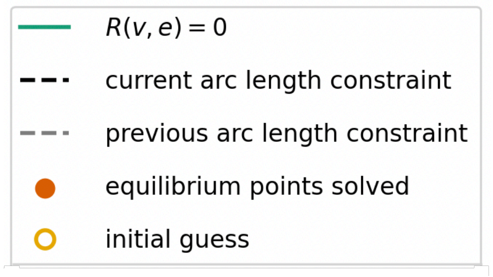
Multiple Initial Guess Strategy To accelerate convergence of the arc-length method, we evaluate multiple candidate initial guesses (Figure 2.14) and select the one with the smallest initial residual to increase the likelihood and speed of convergence:

1. **Direct approach**: $\boldsymbol{v}^{\text{init},1} = \boldsymbol{v}^{\text{converged}}$, $e^{\text{init},1} = e^{\text{converged}}$

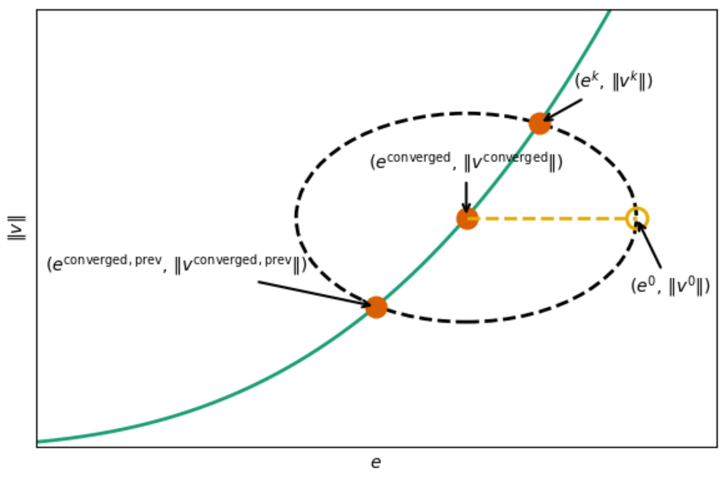
   This uses the last converged solution as the initial guess.

A


$(e^k, \|v^k\|)$
$(e^{\text{converged}}, \|v^{\text{converged}}\|)$
$(e^{\text{converged, prev}}, \|v^{\text{converged, prev}}\|)$
$(e^0, \|v^0\|)$
$\|v\|$
e


$R(v, e) = 0$
current arc length constraint
previous arc length constraint
equilibrium points solved
initial guess

B


$(e^k, \|v^k\|)$
$(e^{\text{converged}}, \|v^{\text{converged}}\|)$
$(e^{\text{converged, prev}}, \|v^{\text{converged, prev}}\|)$
$(e^0, \|v^0\|)$
$\|v\|$
e

C

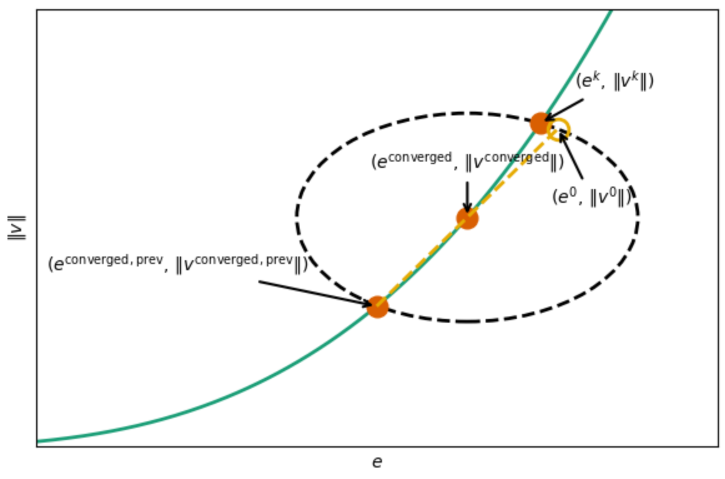

$(e^k, \|v^k\|)$
$(e^{\text{converged}}, \|v^{\text{converged}}\|)$
$(e^0, \|v^0\|)$
$(e^{\text{converged, prev}}, \|v^{\text{converged, prev}}\|)$
$\|v\|$
e

Figure 2.14: **Multiple initial guess strategy**: **(A)** A sample arc-length convergence step and the initial guess of the unknown coefficients and evolution parameter is set as the previously converged values. **(B)** The same arc-length convergence step with the initial guess set using the evolution biased approach where the coefficient vector is initialized as the previously converged value but the evolution parameter is incremented to lie on the surface of the arc-length constraint. **(C)** The same arc-length convergence step with the initial guess being extrapolated from 2 previously converged solutions.

Algorithm 4 Arc-Length Growth Evolution Algorithm

```
1: Initialize: $\boldsymbol{v}^{\text{converged}} = \mathbf{0}$, $e^{\text{converged}} = 0$
2: Initialize arc-length parameters: $a_{\text{arc}}$, $b_{\text{arc}}$
3: Initialize allowable iterations: $k^{\max}$
4: while $e^{\text{converged}} < 1$ do
5:     Initialize: $k = 1$, $\boldsymbol{v}^0 = \boldsymbol{v}^{\text{converged}}$, $e^0 = e^{\text{converged}}$
6:     while $\|\boldsymbol{R}_{\text{arc}}(\boldsymbol{v}^k, e^k)\| > \epsilon$ and $k \leq k^{\max}$ do
7:         Compute growth tensor based on $g = e^k \cdot g^{\text{final}}$
8:         Evaluate displacement field, residual, and tangent matrices: $\boldsymbol{u}^k$, $\boldsymbol{R}_{\text{arc}}$, $\nabla\boldsymbol{R}_{\text{arc}}$
9:         Solve for increments: $\Delta\boldsymbol{v}^k$, $\Delta e^k$
10:        Update solution: $\boldsymbol{v}^k = \boldsymbol{v}^{k-1} + \Delta\boldsymbol{v}^k$, $e^k = e^{k-1} + \Delta e^k$
11:        Increment iteration counter: $k = k + 1$
12:    end while
13:    if $k > k^{\max}$ then
14:        return Arc length step failed after $e^{\text{converged}}$
15:    end if
16:    Update converged solution: $\boldsymbol{v}^{\text{converged}} = \boldsymbol{v}^k$, $e^{\text{converged}} = e^k$
17:    Store the point on solution trajectory: $\boldsymbol{v}(\boldsymbol{X}, e^{\text{converged}}) = \boldsymbol{v}^{\text{converged}}$
18:    Update $\boldsymbol{v}^{\text{converged}}$, $e^{\text{converged}}$ values in arc length equation $f_{\text{arc}}$
19: end while
```

2. **Evolution-biased approach**: $\boldsymbol{v}^{\text{init},2} = \boldsymbol{v}^{\text{converged}}$, $e^{\text{init},2} = e^{\text{converged}} + b_{\text{arc}}$

   This guess advances the evolution parameter while keeping displacements fixed, effectively placing the initial guess on the ellipsoid boundary along the evolution axis. It performs well in regions where the structure's deformation does not change drastically with evolution parameter.

3. **Extrapolation**: $\boldsymbol{v}^{\text{init},3} = \boldsymbol{v}^{\text{converged}} + (\boldsymbol{v}^{\text{converged}} - \boldsymbol{v}^{\text{converged,prev}})$,
   $e^{\text{init},3} = e^{\text{converged}} + (e^{\text{converged}} - e^{\text{converged,prev}})$

   This guess extrapolates from the two most recent converged solutions, assuming the solution will continue to evolve with the same trend. It is particularly effective in regions of steady, monotonic deformation.

We automate determining the optimal initial guess at each arc-length step by evaluating the residual magnitude for each candidate:

$$(\boldsymbol{v}^{\text{init}}, e^{\text{init}}) = \arg\min_{(v,e)} \left\{ \|\boldsymbol{R}_{\text{arc}}(\boldsymbol{v}, e)\|; (\boldsymbol{v}, e) \in \{(\boldsymbol{v}^{\text{init},1}, e^{\text{init},1}), (\boldsymbol{v}^{\text{init},2}, e^{\text{init},2}), (\boldsymbol{v}^{\text{init},3}, e^{\text{init},3})\} \right\} \tag{2.74}$$

This adaptive selection strategy reduces the number of iterations required for convergence by selecting a guess that is likely closest to the actual solution. With this strategy the arc-length step converges robustly and effectively for simulations where the solution path is unpredictable. The effective speed-up arises from the fact that residual evaluation step is much faster than a convergence iteration for large scale systems. If there is prior knowledge of the trend of the solution path, either analytically or from a prior similar simulation, we can also consistently choose one initial guesses that works the best for most steps and skip evaluating the residuals for comparison.

Dynamic Arc-Length Parameter Adaptation To optimize the step size based on the local characteristics of the solution path, we dynamically adjust the arc-length parameters after each successful convergence:

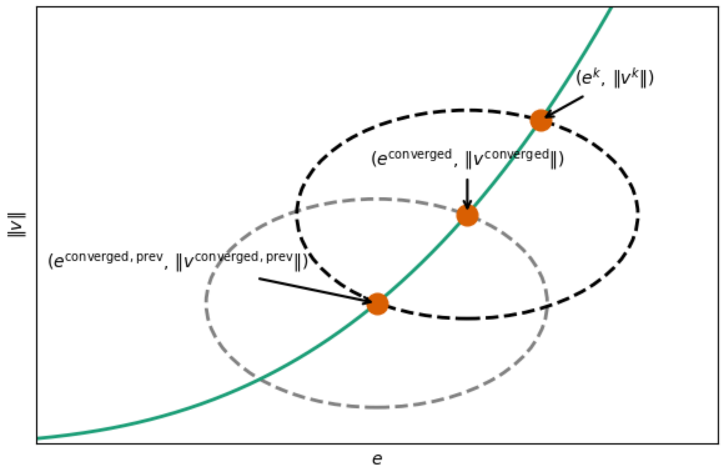
$$a_{\text{arc,new}} = \sqrt{2}\|\boldsymbol{v}^{\text{converged}} - \boldsymbol{v}^{\text{converged,prev}}\|, \quad b_{\text{arc,new}} = \sqrt{2}|e^{\text{converged}} - e^{\text{converged,prev}}| \tag{2.75}$$

This adaptation adjusts the relative contribution of displacement and evolution parameter in the ellipsoid based on the observed solution behavior (Figure 2.15 B). The scaling factor $\sqrt{2}$ ensures stability in the adaptation process by capping the total constraint surface size, $a^2_{\text{arc,new}} + b^2_{\text{arc,new}} < 2(a^2_{\text{arc}} + b^2_{\text{arc}})$, while tracing the solution trajectory.

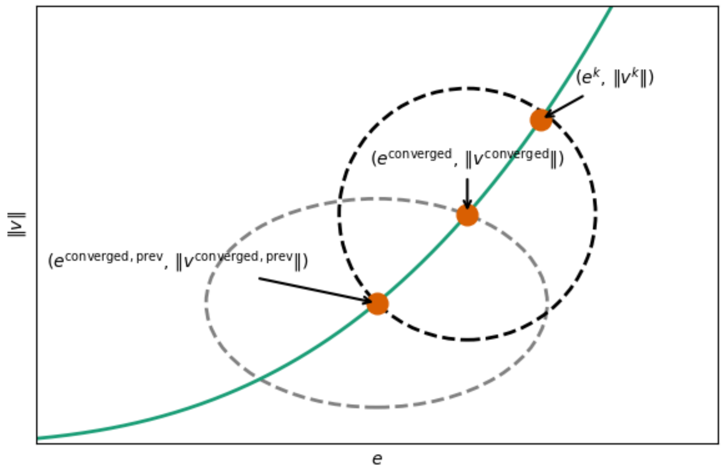
Fallback Strategy for Convergence Failures When the arc-length method fails to converge for a given constraint surface, we implement an automatic fallback strategy:

$$a_{\text{arc,new}} = \frac{a_{\text{arc}}}{\sqrt{10}}, \quad b_{\text{arc,new}} = \frac{b_{\text{arc}}}{\sqrt{10}} \tag{2.76}$$

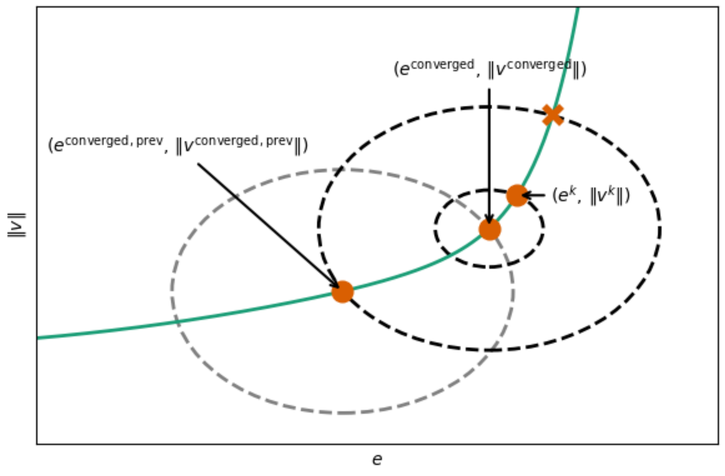
A


$(e^k, \|v^k\|)$
$(e^{\text{converged}}, \|v^{\text{converged}}\|)$
$(e^{\text{converged, prev}}, \|v^{\text{converged, prev}}\|)$
$\|v\|$
e

B


$(e^k, \|v^k\|)$
$(e^{\text{converged}}, \|v^{\text{converged}}\|)$
$(e^{\text{converged, prev}}, \|v^{\text{converged, prev}}\|)$
$\|v\|$
e

C


$(e^{\text{converged}}, \|v^{\text{converged}}\|)$
$(e^{\text{converged, prev}}, \|v^{\text{converged, prev}}\|)$
$(e^k, \|v^k\|)$
$\|v\|$
e

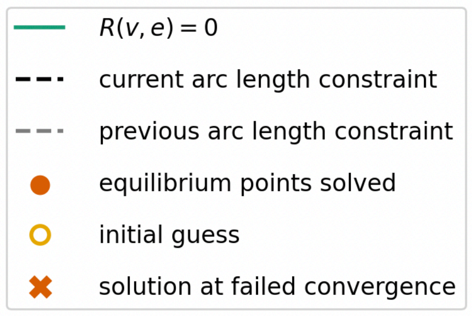

$R(v, e) = 0$
current arc length constraint
previous arc length constraint
equilibrium points solved
initial guess
solution at failed convergence

Figure 2.15: **(A)** A sample arc-length step where the constraint surface from the current previous convergence step are shown. **(B)** Arc-length steps charting the same solution trajectory as in (A) with dynamic arc-length parameter adaptation. Notice the change of the current arc-length constraint surface based on the trend of the converged solutions. **(C)** Arc-length step has previously converged but fails in the current step because the deformation response suddenly increases steeply. The fallback strategy reduces the arc-length parameters so the new arc-length constraint ensures lower deformation increments that allows for convergence to be possible.

This modification reduces the arc-length ellipsoid size by a factor of 10 in the convergence iterations following failure (Figure 2.15 C). This adaptive shrinking continues until either convergence is achieved or a minimum ellipsoid size is reached.

When incorporated, these enhancements result in a highly robust path-following algorithm capable of navigating complex instability patterns in growth-induced deformation problems, including snap-through buckling, wrinkling, and bifurcation phenomena, that commonly arise in morphing biological and engineered structures. The proposed enhancements are implemented all together or optionally based on the requirement of the particular simulation for continuous convergence and the gains in compute time each enhancement contributes.

2.4.4 Computational complexity

Understanding the computational complexity of the nonlinear solver algorithms is important to select the optimal solvers based on system size and available computational resources, if both solvers have satisfactory simulation results. The computational complexity for both algorithms is dominated by solving the linear system involving the gradient matrix $\nabla \boldsymbol{R}$ or $\nabla \boldsymbol{R}_{\text{arc}}$. For a system with N degrees of freedom, the complexity is as follows:

- **Matrix assembly**: $\mathcal{O}(N)$ for forming the augmented system matrices $\nabla \boldsymbol{R} \in \mathbb{R}^{N \times N}$ or $\nabla \boldsymbol{R}_{\text{arc}} \in \mathbb{R}^{(N+1)\times(N+1)}$.
- **Direct matrix solver**: For a general dense matrix, the complexity would be $\mathcal{O}((N+1)^3)$ using Gaussian elimination. However, the finite element formulation produces a sparse matrix with bandwidth $B \ll N$. Using sparse direct solvers such as multifrontal methods, the complexity reduces to approximately $\mathcal{O}(NB^2)$ for factorization and $\mathcal{O}(NB)$ for forward/backward substitution.
- **Iterative matrix solver**: For very large systems ($N > 10^5$), iterative methods such as preconditioned conjugate gradient or GMRES are often employed, with complexity

approximately $\mathcal{O}(N)$ [26].

In practice, the sparsity pattern of the tangent stiffness matrix is determined by the finite element mesh connectivity, with each degree of freedom directly coupled only to degrees of freedom associated with adjacent elements. For structured meshes in 3D, this results in a bandwidth $B \approx \mathcal{O}(N^{2/3})$, leading to an overall complexity of approximately $\mathcal{O}(N^{7/3})$ for direct methods. The additional row and column introduced by the arc-length constraint f_{arc} do not significantly impact the overall complexity, as they represent a rank-one update to the original system and can be handled efficiently using the Sherman-Morrison formula. This makes the arc-length algorithm a computationally efficient upgrade for robustness over the algorithm based on modified Newton solver.

In this section, we have presented two advanced numerical solution strategies specifically developed for growth mechanics problems: the modified Newton's method with adaptive evolution stepping and the enhanced arc-length method. These algorithms represent significant methodological advances for simulating the complex mechanical behavior of morphing structures driven by heterogeneous growth fields.
The modified Newton's method provides an efficient approach for problems with monotonic deformation paths by employing predictive initial guesses and adaptive step size control. The arc-length method extends these capabilities to handle instabilities and non-monotonic deformation paths through evolution parameter relaxation. Both methods incorporate sophisticated adaptations to enhance convergence robustness and computational efficiency.

2.5 Summary

The computational framework described in this chapter forms the foundation for our investigations into the morphogenetic mechanisms of biological growth and the design principles of shape-programmable materials in the following chapters.

Chapter 3

Stress-ball morphogenesis

This chapter is based on the published research done in collaboration with the Nelson Group [1], the experiments and imaging were led by Michael A. Palmer and the meshes, physical models, and simulations were done by myself.

3.1 Background

Function dictates form, and as a consequence, the physiology of many organ systems across the evolutionary tree shows conservation in underlying anatomy [27]. The lungs, however, are radically different across classes of vertebrates [28]. Both mammals and birds have lungs that are structured anatomically into distinct tissue compartments that physically segregate the conduction of air (through bronchi and air sacs, respectively) from gas exchange (through alveoli and parabronchi), despite the fact that air flows tidally in the mammal but unidirectionally in the bird [29, 30]. In contrast, the lungs of many reptiles are simple sacs in which both conduction and gas exchange take place in the same anatomic chamber [31]. Unidirectional airflow was thought to require the complex anatomy of the parabronchial lung until the recent discovery of unidirectional flow patterns in the simple sac-like lungs of the green iguana lizard [32]. This pattern of airflow is hypothesized to depend in part on the unique, honeycomb-shaped structure of the wall of the lizard lung [33], but the biological and physical basis of its morphogenesis has not been reported.

3.2 Introduction

To investigate the physical mechanisms underlying lizard lung morphogenesis, the brown anole (Anolis sagrei) was established as a model system. We selected this species because of its small size, short incubation time (~ 30 days) and high breeding potential [34], and because a reference genome for the Anolis genus has been published [35].

The lizard lung is faveolar, characterized by shallow primary and secondary corrugations arranged in a hexagonal, honeycomb-like pattern that increases surface area for gas exchange (Fig. 3.1, A and B). In the anole embryo, the lung buds into a simple wishbone-shaped structure that separates from the ventral foregut, similar to mammalian and avian lungs (Fig. 3.1 A). Between embryonic days (E) 5 and 6, the lumen of the wishbone inflates (Fig. 3.1 A). At E7, the initially smooth epithelium forms into the rounded corrugations (Fig. 3.1, A and B) that eventually give rise to the faveoli, which are alveolus-like structures lined by a thin, squamous epithelium used for gas exchange [36].

Smooth muscle shapes the developing lung epithelium of the mouse [5, 37] and chicken [38] and also drives epithelial morphogenesis in other organs including the mouse gut [39, 40] and prostate [41]. Mature trabeculae in adult lizard lungs contain a smooth muscle core [36], contraction of which maintains the lung structure and promotes airflow [42]. To understand this process in the anole, the spatial distribution of α-smooth muscle actin (αSMA)–expressing cells was mapped across developmental time and samples were labeled with antibodies specific to epithelial cytokeratins (CKs)[1]. During the inflation stage at E6, αSMA+ cells are spread uniformly over the basal surface of the epithelium (Fig. 3.1 B). As epithelial corrugations appear at E7, the αSMA+ cells form a hexagonal meshwork of thick bundles (Fig. 3.1 B) that creates a contractile cage around the entire epithelium. Epithelial corrugations only emerge from the gaps within the hexagonal mesh (Fig. 3.1, C), which is localized immediately adjacent to the regions of the highest epithelial curvature, suggesting that the differentiation of this contractile tissue might be linked to morphogenesis of the epithelium.

There are two possible mechanisms by which epithelial corrugations could be sculpted by a

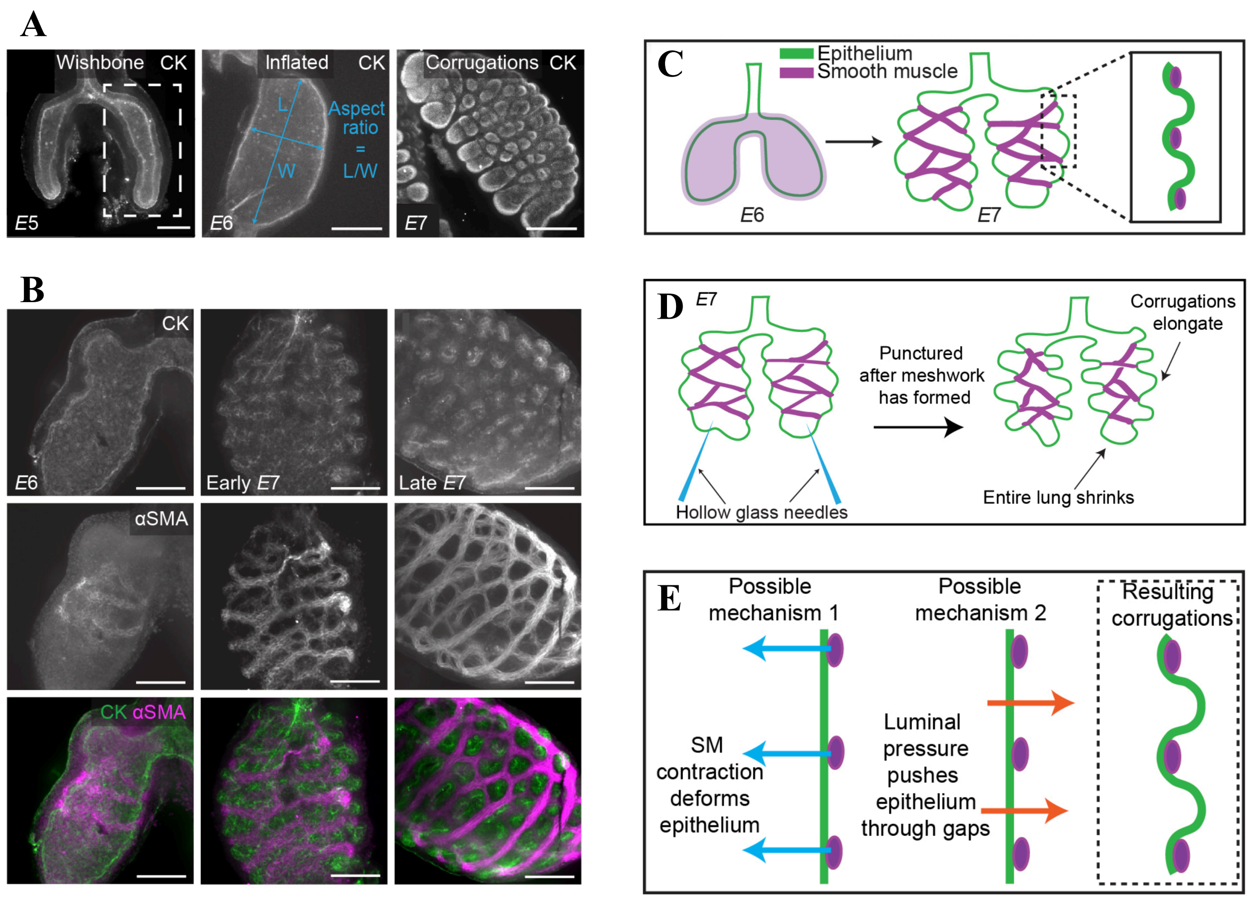


Figure 3.1: **(A)** Immunofluorescence analysis for CK showing the transition from simple wishbone shape (left) to inflated lung (middle) to corrugations (right). L, length; W, width; Scale bars, 100 μm. **(B)** Immunofluorescence analysis for CK and αSMA showing the formation of bundles of smooth muscle cells that organize into a meshwork around the lung epithelium and emergence of epithelial corrugations through gaps in this meshwork; Scale bars, 100 μm. **(C)** Schematic illustrating the processes of smooth muscle differentiation and epithelial morphogenesis. **(D)** Schematic illustrating predicted change in epithelial morphology after puncturing an E7-stage lung. **(E)** Possible mechanisms to form epithelial corrugations.

smooth muscle mesh (Fig. 3.1 E). The smooth muscle meshwork could actively contract to deform the underlying epithelium. Alternatively, luminal fluid pressure could push the epithelial sheet through the holes within a passive but stiff meshwork. Because *in vivo* experiments offer limited ability to isolate and test these mechanical hypotheses independently, a computational model is necessary to systematically evaluate how epithelial corrugations form.

To determine the mechanism that promotes corrugations, we generated a computational model of the developing anole lung based on measured geometrical and biophysical parameters. The model was initialized with a smooth, spheroidal epithelial surface whose thickness and diameter approximated those of the inflation-stage anole lung (E6). We imposed a pressure on the fluid within the lumen and surrounded the basal surface of the epithelium with a stiff hexagonal lattice to mimic the geometry and mechanical properties of the smooth muscle meshwork (Fig. 3.2 K). Our simulations revealed that increasing the pressure within the lumen causes the epithelial sheet to expand through the holes within the stiff hexagonal lattice, thus generating corrugations on the surface and aligning with observations of development in E7-stage of the anole lung (Fig. 3.3 C). Our simulations also predicted that once the corrugations have formed, a sudden drop in luminal pressure would decrease the resistance to contraction of the elastic meshwork and result in elongation of the sheet through the holes in the mesh (Fig. 3.3, D and E). To test this prediction experimentally, we used needles to puncture corrugation-stage anole lungs (Fig. 3.1 D). As predicted by our model, decreasing luminal pressure at this stage led to elongated epithelial corrugations, separated by a contracted smooth muscle meshwork (Fig. 3.3 F and Fig. 3.5 E); simultaneous treatment with nifedipine prevented elongation of the corrugations in the punctured lungs (Fig. 3.5, F and G). Our simulations thus reveal that fluid pressure induces the formation of the corrugated epithelium of the lizard lung, driving the morphogenesis of the faveoli.

In the following sections, we walk through how we constructed the computational model of the lizard lung and simulated the conditions of inflation, to capture E7 development, and puncturing, to capture smooth muscle contraction in the punctured anole lungs. We further analyze the simulations repeated with altered physical parameters to reveal the mechanisms and conditions required for development of epithelial corrugations, which later become faveolae, leading us to understand a crucial step of lung morphogenesis. Further, we see that the physical principles used to sculpt the unusual architecture of the lizard lung can be exploited as a novel strategy to engineer tissues.

3.3 Domain setup

The lung of the adult anole is a single-chambered (unicameral) hollow lumen surrounded by a simple cuboidal epithelium. As epithelial corrugations appear at E7, the smooth muscle cells form a hexagonal meshwork of thick bundles (Fig. 3.1, B and C) that creates a contractile cage around the entire epithelium. To capture this structure in the computation model, the lizard lung is composed of two physical materials, viz., the epithelium, modeled as a spherical shell, and the smooth muscle, modeled as a thick hexagonal lattice on the outer surface of the epithelium (Fig. 3.2, K and L). The epithelium and smooth muscle are assumed to be elastic solids characterized by different physical properties, and these tissues are denoted by subscripts "ep" and "sm", respectively.

3.3.1 Model geometry

To approximate the structure of the anole lung in simulations, we first construct a hexagonal lattice to spatially represent the smooth muscle bundles. We then assign the appropriate thickness and width to these bundles, and finally adjoin a spherical epithelial shell to complete the composite model.

Hexagonal tiling on a spherical surface

The goal was to generate a hexagonal lattice on a spherical surface by partitioning it into hexagons. But, tiling the sphere with hexagons alone is topologically impossible, as demonstrated by Euler's formula [43]. Euler's formula for convex polyhedra—one of the cornerstones of topology and combinatorial geometry—dictates that for any convex polyhedron (or, more generally, any polyhedral decomposition of a sphere), the relationship holds,

$$V - E + F = 2$$

where V is the number of vertices, E is the number of edges, and F is the number of faces. This invariant, which remains constant regardless of deformations (provided the polyhedron remains topologically equivalent to a sphere), imposes strict limitations on the possible tilings of a spherical surface.

Consider a hypothetical tiling in which every face is a regular hexagon. In this case, the total number of faces F is equal to the number of hexagons F_6, i.e., $F = F_6$. Assume that three hexagons meet at each vertex. Since each hexagon has 6 edges and each edge is shared by two hexagons, the total number of edges is

$$E = \frac{6F_6}{2} = 3F_6.$$

Furthermore, because three hexagons meet at each vertex, the total number of vertex–face incidences is $6F$, yielding

$$3V = 6F_6 \quad \Longrightarrow \quad V = 2F_6$$

Substituting these expressions into Euler's formula gives

$$V - E + F = 2F_6 - 3F_6 + F_6 = 0,$$

which contradicts the required value of 2. This contradiction demonstrates that a tiling composed solely of hexagons is topologically impossible on a spherical surface.

To resolve this issue and satisfy Euler's formula, the tiling must incorporate faces with fewer than six edges. Denote by F_5 the number of pentagons and by F_6 the number of hexagons, so that

$F = F_5 + F_6$. Assuming that the tiling is trivalent (i.e., three faces meet at each vertex), we have

$$E = \frac{5F_5 + 6F_6}{2} \quad \text{and} \quad V = \frac{5F_5 + 6F_6}{3}.$$

Substituting these expressions into Euler's formula,

$$\frac{5F_5 + 6F_6}{3} - \frac{5F_5 + 6F_6}{2} + (F_5 + F_6) = 2 \implies \frac{F_5}{6} = 2, \implies F_5 = 12.$$

Thus, any spherical tiling that is predominantly hexagonal must necessarily contain exactly 12 pentagons. These pentagons account for the sphere's curvature and ensure that the tiling satisfies Euler's invariant.

Now we know that a hexagonal tiling on a sphere has 12 pentagonal defects, we move on to generating the tiling. Since triangulation is more straightforward than direct hexagonal tiling, a regular triangular mesh was generated first. Its dual structure yielded the desired hexagonal mesh (Fig. 3.2 D).

The icosahedron (Fig. 3.2 A), inscribed in the the spherical surface(say with radius R_{ep}) to be tiled, is employed as the starting template for such triangulations due to its exceptional symmetry and nearly uniform distribution of vertices over the sphere. With 12 vertices, the icosahedron naturally aligns with the necessity for 12 pentagons, and its 20 triangular faces provide a convenient framework for further subdivision.

In practice, each triangular face of the icosahedron is subdivided into smaller equilateral triangles (Fig. 3.2 B), and the newly created vertices are radially projected onto the sphere (Fig. 3.2 C). The dual of this triangulated mesh (Fig. 3.2 D) is then constructed by connecting the centroids of adjacent triangles. In the resulting dual tiling, most vertices of the triangulated mesh give rise to hexagonal cells; however, at the locations corresponding to the original icosahedron vertices—where the spherical curvature is most pronounced—the arrangement of centroids forms pentagonal cells. Consequently, the dual of the subdivided icosahedron yields a tiling composed primarily of hexagons with exactly 12 pentagons, in full accordance with Euler's formula (Fig.

3.2, E-G).

The density of this hexagonal lattice is controlled by a index number that is used to generate the vertex coordinates, examples of different polyhedra with different densities (Fig. 3.2, E-G). The lattice density that most closely resembled the structure of the lizard lung was selected (Fig. 3.2, F and H).

The hexagonal lattice which is created by using the coordinates of the corresponding polyhedra, with the vertices still lying on the spherical surface, is now projected onto a sphere by making the the edges connecting the vertices into radial arcs of the sphere.

Making the composite lung

Our hexagonal lattice exists as an interconnected network of one-dimensional curves on the spherical surface with radius R_{ep}. The lattice was then transitioned into three-dimensional structural elements representing smooth muscle bundles.

We first introduced a finite width w_{sm}, effectively transforming these curves into planar, two-dimensional components (Fig. 3.2 I). Subsequently, we expanded this structure radially outward by introducing a thickness h_{sm}, thus creating a three-dimensional representation corresponding to a hexagonal cage-like network of smooth muscle (Fig. 3.2 J).

The epithelial layer is represented as a hollow spherical shell, bounded externally by the spherical surface of radius R_{ep} and extending radially inward with thickness h_{ep}. The epithelium and smooth muscle components are constructed so the smooth muscle cage is outside and in contact with the epithelial layer (Fig. 3.2, K and L).

The geometrical parameters were chosen based on experimentally measured values and are expressed relative to the outer radius R_{ep} of the hollow epithelial sphere. The thickness of the epithelial sheet and smooth muscle lattice were $h_{ep}/R_{ep} = 0.05$ and $h_{sm}/R_{ep} = 0.075$, respectively. The circumferential width of the smooth muscle lattice is expressed relative to the spacing between muscle fibers as $w_{sm}/\sqrt{3}l_{sm} = 0.1$, where l_{sm} is the edge length of the hexagonal lattice and $l_{sm}/R_{ep} = 0.304$.

3.3.2 Meshing

The domain was discretized using first-order tetrahedral elements generated with the open-source software Gmsh [8]. The entire 3D lung structure was meshed using $\sim 230,000$ tetrahedral elements, which were divided between the smooth muscle ($\sim 85,000$) and epithelium ($\sim 145,000$), and the final mesh contained $\sim 57,000$ vertices/nodes and $\sim 170,000$ degrees of freedom. Accuracy of the obtained results was tested against finer mesh refinements.

3.3.3 Material properties

The two material properties, Young's modulus and Poisson's ratio, for the epithelium and smooth muscle are set based on previous studies [44]. Poisson's ratios of both tissues were set to be $\nu_{\text{sm}} = \nu_{\text{ep}} = 0.3$. For the simulation, the "pressure" scale is assumed to be the Young's modulus of the epithelium. The Young's modulus of the smooth muscle was non-dimensionalized relative to that of the epithelium, yielding $E_{\text{sm}}/E_{\text{ep}} = 75$. Similarly, luminal pressure was measured relative to E_{ep}.
This reconstruction of the lizard lung effectively captures the composition of the tissues for simulations to quantitatively analyze the major drivers of morphogenesis in lizard lung.

3.4 Simulating deformation

In this section, we present our approach to simulating the deformation of the lizard lung. First, we identify the key physical factors driving morphological change during development. We then construct a computational framework capable of numerically solving for the deformation field for any given set of physical parameters. Using this framework, we simulate the morphogenesis by systematically varying relevant physical parameters and capturing the resulting tissue deformation at each stage.

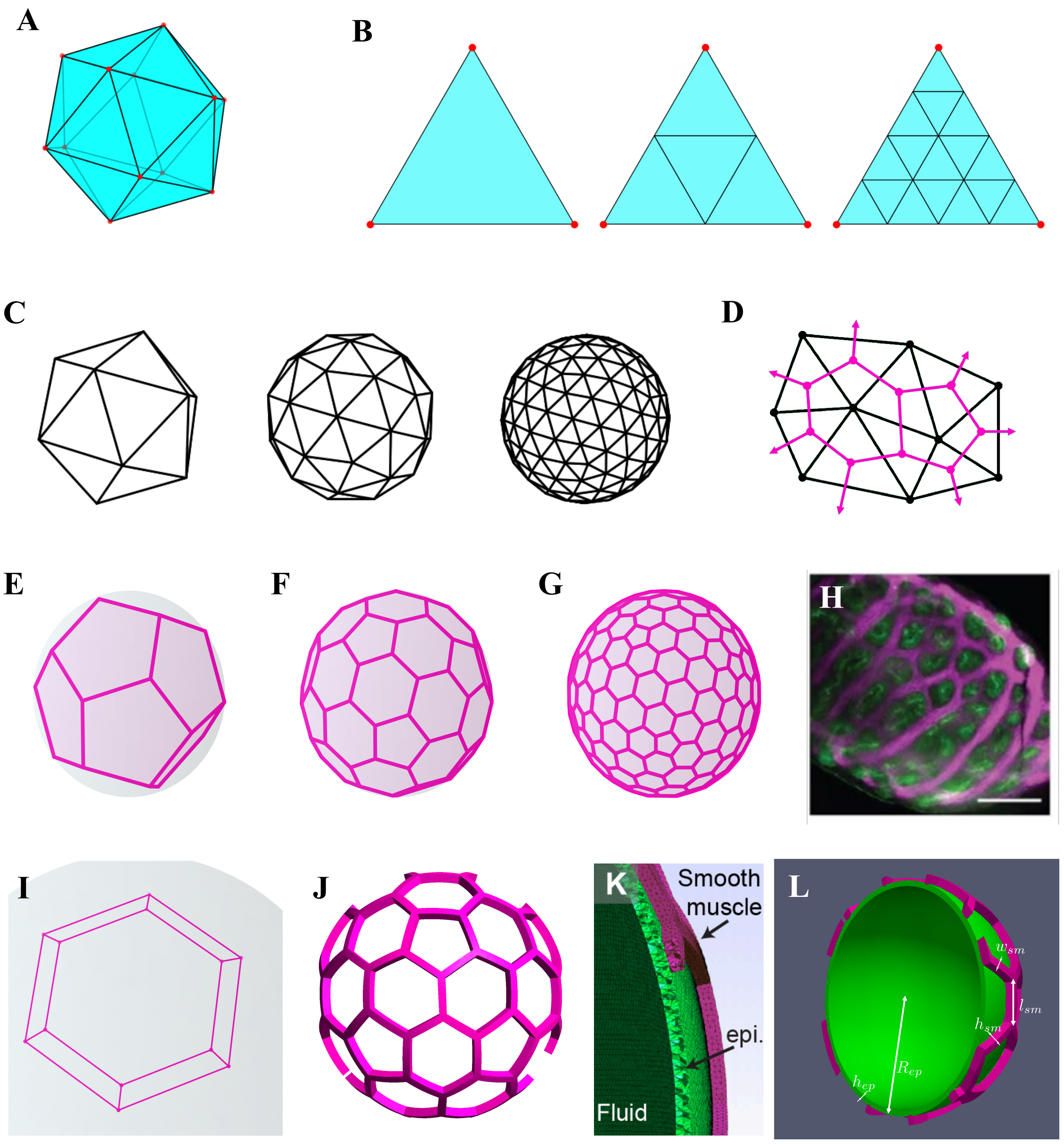


Figure 3.2: **(A)** Icosohedron inscribed in the sphere. **(B)** Barycentric subdivisions (equilateral edge divided by 1, 2, and 4) of each icosohedron face for triangulation. **(C)** Corresponding triangulations from (B). **(D)** Creating a dual mesh of hexagons/pentagons from triangulations. **(E)** Polyhedra (hexagon and pentagon) lattice created as the dual of triangulations from icosohedron. **(F)** Polyhedra lattice created as the dual of triangulations from icosohedron faces subdivided to 2, chosen as the framework for smooth muscle bundle. **(G)** Polyhedra lattice created as the dual of triangulations from icosohedron faces subdivided to 4. **(H)** Immunofluorescence analysis showing formation of smooth muscle bundles(magenta) over epithelium(green), scale bar 100 μm. **(I)** Hollow hexagon patch with width $w_{sm}/2$ created from 1D lattice. **(J)** 3D hexagonal cage. **(K)** Tissue layers in the mesh of the computational model. **(L)** Geometrical parameters of the computational model.

3.4.1 Driving factors

Two primary mechanisms may underlie the formation of epithelial corrugations in the gaps of the smooth muscle mesh (Fig. 3.1 E). The smooth muscle meshwork could actively contract to deform the underlying epithelium. Alternatively, luminal fluid pressure could push the epithelial sheet through the holes within a passive but stiff meshwork.

Contraction of smooth muscle

Deformation due to growth and contraction is uniquely determined by the growth tensor $\mathbf{F}_g$. For simplicity, we neglected epithelial proliferation to focus on the effect of mechanical forces, i.e., $\mathbf{F}_{\text{g,ep}} = \mathbf{I}$. We assumed that smooth muscle contraction was uniform and either isotropic or anisotropic (contraction only along the fiber length). For isotropic contraction, the growth deformation tensor was taken as

$$\mathbf{F}_{\text{g,sm}} = (1 + g_{\text{sm}})\mathbf{I}$$

where g_{sm} is the strain associated with contraction. For anisotropic contraction, we assume the stretch in the smooth muscle is $(1 + g_{sm})$ along the fiber direction $\boldsymbol{f}^0$ (Fig. 3.3 A) with no contraction in the orthogonal directions, and the growth deformation tensor is given by [45]

$$\mathbf{F}_{\text{g,sm}} = (1 + g_{\text{sm}})\boldsymbol{f}^0 \otimes \boldsymbol{f}^0$$

Smooth muscle contraction was modeled as negative growth, $-1 < g_{\text{sm}} \leq 0$.

Luminal pressure

The presence of a lumen inside the lung was modeled by a positive normal pressure acting on the inner surface of the spherical epithelial shell. This pressure acts in the opposite direction of the outer normal **n** on the current deformed boundary of the mesh (Fig. 3.3 B). That is, on the inner surface the traction force **t** can be written as , $\mathbf{t} = -p\,\mathbf{n}$, where p is the luminal pressure. The outer surface of the lung was considered to be traction free and the traction from the luminal pressure is the only external force acting on the lung.

3.4.2 Solving for deformation

To determine how mechanical forces shape the developing lung, we computed tissue deformation as the elastic response of the composite lung structure to smooth muscle contraction and luminal pressure.

The deformation of growing tissues was formulated by adopting the method of deformation gradient decomposition [11] (reference ch.2). We assumed that growth and contraction of smooth muscle are much slower than mechanical relaxation, and hence, the system is always in quasi-mechanical equilibrium allowing us to solve for deformation by minimizing the total elastic energy(reference ch.2).

To numerically solve this problem using finite element method [12], drawing from the methods in Section 2.3, we incorporate in the total potential energy, the elastic energy density of the composite lung and the virtual work from traction. We further want to constrain rigid body translation and rotation of the lizard lung model during deformation. The total potential energy functional was constructed to solve for the displacement field by applying minimization as follows:

$$\Pi(\boldsymbol{u}, \lambda_{\text{tr}}, \lambda_{\text{rot}}) = \int_{\Omega} \mathbf{J_g}\, \psi(\mathbf{F}_e, \boldsymbol{X})\, dV - \int_{\partial\Omega^{\text{in}}} t \cdot \boldsymbol{u}\, ds - \int_{\Omega} \lambda_{\text{tr}} \cdot \boldsymbol{u}\, dV - \int_{\Omega} \lambda_{\text{rot}} \cdot (\boldsymbol{X} \times \boldsymbol{u})\, dV$$

$$\begin{aligned}
\delta\Pi = & \int_{\Omega} \left[\mathbf{J_g} \frac{\partial \psi(\mathbf{F_e}, \boldsymbol{X})}{\partial \boldsymbol{u}} \right] \cdot \delta\boldsymbol{u}\, \mathrm{d}V - \int_{\partial\Omega^{\text{in}}} \boldsymbol{t} \cdot \delta\boldsymbol{u}\, \mathrm{d}\boldsymbol{s} \\
& - \int_{\Omega} \lambda_{\text{tr}} \cdot \delta\boldsymbol{u}\, \mathrm{d}V - \int_{\Omega} \delta\lambda_{\text{tr}} \cdot \boldsymbol{u}\, \mathrm{d}V \\
& - \int_{\Omega} \lambda_{\text{rot}} \cdot (\boldsymbol{X} \times \delta\boldsymbol{u})\, \mathrm{d}V - \int_{\Omega} \delta\lambda_{\text{rot}} \cdot (\boldsymbol{X} \times \boldsymbol{u})\, \mathrm{d}V = 0
\end{aligned}$$

$$\begin{aligned}
\delta\Pi = &\int_{\Omega}\left[\mathbf{J_g}\,\frac{\partial\psi(\mathbf{F_e},\boldsymbol{X})}{\partial\boldsymbol{u}}\right]\cdot\delta\boldsymbol{u}\,\mathrm{d}V - \int_{\partial\Omega^{\text{in}}} p(J\mathbf{F}^{-\mathbf{T}}\boldsymbol{N})\cdot\delta\boldsymbol{u}\,\mathrm{d}\boldsymbol{s}\\
&-\int_{\Omega}\lambda_{\text{tr}}\cdot\delta\boldsymbol{u}\,\mathrm{d}V - \int_{\Omega}\delta\lambda_{\text{tr}}\cdot\boldsymbol{u}\,\mathrm{d}V\\
&-\int_{\Omega}\lambda_{\text{rot}}\cdot(\boldsymbol{X}\times\delta\boldsymbol{u})\,\mathrm{d}V - \int_{\Omega}\delta\lambda_{\text{rot}}\cdot(\boldsymbol{X}\times\boldsymbol{u})\,\mathrm{d}V = 0
\end{aligned}$$

Here, $\boldsymbol{u}$ is the displacement field of the lung from its reference configuration. λ_{tr} and λ_{rot} are the the Lagrange multipliers to constrain rigid body translation and rotation, respectively. $\psi(\mathbf{F}_e,\boldsymbol{X})$ is the elastic energy density calculated from nonlinear strains and assuming both tissues are Neo-Hookean solids (Section 2.2), and its value changes over the integration domain dV because the tissues have different material properties. $J=\det(\mathbf{F})>0$ is the Jacobian of the deformation gradient due to growth/contraction. $\mathrm{d}S, \partial\Omega_0^{\text{in}}$, and $\mathrm{d}s, \partial\Omega^{\text{in}}$ are the infinitesimal surface elements and the inner surface boundary in the reference and deformed configuration, respectively. The virtual work due to pressure p in the second line of the equation above is translated to the undeformed reference configuration by using Nanson's formula, where $\boldsymbol{N}$ is the outer normal in the reference configuration [46] (Fig. 3.3 A).

To solve the minimization problem numerically, the domain Ω was discretized (see Section 3.3) and the potential energy minimization problem was implemented using FEniCS [47]. The Newton-Raphson algorithm [48], modified by adding the flexibility to reduce step sizes for higher deformation, was used to find the displacement field that minimized the total potential energy (Section 2.4.2).

3.4.3 Simulating lung experiments

Lung inflation and puncturing were simulated, with physical parameters adjusted to match experimental observations. At each time step, we computed the current equilibrium configuration by solving for displacements from the energy minimization. Simulation results remained consistent across different time step resolutions. The simulation results were visualized using Paraview software [49].

Inflation

During the inflation stage, luminal pressure was linearly increased from 0 to p_{fin} as $p = n\Delta p_i$, where n is the number of time steps, and we assumed no smooth muscle contraction, i.e., $g_{\text{sm}} = 0$. For simulating inflation, the final pressure was set to $p_{\text{fin}}/E_{\text{ep}} = 0.11$ and $\Delta p_i = p_{\text{fin}}/100$, and inflation was achieved in 100 time steps.

Puncturing

During lung puncture, the luminal pressure was reduced to a fraction of its inflated value linearly as $p = p_{\text{fin}} + n\Delta p_{\text{d}}$ where $\Delta p_{\text{d}} < 0$, while smooth muscle contraction was increased linearly from 0 as $g_{\text{sm}} = n\Delta g_{\text{sm}}$ with $\Delta g_{\text{sm}} < 0$. For simulating lung puncture, pressure was reduced in steps of $p_{\text{d}} = -0.1 p_{\text{fin}}/200$ and anisotropic contraction of smooth muscle along the fibers was increased in steps of $\Delta g_{\text{sm}} = -0.4/200$, thus resulting in a drop of 10% in luminal pressure and smooth muscle contraction of 40% at the end of 200 time steps.

3.5 Analysis

We conducted simulations under varying conditions of luminal pressure, smooth muscle contraction, and smooth muscle stiffness. Simulations using a simplified mechanical model were analyzed to motivate the inclusion of mechanical complexity in the final computational framework. Across all analyses, morphological variation was quantified using the aspect ratio of epithelial corrugations, defined as the ratio of protrusion height to the spacing between muscle fibers in the final deformed state of the lung. Protrusion height (i.e., corrugation depth) was calculated as the difference between the maximum and minimum values of radial displacement $\boldsymbol{u} \cdot \boldsymbol{N}$ of the epithelium, where $\boldsymbol{u}$ is the displacement field of the epithelium and $\boldsymbol{N}$ is the normal in the reference configuration (Fig. 3.3 A).

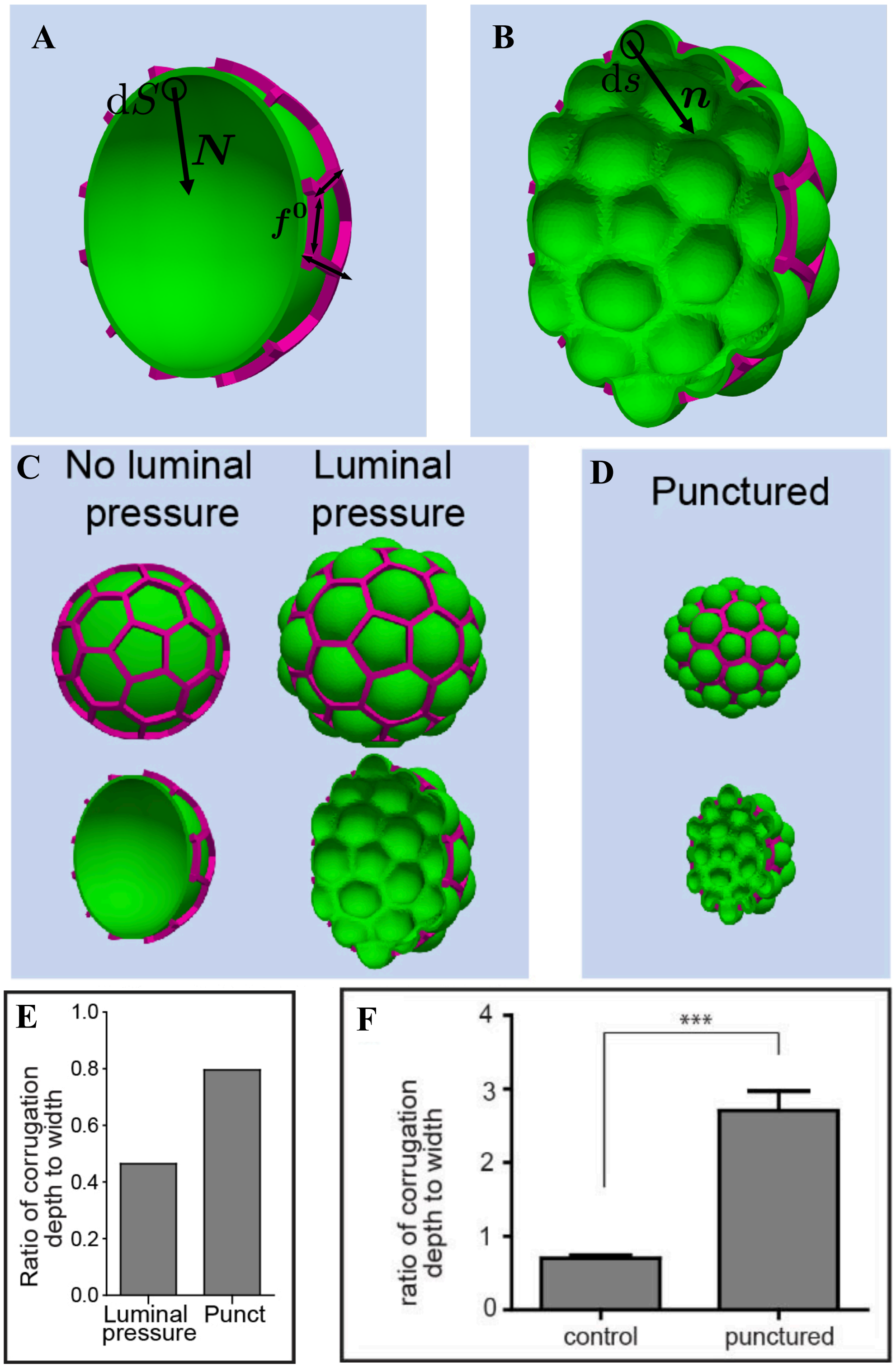


Figure 3.3: **(A)** Parameters of the computational model defined on the reference configuration of the anole lung. **(B)** Parameters of the computational model defined on the current/deformed configuration of the anole lung. **(C)** Results of simulations, showing pressure-induced formation of epithelial corrugations through a passive smooth muscle mesh. **(D)** Results of simulations of lung puncture, showing elongation of epithelial corrugations. **(E)** Graph of the ratio between corrugation depth and width in simulations with and without puncture. **(F)** Graph of the ratio between corrugation depth and width for control and punctured E7 lung explants;***, $p<0.001$ per one-way ANOVA or two-sided t-test.

3.5.1 Effect of luminal pressure

In the absence of luminal pressure, contraction of the smooth muscle mesh produces localized bending without significant outward epithelial protrusion, resulting primarily in overall shrinkage of the epithelial sac. In contrast, increasing luminal pressure in the absence of smooth muscle contraction generated epithelial protrusions that elongated progressively with rising pressure (Fig. 3.4 A).

To represent lumen inflation during lung development and puncturing, the luminal pressure was systematically varied and morphological changes observed. Specifically, the luminal pressure parameter p_{fin} was determined by looking for significant simulated epithelial protrusion dimensions during inflation and puncturing. The corrugation aspect ratio measured from both simulations and experiments increased following puncturing (Fig. 3.3, E and F).

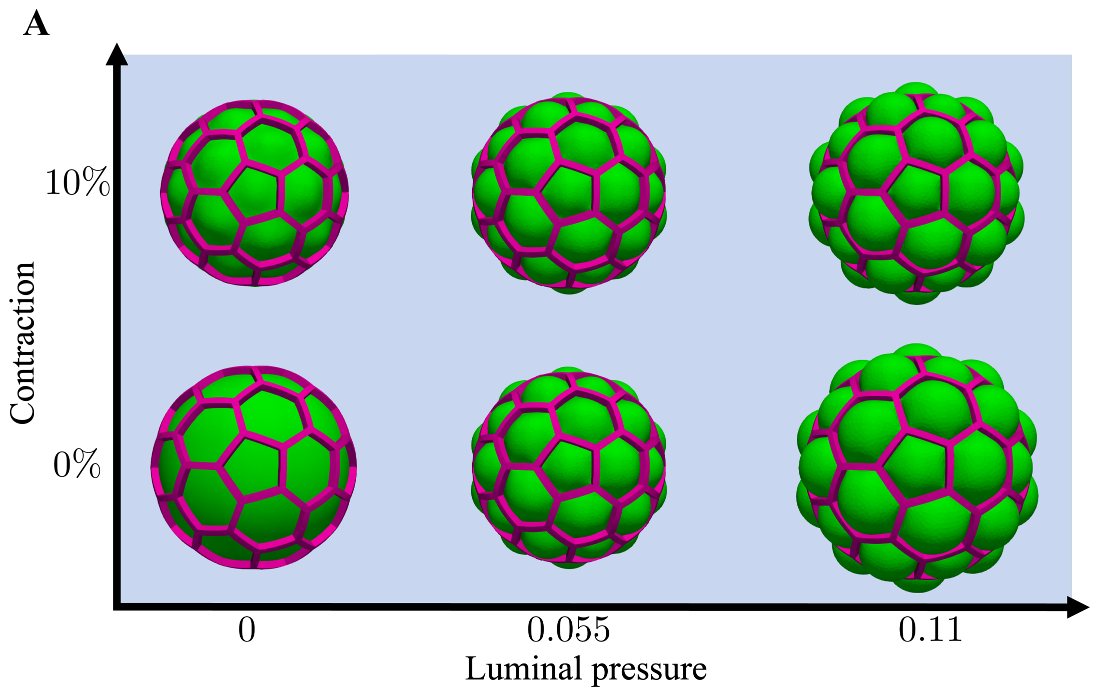
Subsequently, a sensitivity analysis was performed by varying the luminal pressure around p_{fin} by $\pm 5\%$. This analysis revealed that although small changes in luminal pressure quantitatively influenced the deformation, the overall qualitative morphology remained consistent (Fig. 3.4, B-D).

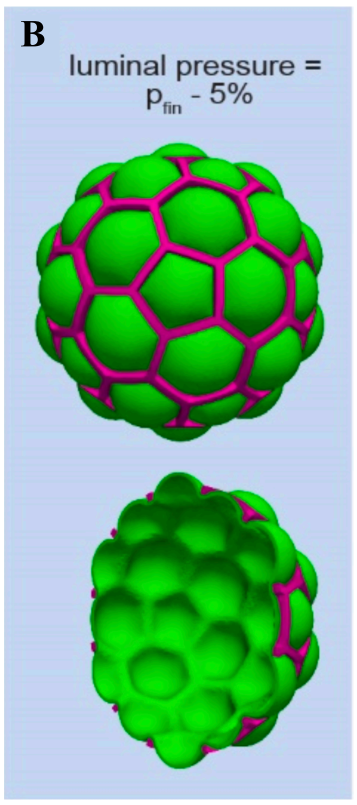
3.5.2 Effect of smooth muscle contraction

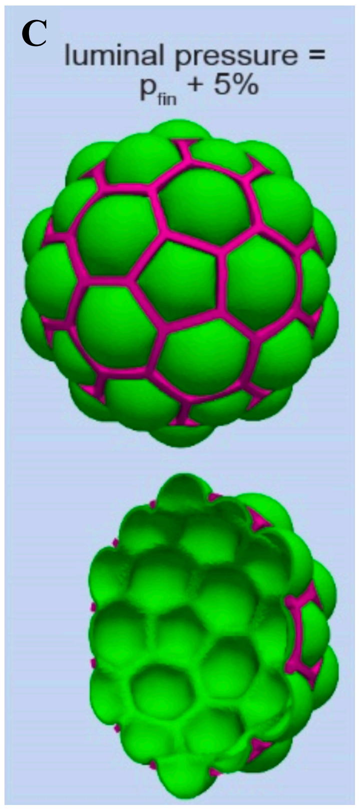
Smooth muscle contraction, in the presence of luminal pressure, plays a significant role in shaping epithelial protrusions. In the puncturing simulations, contraction acts to elongate protrusions that are already present within the tissue (Fig. 3.3 E). Introducing contraction in puncturing simulations is derived from the experimental results in which smooth muscle contraction was inhibited using nifedipine; under these conditions, the increase in protrusion length during puncturing was not observed (Fig. 3.5 E-G).

When the level of contraction is increased in simulations, maintaining all other physical parameters, these protrusions become further extended, indicating a direct relationship between contractile intensity and protrusion length (Fig. 3.3 D and Fig. 3.5, A and B).

To investigate the role of directional contractility, smooth muscle contraction was modeled using







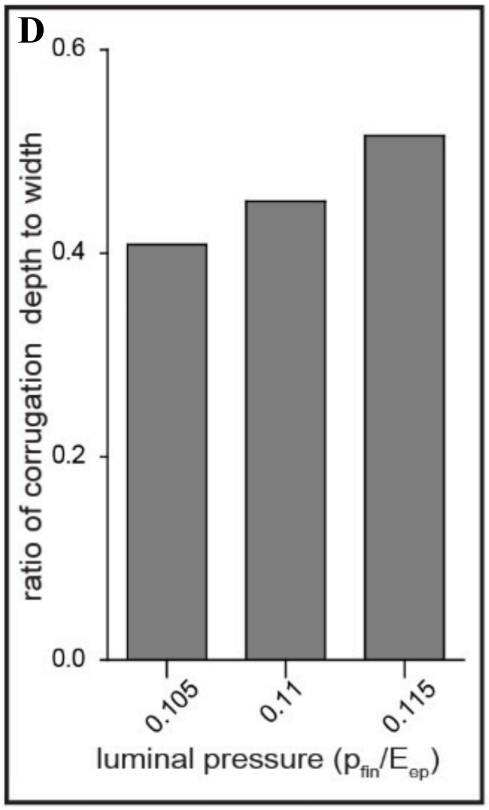


Figure 3.4: **(A)** Simulation results for varying contraction, g_{sm}, and luminal pressure, p_{fin}/E_{ep}. **(B)** Inflated state of the simulated lung when the final pressure is 5% lesser than p_{fin}. **(C)** Inflated state of the simulated lung when the final pressure is 5% greater than p_{fin}. **(D)** Comparison of the aspect ratio of corrugations in inflated state across final luminal pressures.

both isotropic, $\mathbf{F}_{\text{g,sm}} = (1 + g_{\text{sm}})\mathbf{I}$, and anisotropic growth tensors. Simulation results revealed no significant differences in the deformation patterns of the lizard lung during puncturing events (Fig. 3.3 D and Fig. 3.5, C and D). This finding suggests that the directional dependence of smooth muscle contraction does not substantially affect the macroscopic deformation behavior of the lizard lung epithelium.

3.5.3 Effect of smooth muscle stiffness

Smooth muscle stiffness significantly influences the emergence and extent of tissue protrusions. Simulations with varying stiffness ratios revealed a critical threshold, $E_{\text{sm}}/E_{\text{ep}} \approx 25$, below which significant protrusions did not form for the tested conditions (Fig. 3.6 A). Furthermore, protrusion depth plateaued for stiffness ratios $E_{\text{sm}}/E_{\text{ep}} \geq 75$ (Fig. 3.6, B-D). A possible reason might be that beyond this point additional increases in smooth muscle stiffness primarily enhance overall structural rigidity, particularly given the assumption of tissues to be Neo-Hookean elastic solids.

3.5.4 Effect of model complexity

To evaluate the impact of computational model complexity on accurately capturing lizard lung development, we compared results from simulations using identical meshes but two distinct modeling approaches. The primitive model incorporated linearized strain assumptions, linear elastic tissue behavior, and a simplified luminal pressure approach, always oriented radially or perpendicular to the initial undeformed epithelial surface so tractions are $\boldsymbol{t} = -p\boldsymbol{N}$ (Fig. 3.3 A). Conversely, the advanced model, employed as the standard computational framework throughout this study, incorporated improvements including non-linear strain formulations, Neo-Hookean solid behavior of tissues, and a dynamic luminal pressure consistently oriented normal to the epithelium's current surface. Results from both inflation and puncturing simulations indicated that the advanced model produced more realistic representations of lizard lung development. In contrast, the primitive model resulted in unphysical artifacts, including sharper curvatures and

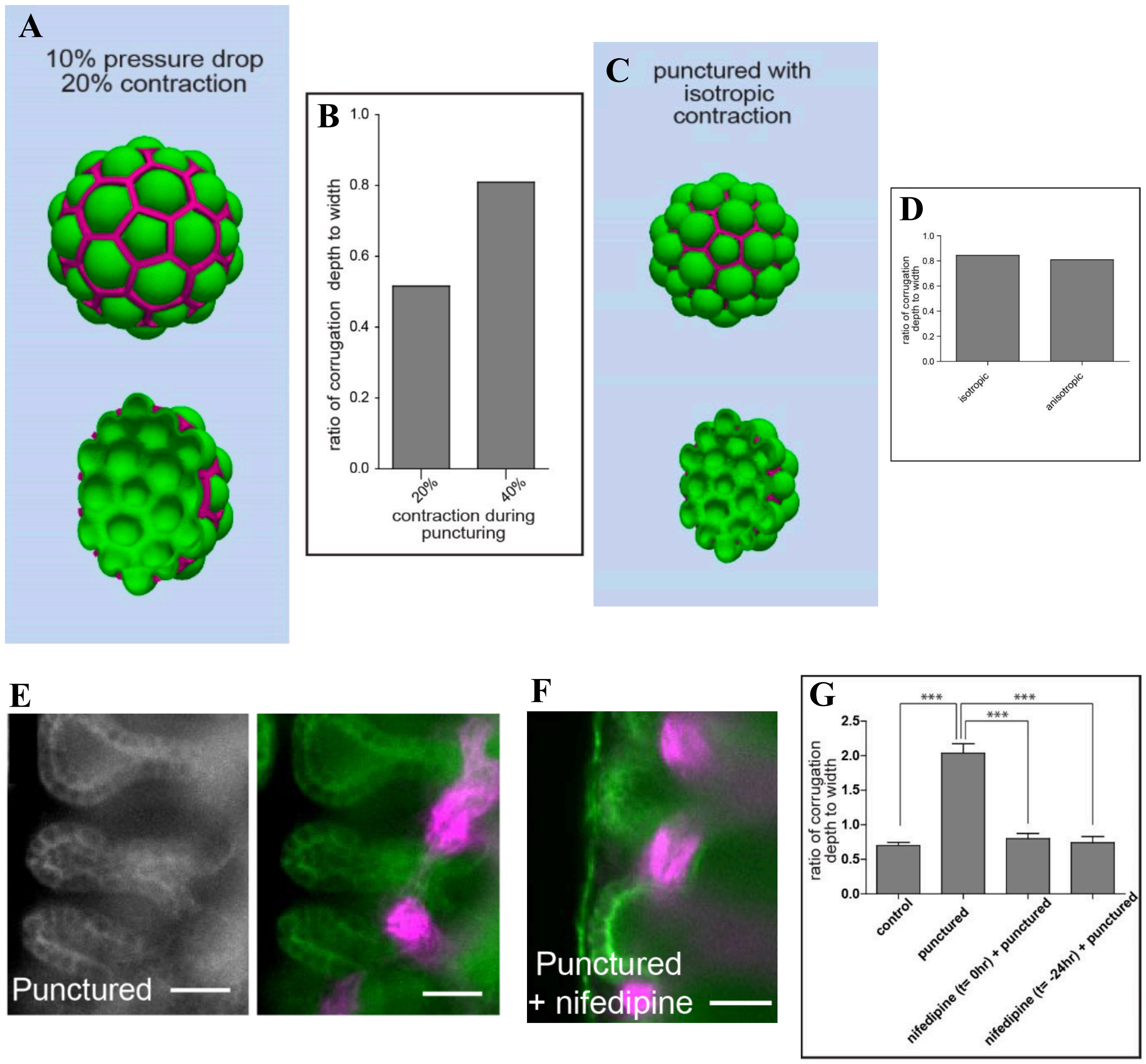


Figure 3.5: **(A)** Punctured state of the simulated lung after contraction of 20% and pressure drop of 10%. **(B)** Comparison of aspect ratio of corrugations between smooth muscle contraction of 20% and 40% during puncturing with a pressure drop of 10%. **(C)** Punctured state of the simulated lung when smooth muscle contraction is implemented as isotropic contractions. **(D)** Comparison of the aspect ratio of corrugations in punctured state between isotropic and anisotropic smooth muscle contraction. **(E)** Immunofluorescence analysis showing epithelium(green) and smooth muscle bundles(magenta) in a E7-stage lung explant that was punctured. Scale bars, 25 μm. **(F)** Immunofluorescence analysis showing epithelium(green) and smooth muscle bundles(magenta) in a E7-stage lung explant that was punctured in the presence of nifedipine. Scale bars, 25 μm. **(G)** Graph of the ratio between corrugation depth and width for unpunctured control lung explants, lung explants punctured without treatment, and lung explants punctured concurrently with addition of nifedipine or after 24 hours of nifedipine treatment. ***, $p<0.001$ per one-way ANOVA or two-sided t-test.

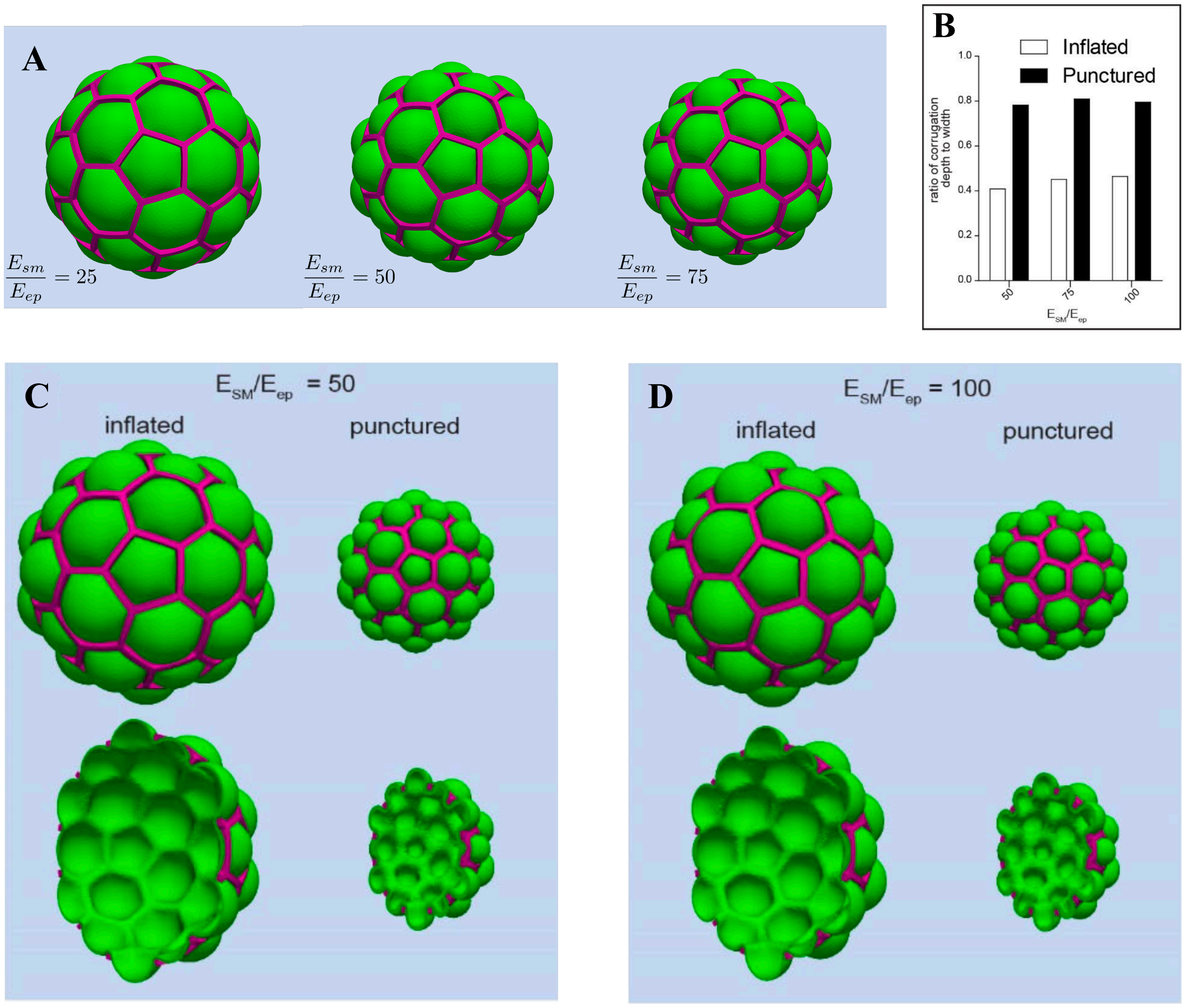


Figure 3.6: **(A)** Variation of inflated states of the lizard lung with different stiffness ratios $E_{\text{sm}}/E_{\text{ep}}$ and same final pressure. **(B)** Comparison of the aspect ratio of corrugations in inflated and punctured states for different smooth muscle stiffness. **(C)** Inflated and punctured states of the simulated lung when the stiffness of the smooth muscle relative to the epithelium is $E_{\text{sm}}/E_{\text{ep}} = 50$. **(D)** Inflated and punctured states of the simulated lung when the stiffness of the smooth muscle relative to the epithelium is $E_{\text{sm}}/E_{\text{ep}} = 100$.

unrealistic spatial variations in epithelial thickness (Fig. 3.3, C and D, and Fig. 3.7).

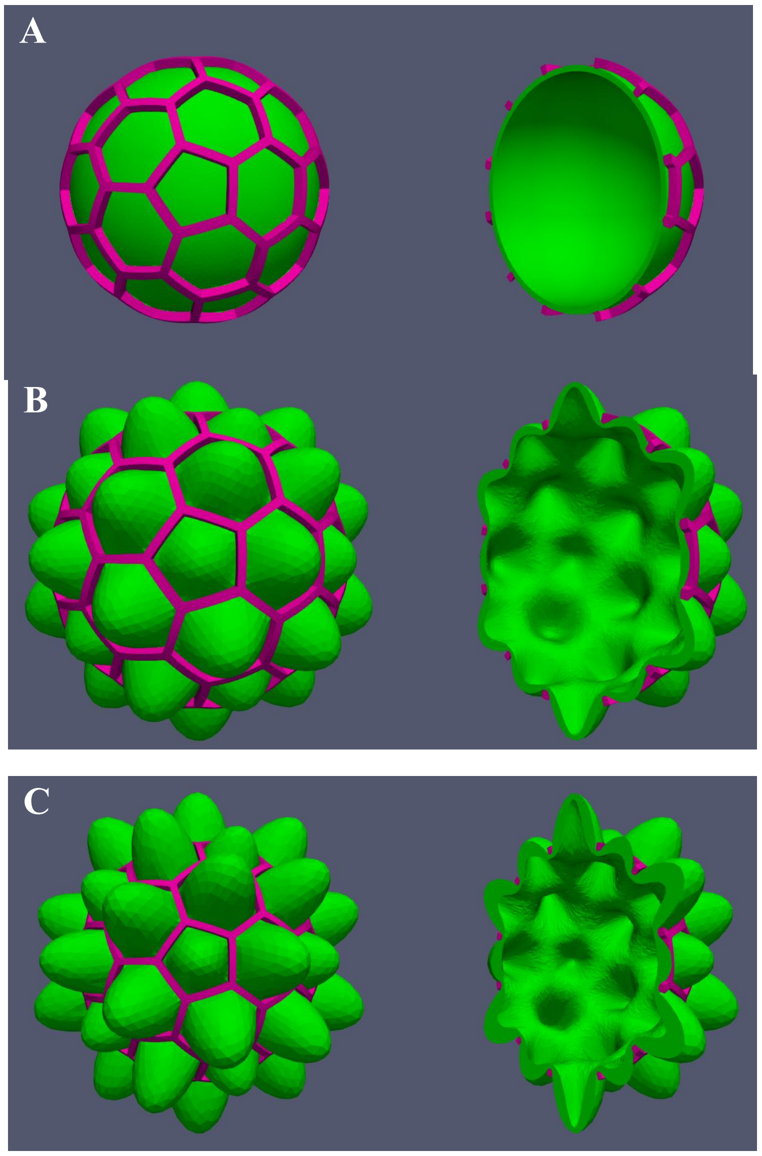


Figure 3.7: Simulation results using primitive physical model with the original mesh. **(A)** Initial state at zero pressure. **(B)** Final state, fully inflated, of the simulation of lumen inflation. **(C)** Final state of the puncturing simulation

3.6 Conclusion

3.6.1 Mechanism of stress-ball morphogenesis

Computational modeling revealed key mechanical drivers of epithelial corrugation in the developing lizard lung.

First, the relative stiffness of smooth muscle compared to the epithelium plays a critical role: a threshold stiffness ratio of smooth muscle to epithelium of approximately 25, $E_{\text{sm}}/E_{\text{ep}} \geq 25$, was found to be necessary for protrusions to form. Below this ratio, epithelial corrugations fail to emerge, highlighting the importance of mechanical contrast between tissue layers.

Second, luminal pressure was identified as a necessary condition for initiating protrusions from deformation. While direct experimental measurements of this pressure are not available, the computational model enabled estimation of this pressure magnitude. Protrusions consistently formed when luminal pressure reached approximately 10% of the epithelial stiffness($p_{\text{fin}} \approx 0.1E_{\text{ep}}$), providing a predictive benchmark for future experimental validation.

Finally, smooth muscle contraction was shown to act primarily by elongating pre-existing protrusions. This mechanism emphasizes the role of contractility not in initiating protrusions, but in possibly amplifying and maintaining their morphology during development.

Together, these findings reveal how tissue-level material properties and mechanisms interact to generate the epithelial architecture observed in the developing lung.

3.6.2 Limitations

While the computational model developed captures key mechanisms of anole lung morphogenesis, limitations remain that could influence the accuracy and extension of the model. First, epithelial proliferation was not included in the simulations. This simplification was made to focus specifically on a developmental window dominated by the mechanical effects of luminal pressure and smooth muscle mesh architecture. Although this assumption is justified for the targeted stage of development, proliferation could contribute significantly to epithelial deformation over longer timescales.

Second, tissue viscoelasticity was not incorporated into the model. Including viscous effects would require a different modeling approach involving real-time dynamics, as the current framework is based on elastic energy minimization which assumes a static solution in response to any parameter changes.

Third, the model assumes a simplified spheroidal geometry for the lung, rather than capturing the full aspect ratio of the inflated epithelium observed experimentally (Fig. 3.1 A). This assumption was made for computational simplicity and to avoid potential numerical instabilities arising from localized stress concentrations or curvature in an asymmetric mesh. Though this abstraction omits certain anatomical details, it is unlikely to affect the conclusions regarding the mechanical origins of epithelial corrugations.

Although the model successfully captures macroscopic aspects of lung shape and protrusion patterning, the extent of protrusion depth—particularly the higher aspect ratios observed in experiments—is not fully replicated (Fig. 3.3 E and F). Incorporating epithelial proliferation and viscoelasticity into future models may allow for a more complete replication of in vivo morphogenesis, particularly the high aspect ratios of epithelial protrusions.

3.7 Applications and Experimental Extensions

To translate the insights from computational modeling into experimental validation that explores the bioengineering potential of these mechanisms for tissue sculpting, two models were developed in this study: a physical model and a bio/synthetic hybrid engineered tissue model [1]. In both experimental models, the lung epithelium was replicated using the same approach. The flexural rigidity of the lung epithelium was calculated as, $D = E_e h_e^3/12(1-\nu^2)$, where E_e is the elastic modulus of the epithelial tissue, ν is Poisson's ratio, and h_e is the epithelial thickness, measured experimentally. Based on known values of E_e [50] and ν [51], along with experimental measurements of h_e, thin films of silicone elastomer were fabricated to approximate the flexural rigidity—and thus the mechanical properties—of the lung epithelium. These elastomer films, floated on water, served as mechanical analogs for the epithelial sheet.

In the physical model, to apply active contractions a synthetic hectorite clay was 3D-printed in a hexagonal mesh pattern onto the silicone elastomer (Fig. 3.8 A). As the clay dried, it contracted, generating localized strain that produced corrugations within the elastomer sheet. These corrugations emerged through the gaps in the mesh, reproducing the topology of epithelial deformations characteristic of the developing anole lung (Fig. 3.8, C and D). This physical model demonstrated that active contraction of a patterned mesh is sufficient to induce corrugations in an underlying compliant sheet subjected to normal pressure.

A bio/synthetic hybrid engineered tissue model was created from an *ex vivo* meshwork of smooth muscle cells using 3D printing and optogenetics. A hexagonal muscle fiber architecture was created using 3D printing, and contractile activity was induced optogenetically via stimulation of calcium influx, yielding temporal control over the magnitude of contraction [1]. The hexagonal mesh of muscle cells contracted upon stimulation with light, generating strain that caused wrinkling of the silicone sheet and the formation of three-dimensional corrugations through the mesh holes (Fig. 3.8, E and F).

These experimental models were designed to validate the mechanical principles revealed through computational simulations, translating *in silico* findings into physical systems that can be directly observed and perturbed. While the bio/synthetic model produced deeper corrugations than the clay-based physical model, both remained shallower than those observed *in vivo* in the anole lung (Fig. 3.8 B). This suggests that additional biological factors, such as increasing luminal pressure, tissue proliferation, or viscoelastic effects, may further sculpt epithelium to mimic morphogenesis. Together, these experimental models provide a bridge between simulation and experiment, demonstrating that stress-ball morphogenesis can be potentially replicated through tunable physical and biological systems. By uncovering the fundamental principles underlying epithelial deformation, this work opens avenues for engineering reproducible epithelial structures *ex vivo*, with potential applications in organoid development and tissue engineering.

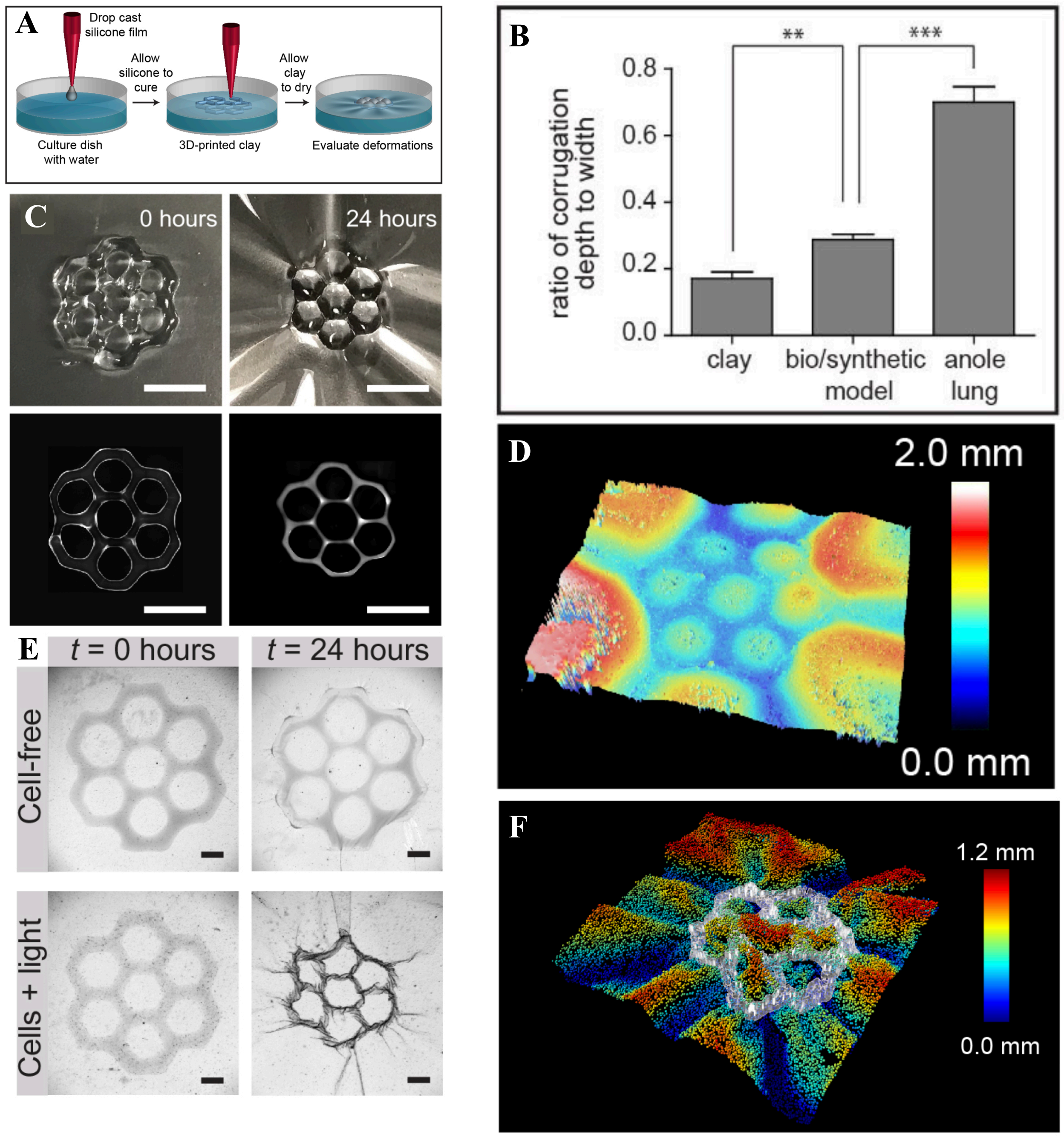


Figure 3.8: **(A)** Procedure for generating a physical model of the embryonic anole lung. **(B)** Graph of the ratio between corrugation depth and width of corrugations for clay, bio/synthetic tissue model, and embryonic anole lungs. **(C)** Images of physical model at 0 and 24 hours after fabrication, showing the formation of corrugations as the clay dries and contracts. Scale bars, 2.5 mm. **(D)** Topology plot of silicone film after the formation of corrugations. Color-coding indicates depth. **(E)** Images of bio/synthetic engineered tissue model with or without cells at 0 and 24 hours after stimulation with light, showing the formation of corrugations. Scale bars, 1 mm. **(F)** Topology plot of bio/synthetic engineered tissue model after 24 hours of cell-induced strain.

Chapter 4

Predictive Design of Large-Scale Self-Folding

This chapter is based on research conducted as part of the NSF FMRG (Future Manufacturing Research Grant): "Threading High-Performance, Self-Morphing Building Blocks Across Scales Toward a Sustainable Future".

4.1 Introduction and Motivation

4.1.1 The Promise and Challenge of Self-Folding Systems

The convergence of materials science and biomimetic design has given rise to a transformative class of adaptive structures: self-folding systems that can autonomously reconfigure their geometry in response to external stimuli [52, 53]. These origami-inspired technologies promise revolutionary applications including deployable space structures and reconfigurable infrastructure systems [54, 55]. Self-folding structures differ fundamentally from traditional mechanical assemblies. Rather than relying on external actuation, they encode deployment logic directly into the material architecture [56]. This approach enables three key advantages: compact storage, simplified mechanisms, and reduced maintenance requirements.

The fundamental appeal of self-folding systems lies in their ability to achieve dramatic shape transformations through minimal energy input and absence of external actuation [17]. By strategically patterning stimuli-responsive materials such as temperature-sensitive polymers, shape memory alloys, or pH-responsive hydrogels, researchers have created structures that fold or unfold to reconfigure autonomously when exposed to appropriate environmental triggers [57, 58]. This capability has profound applications where manual assembly is impractical or impossible, such as space-deployed solar arrays [59], emergency shelter systems, or biomedical implants that must navigate confined anatomical spaces before expanding to their functional configuration [60].

While these applications demonstrate the immense potential of self-folding systems, their practical implementation faces a critical challenge as system dimensions increase: the emergence of weight forces as the dominant energy barrier. At the microscale, where most self-folding demonstrations have been successful, surface forces and elastic energy dominate the mechanical behavior, allowing relatively weak stimuli-induced stresses to drive complete folding transformations [61]. As system dimensions increase toward engineering-relevant scales, spanning meters rather than millimeters, two competing effects emerge. Weight forces due to gravity grow with the cube of the characteristic length, while elastic restoring forces scale only with the cross-sectional area [62]. This fundamental scaling mismatch creates an increasingly unfavorable energy balance that can prevent successful folding or lead to incomplete, unstable configurations. This represents the primary obstacle preventing the translation of promising microscale self-folding demonstrations into practical macroscale engineering systems.

4.1.2 Current State of Self-Folding Technology

Self-folding research has demonstrated remarkable capabilities across diverse material platforms and actuation mechanisms. Temperature-responsive polymer systems, exemplified by the work of Na et al. [63], have achieved complex three-dimensional folding sequences through the strategic

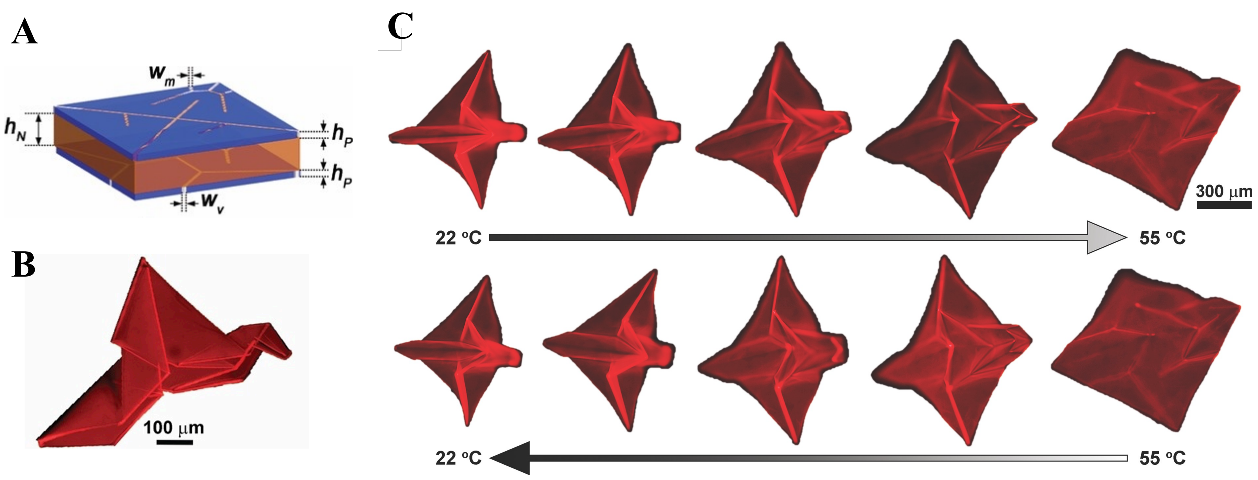
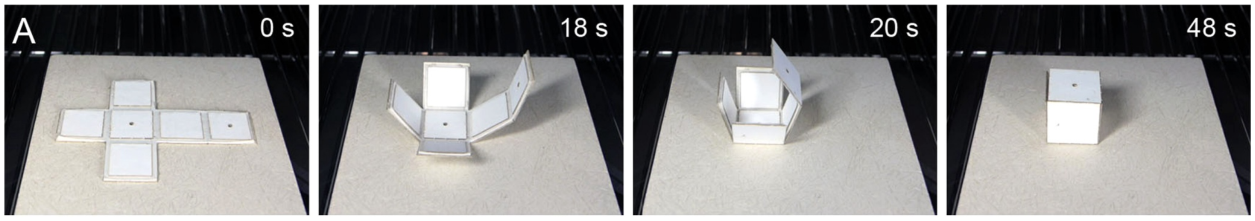


Figure 4.1: **(A, B)** Self-folding microscale origami patterns are demonstrated in polymer films with control over mountain/valley assignments and fold angles using trilayers of photo-crosslinkable copolymers with a temperature-sensitive hydrogel as the middle layer. The characteristic size scale of the folds w = 30 μm. **(C)** Thermal actuation of self-folding origami. When the temperature is increased, the hydrogel polymer layer deswells, causing the Randlett's bird to revert to an unfolded, nearly flat, shape by 55°C. Upon cooling to 22°C, reswelling back to the folded state occurs through a similar pathway. Dry thicknesses of h_P = 70 nm and h_N = 1.5 μm are used, while the size of the square sheet is 800 μm on a side. [63]

patterning of materials with different thermal expansion coefficients (Figure 4.1). These bilayer and trilayer architectures exploit differential thermal strains to generate curvature changes that drive folding motions, with demonstrated applications in microrobotics and biomedical devices. Shape memory polymer systems represent another promising approach, offering programmable folding sequences that can be triggered by heating above the glass transition temperature [65]. The work of Tolley et al. [64] has shown how composite structures incorporating both hyperelastic and shape memory polymer components can achieve controlled folding with well-defined intermediate states (Figure 4.2). These systems demonstrate particular promise for applications requiring precise timing and sequencing of folding events [66].

At larger scales, several demonstrations have captured the potential of self-folding systems for different applications. The BYU-developed origami-inspired solar array represents one of the most successful examples of scaling self-folding principles to space applications, where the absence of gravity in the environment enables folding to be executed with lesser energy than



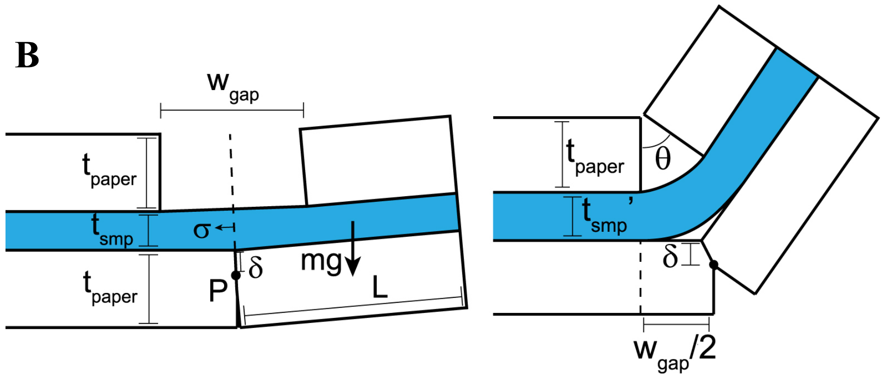


Figure 4.2: **(A)** Self-folding experiments. These image sequences are from self-folding experiments for the cube, the self-folding shape memory composite was inserted into an oven preheated to 130°C. The time elapsed since the start of the experiment is indicate in the upper-right corner of each frame and the maximum dimension in folded state is 15mm. **(B)** Schematic cross-section of a shape memory composite self-folding hinge. The shape memory composite consists of a layer of shape memory polymer (SMP, colored here in blue), with two patterned layers of paper (colored here in white). Left: when the laminate is activated with uniform heating, the SMP layer contracts, causing the laminate to bend towards the side with a gap about the perpendicular axis P, fighting against a moment caused by the force of gravity on the free end, mg. Right: the laminate continues bending until the two sides of the upper layer come into contact, resulting in a final bend angle θ. [64]

under terrestrial conditions (Figure 4.3) [54]. Similarly, deployable shelter systems have demonstrated the practical utility of origami-inspired design principles, though these typically rely on external actuation rather than autonomous self-folding mechanisms (Figure 4.4) [67, 68].

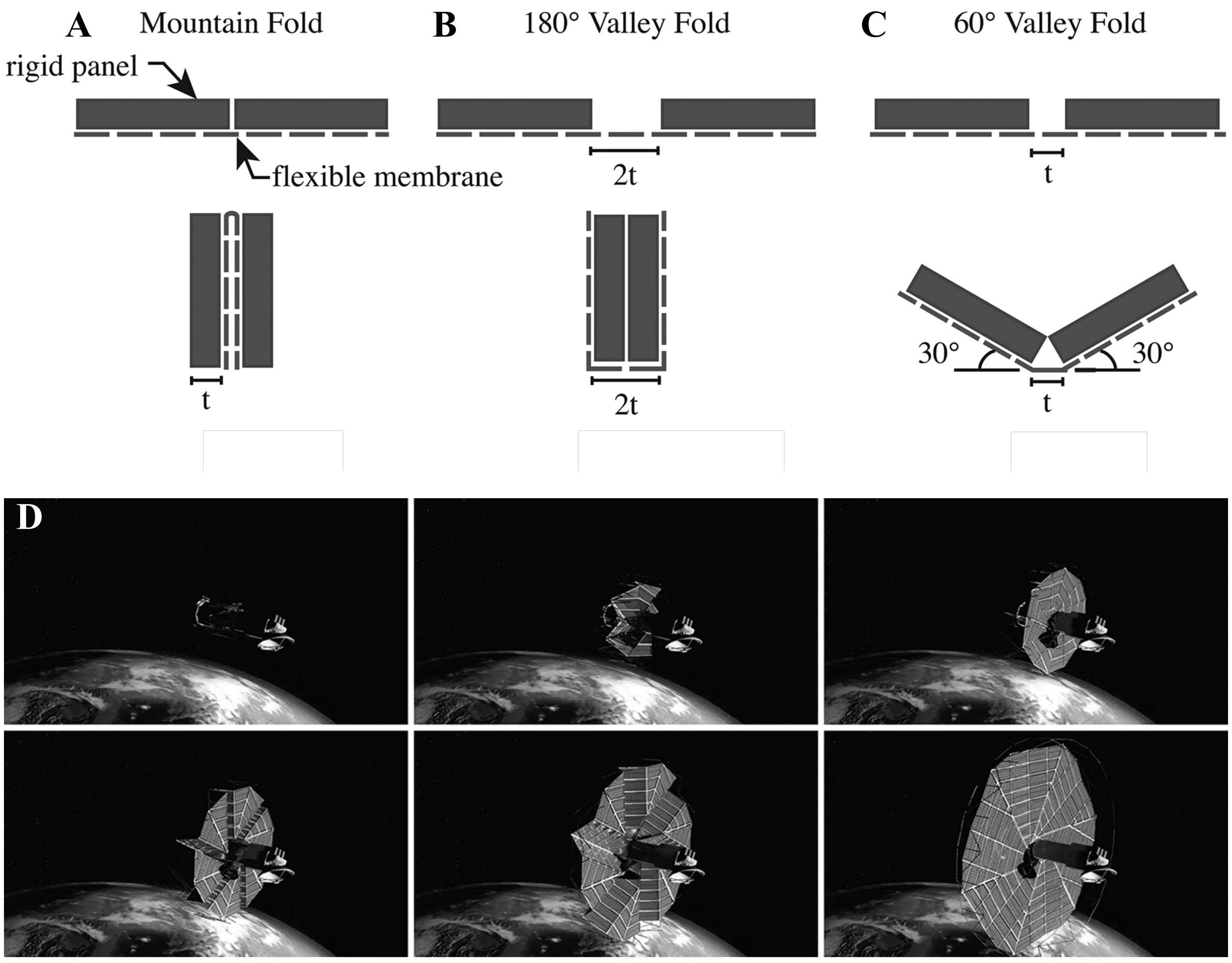


Figure 4.3: The folds of the membrane model in [54] are illustrated with the panels as gray blocks and the membrane as the dashed line. The gap sizes defined here are lower limits. **(A)** The 180 deg mountain folds require minimal gap between panels because the membrane folds back on itself. **(B)** The 180 deg valley folds require a minimum of twice the thickness of the panels because the membrane folds around two panels. **(C)** The 60 deg valley folds require a minimum of one thickness because of the triangle formed by the panel edges. **(D)** A computer-generated model of the solar array, constructed from multiple different folds, to work on a spacecraft.

Despite these successes, there exist gaps between proof-of-concept demonstrations and large-scale technologies that can be deployed on ground. Most self-folding demonstrations

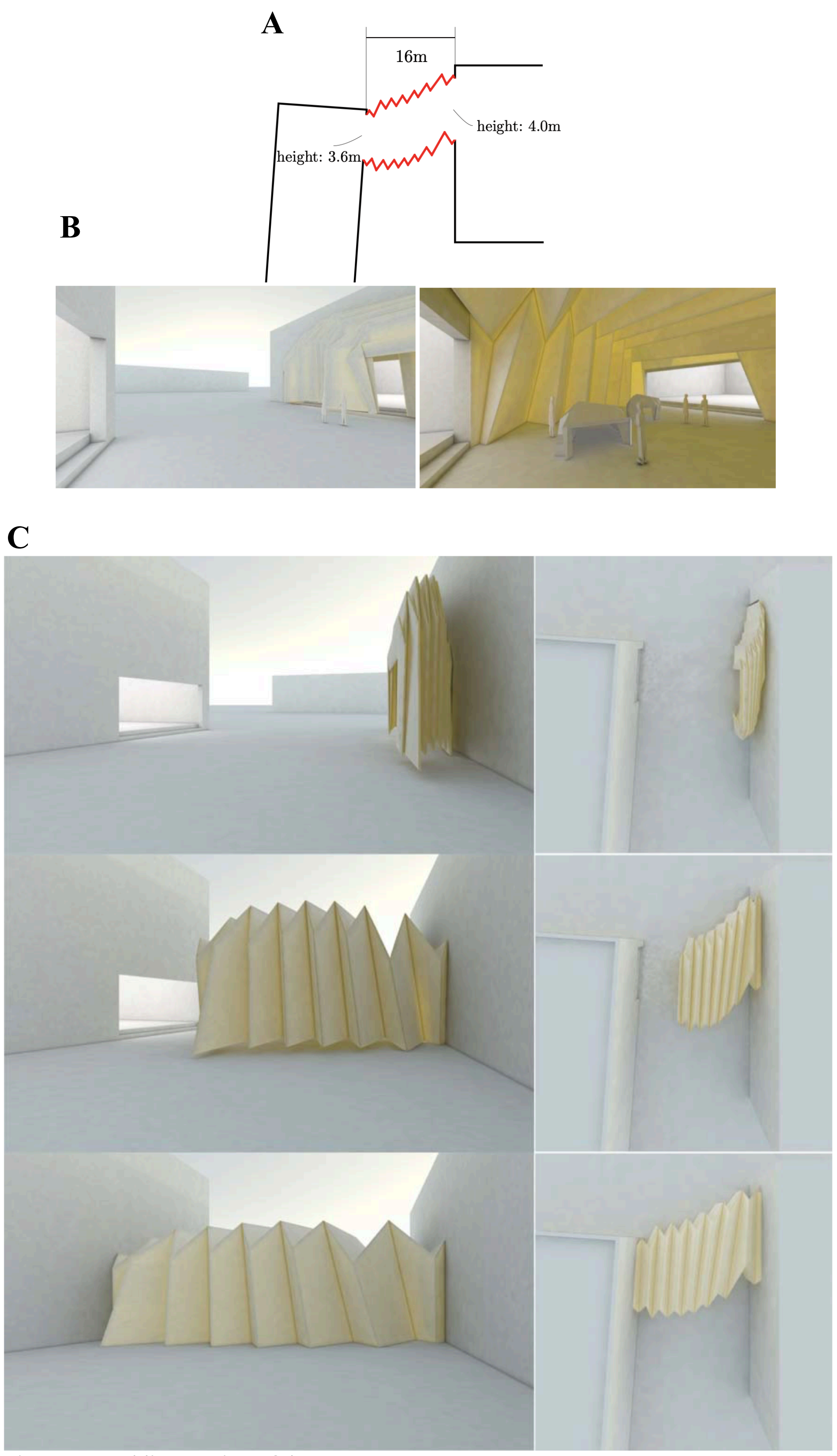


Figure 4.4: **(A)**The layout to build a foldable space by connecting the openings of existing two separate buildings having different sizes and orientations. **(B)** Perspective view of the initial and deployed states of the folded structure. **(C)** The folding motion of the resulting structure designed. [67]

operate at length scales below one centimeter, where gravity effects are negligible compared to elastic and surface forces. The few examples of larger-scale self-folding systems either operate in reduced-gravity environments or require external assistance to overcome weight forces. This limitation has created a clear technological gap: while the fundamental principles of self-folding are well-established, the design tools and methodologies needed to systematically engineer these systems for terrestrial applications at practical scales remain underdeveloped.

The absence of reliable predictive models represents a critical limitation. Current design approaches rely heavily on empirical iteration and prototype testing, making it difficult to systematically explore design spaces or optimize performance for specific applications. Without quantitative frameworks for predicting folding success or failure as functions of material properties and geometric parameters, engineers cannot confidently specify system requirements or assess manufacturability. This limitation significantly impedes the technology transfer process from experimental research to real-world applications.

4.1.3 Research Objectives and Scope

Our research tackles the challenge of designing self-folding systems that reliably function when gravity becomes a significant contributor to potential energy, thereby enabling engineering or architectural implementation of these adaptive structures. We aim to develop a comprehensive computational framework that links material properties and geometric parameters to folding performance, paving the way for systematic design and optimization. Our approach differs from previous research efforts by explicitly focusing on scaling challenge of folding in the presence of gravity and developing precise design tools for self-folding systems. Rather than pursuing incremental scale improvements to existing microscale demonstrations, we directly address the physics that has limited the practical implementation of self-folding technologies, providing a pathway toward reliable, predictable, and deployable self-folding systems at larger scales. By developing a reliable design workflow, this work accelerates the translation of self-folding technologies from laboratory prototypes to practical engineering solutions for deployable structures, adaptive buildings, and reconfigurable infrastructure systems (Figure 4.5).

Our approach integrates continuum mechanics modeling with machine learning methodologies to create predictive capabilities spanning multiple scales and design parameter ranges. By establishing the bilayer hinge with attached panels as the fundamental unit of analysis, we develop both forward prediction capabilities that determine folding outcomes from specified design parameters and inverse design methodologies that identify parameter combinations to achieve target performance criteria. We pursue two interconnected objectives: forward and inverse design capabilities for self-folding systems (Figure 4.6). To achieve these objectives, we develop a comprehensive computational framework comprising three key independent components. First, we develop a physics-based computational model that accurately captures the competition between elastic stress, gravity, and contact forces during folding processes. These models employ energy minimization principles to simulate shape change of a self-folding unit to its final equilibrium configuration. This computational model enables us to assess folding success for specified material properties, geometry, and stimulus response. Second, we create algorithms that evaluate folding outcomes based on final shapes predicted by simulations, enabling the analysis of large datasets of simulation results. Third, we integrate these mechanistic models with machine learning approaches to discover interpretable relationships between design parameters and folding performance, facilitating both fundamental understanding and practical design synthesis.

4.2 The Foldability Problem for Active Bilayer Hinges

Having identified the fundamental scaling challenge that prevents self-folding systems from operating reliably under terrestrial gravity, we now develop a systematic framework for predicting and designing large-scale folding behavior. This section establishes the bilayer hinge with

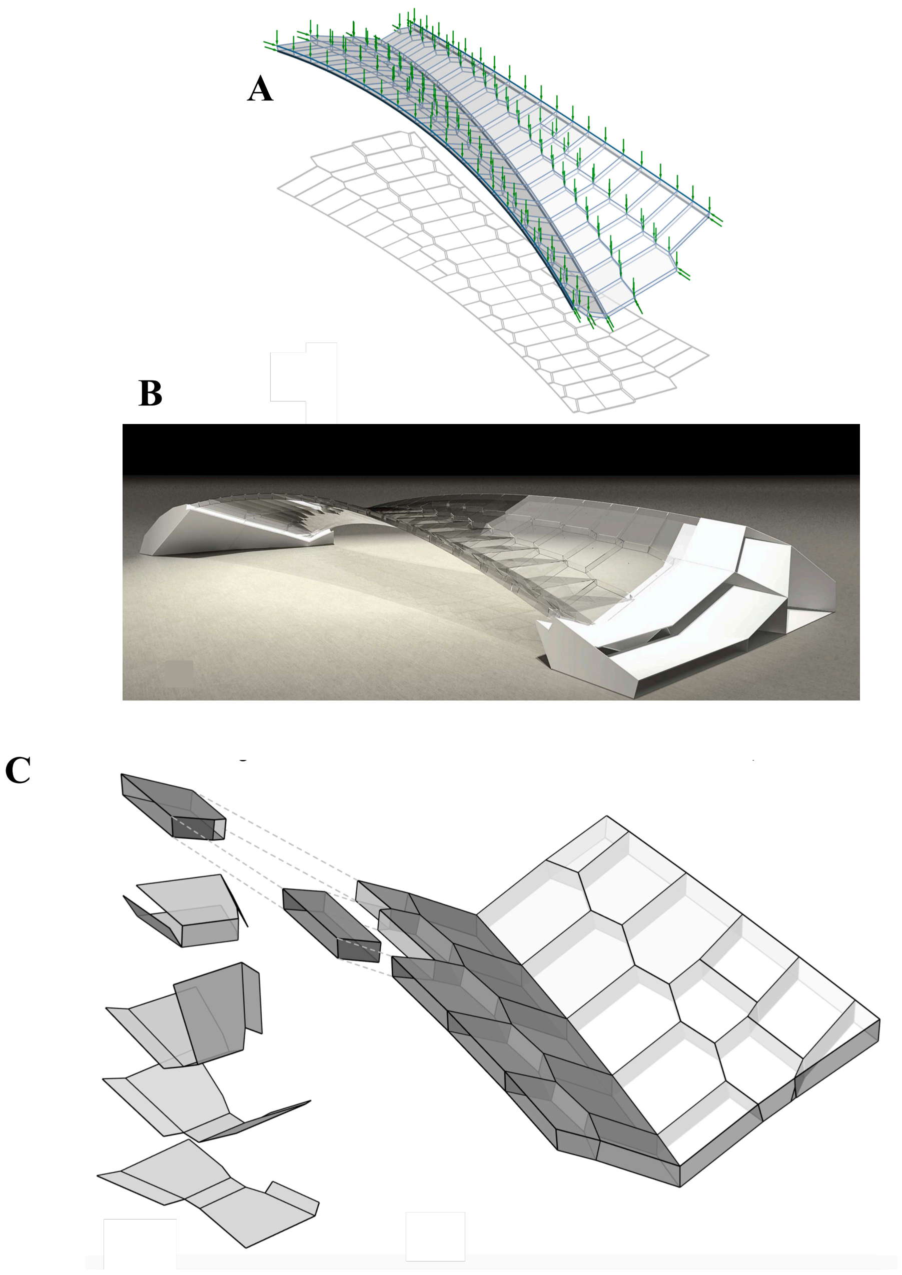


Figure 4.5: **(A)** A funicular structural form of a load-bearing bridge design optimized using 3D graphic statics. **(B)** An early visualization of the final bridge structure. **(C)** The schematic for aggregation of self-folding modules to form a multi-layered construction of the bridge, one of the applications for large-scale self-folding blocks which have the potential to be a energy efficient alternative for sustainable architecture. Design, visualizations, and schematic done at Polyhedral Structures Laboratory, University of Pennsylvania

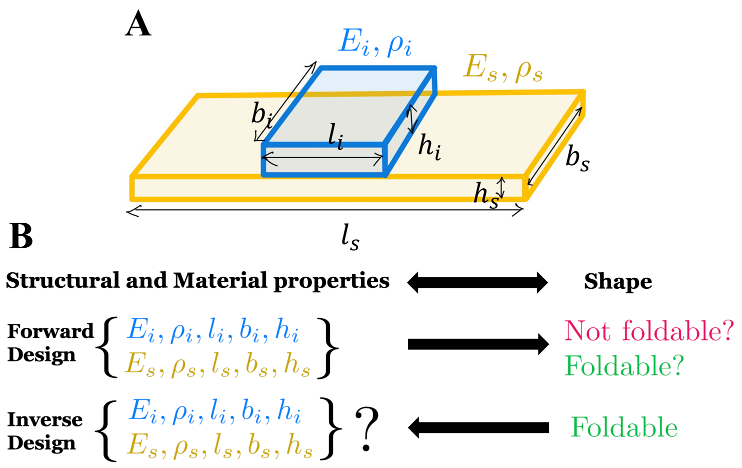

A
E_i, ρ_i
E_s, ρ_s
b_i
l_i
h_i
b_s
h_s
l_s
B
Structural and Material properties
Shape
Forward Design
$E_i, \rho_i, l_i, b_i, h_i$
$E_s, \rho_s, l_s, b_s, h_s$
Not foldable?
Foldable?
Inverse Design
$E_i, \rho_i, l_i, b_i, h_i$
$E_s, \rho_s, l_s, b_s, h_s$
?
Foldable

Figure 4.6: **(A)** Self-folding unit characterized by material and structural properties, with active(blue) and passive(yellow) layer. Subscript "i" and "s" denotes property belongs to active or passive layer respectively. E is the Young's modulus, ρ is the material density, (l, b, h) are the material dimensions of length, breadth, and height. **(B)** Summary of the forward and inverse design problems. In the former, given the structural and material properties we predict the final foldability outcome or deformed shape. In the latter, we determine the combination of these properties required to result in a given desired folded shape.

attached panels as our self-folding unit, conducts dimensional analysis to reduce the design space to key dimensionless groups, and formulates the forward and inverse design problems that will guide our computational approach.

4.2.1 The Self-Folding Unit: Bilayer Hinge with Panels

The self-folding demonstrations reviewed in Section 4.1.2 reveal a common underlying principle: successful autonomous folding invariably emerges from spatial heterogeneity in material response to stimuli, where different regions of the structure exhibit distinct dimensional changes, stiffnesses, or activation thresholds. This observation motivates our selection of the bilayer architecture as the canonical self-folding unit, wherein two materials with contrasting stimuli-responsive properties are intimately coupled to generate the differential deformation necessary for controlled curvature development and subsequent folding motion.

We establish the bilayer hinge with attached panels as the self-folding unit that captures the essential physics governing folding behavior while remaining sufficiently simplified to enable rigorous mathematical treatment. This fundamental self-folding unit enables systematic understanding and prediction of foldability across scales.

Definition

We define the folding unit with a bilayer hinge positioned at the fold line. The bilayer comprises of an active stimuli-responsive layer and a passive structural layer. Upon exposure to appropriate stimuli, such as thermal activation or chemical triggers, the active layer undergoes controlled dimensional changes while the passive layer remains geometrically stable. This differential expansion or contraction between the two layers creates a mismatch in preferred curvature, generating bending moments that drive the development of curvature along the folding edge (Figure 4.7). We recognize that the magnitude and direction of this curvature depend critically on the material properties and geometrical dimesnions of both layers and the degree of stimuli-induced dimensional change in the active component.

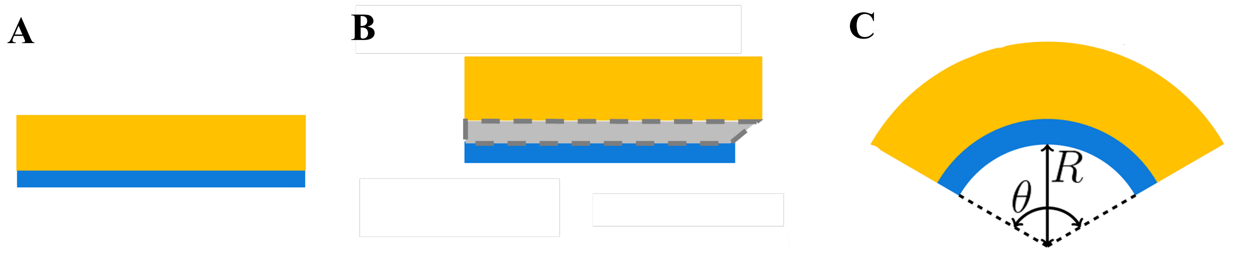


Figure 4.7: Bilayer hinge. **(A)** The bilayer hinge made by attaching two materials of the passive(yellow) layer and active(blue) layer. **(B)** Virtual state showing differential contraction between the active and passive layers, where the active layer contracts but there is no change in the passive layer. The gray area represents the attachment between layers at the surface of contact, the layers have different lengths but are constrained to maintain contact. **(C)** The resulting final shape, where the bilayer bends to minimize residual stresses from differential contraction under contact constraints. The bending is characterized by approximating the bilayer as an arc, the radius of curvature of which is R and subtending an angle θ.

We extend the complete self-folding unit beyond the bilayer hinge to include panels that attach to

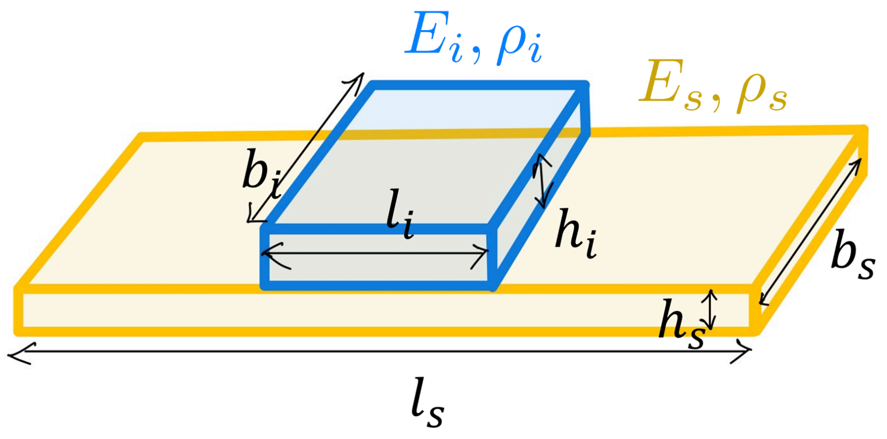


Figure 4.8: Self-folding unit characterized by material and structural properties, with active(blue) and passive(yellow) layer. Subscript "i" and "s" denotes property belongs to active or passive layer respectively.

either side of the actuating element. These panels provide the geometric surfaces that define the folded configuration and their weight opposes the folding motion at engineering scales. We identify that successful folding behavior emerges from the competition between the elastic energy stored in the bent bilayer hinge, which drives folding, and the potential energy associated with lifting and reorienting the attached panels against gravity, which resists folding.

Identifying design parameters

We specify the geometric parameters of this fundamental self-folding unit through multiple geometric parameters that collectively determine folding performance (Figure 4.8). We characterize the bilayer hinge itself by its length l_i, which defines the extent of the active region, and the individual thicknesses of the active and passive layers, denoted h_i and h_s respectively. We determine the total hinge thickness as $h_i + h_s$, while the relative thickness ratio h_i/h_s governs the neutral axis position and bending stiffness characteristics. The attached panels contribute additional geometric parameters including their total length l_s, width b_s, and thickness h_s, where the panel thickness matches that of the passive hinge layer for structural continuity.

We characterize both mechanical and stimuli-response parameters for each layer component. We capture the elastic behavior of both active and passive layers through their respective elastic moduli E_i and E_s, while their densities ρ_i and ρ_s determine weight contributions. Additionally, we characterize stimuli-response parameters as growth tensors $\mathbf{G}_\text{i}$ and $\mathbf{G}_\text{s} = \mathbf{I}$, that govern the degree of dimensional change achievable in the active layer, directly controlling the maximum curvature and folding angles attainable by the self-folding unit.

Handling contact surfaces

We integrate components to form the self-folding unit through careful consideration of mechanical coupling and boundary condition specification, recognizing that two critical contact surfaces exist within the self-folding unit architecture. The first contact surface occurs at the interface between the active and passive layers of the bilayer hinge, where the effectiveness of mechanical coupling determines the degree to which differential dimensional changes translate into coherent bending deformation. The second contact surface exists between the panels and the hinge assembly, where the nature of this attachment governs the efficiency of moment transfer from the actuating hinge to the structural panels that define the folded geometry.

In our theoretical development and computational modeling of the self-folding unit, we adopt the simplifying assumption of perfect attachment at both critical contact surfaces to focus our analysis on the fundamental physics governing foldability while maintaining mathematical tractability. At the bilayer interface, we assume perfect adhesion between active and passive layers, ensuring that differential dimensional changes result in coherent bending without slip or delamination (Figure 4.7 (B)). For the panel-hinge interface, we model the panels as simple continuous extensions of the passive layer, oriented perpendicular to the hinge axis, thereby eliminating any incompatibility and ensuring perfect moment transfer from the actuating hinge to the folding panels. This geometric idealization represents the panels as integral components of a monolithic structure rather than separately attached elements, eliminating potential failure modes associated with joint mechanics while preserving the essential physics of gravity and elastic

$$\begin{array}{c} E_i, \rho_i, l_i, b_i, h_i \\ E_s, \rho_s, l_s, b_s, h_s \end{array} \longrightarrow \frac{E_i}{E_s}, \frac{\rho_i}{\rho_s}, \frac{l_i}{l_s}, \frac{b_i}{b_s}, \frac{h_i}{h_s}, \frac{E_s h_s^2}{\rho_s g l_s^3}, \frac{b_s}{l_s}, \frac{h_s}{l_s}$$

Figure 4.9: Non-dimensionalization condenses the structural and material properties of the self-folding unit to non-dimensional groups.

actuation that govern large-scale foldability.

By adopting these idealizations, we establish a theoretical framework that captures the fundamental scaling relationships and design principles governing self-folding unit performance while remaining sufficiently simplified to enable systematic analysis and optimization across diverse parameter ranges and application requirements.

4.2.2 Dimensional Analysis of the Single-Unit Problem

We conduct systematic dimensional analysis to identify the important parameters governing foldability and to reduce the extensive parameter space to a manageable set of dimensionless groups (Figure 4.9). We begin this analysis with the identification of characteristic scales for length, energy, and time that capture the essential physics of the self-folding with gravity.

We identify the characteristic length scale from the panel dimensions, with l_s serving as the primary geometric scale. We recognize alternative length scales including the hinge length l_i and the various thickness parameters, but we establish that the panel length is typically held constant to compare the different folding responses due to its influence on the moment arms. We derive the characteristic energy scale from the competition between elastic energy stored in the deformed hinge and potential energy associated with panel displacement against gravity.

We also group the geometric parameters into characteristic aspect ratios, such as h_i/h_s, l_i/l_s, and

b_s/l_s, that prove essential for understanding localized bending and consequentially folding performance.

Similar to the geometric parameters, we choose the material properties of the passive layer as the scale for those properties. We identify the material property ratios E_i/E_s and ρ_i/ρ_s as design parameters that can influence both the magnitude of folding moments and the resistance to folding.

Inspired by the deflection of beams under their weight, we establish another dimensionless parameter as the ratio of bending resistance to weight, expressed as $(E_s h_s^2)/(\rho_s g l_s^3)$. We interpret this parameter as quantifying the fundamental competition in the panels between remaining straight by resisting bending and bending under their weight. When this ratio is large, elastic effects dominate and the self-folding unit proceeds to fold, while small values indicate weight dominance causing localized bending in the panels that doesn't translate to foldability. Hence, this dimensionless grouping is key to designing for foldability.

We demonstrate through dimensional analysis and the systematic reduction of governing parameters that the fundamental behavior of the self-folding unit depends on a limited number of dimensionless groups rather than the full set of individual parameters. This reduction dramatically simplifies both the modeling and the practical design optimization process by reducing the dimension of the design space. We develop this dimensional analysis framework to provide the foundation for scaling self-folding units from successful computational demonstrations to practical engineering applications.

4.2.3 Forward and Inverse Design Problem

Using the established design space of non-dimensional groups, we formulate both forward and inverse design problems. We develop these two approaches to provide a comprehensive framework for developing reliable self-folding units across diverse scales.

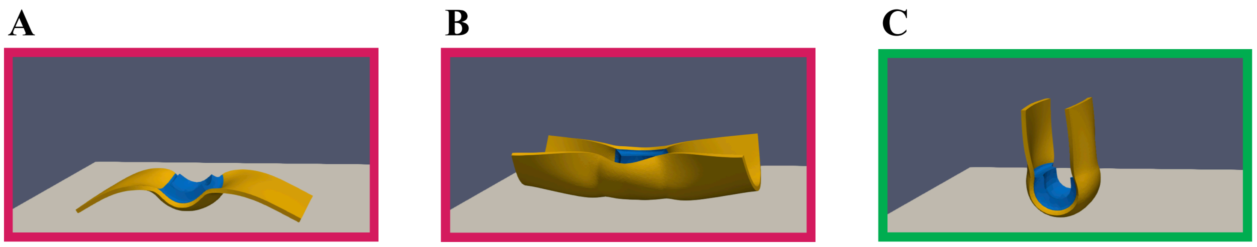


Figure 4.10: Different foldability outcomes from simulating the deformation of the self-folding unit. The foldability outcome of true(green) or false(red) is indicated by the box color of the images of final shapes of self-folding units.**(A)** The bending of the bilayer doesn't lift up the panels to be folded. **(B)** The bilayer bending did not translate to the panels effectively folding along the preferred direction. **(C)** The bilayer bending lifts up the panels to be folded.

Foldability

Before we formulate the forward and inverse design problems, we abstract the desirability of the final shape achieved by creating a parameter called *foldability*. We define successful foldability as achieved when the bilayer hinge can drive the attached panels from their initial flat configuration to a stable, geometrically defined folded state under the influence of stimuli-induced differential contraction (Figure 4.10). We ensure this definition encompasses both kinematic requirements, the ability to achieve the desired geometric transformation, and stability requirements, the capacity to maintain the folded configuration under external perturbations. We require that successful foldability further implies that the folding process occurs within acceptable stimuli-driven dimensional changes and does not induce material failure or undesirable deformations. We capture all these requirements in our foldability parameter which we define as a binary variable.

Forward Design

In the forward design problem, we predict the folding outcomes given complete specification of material properties, stimuli-response, and geometric parameters (Figure 4.10). We structure this problem to take as input the set of non-dimensional parameters, defined in Section 4.2.2, to

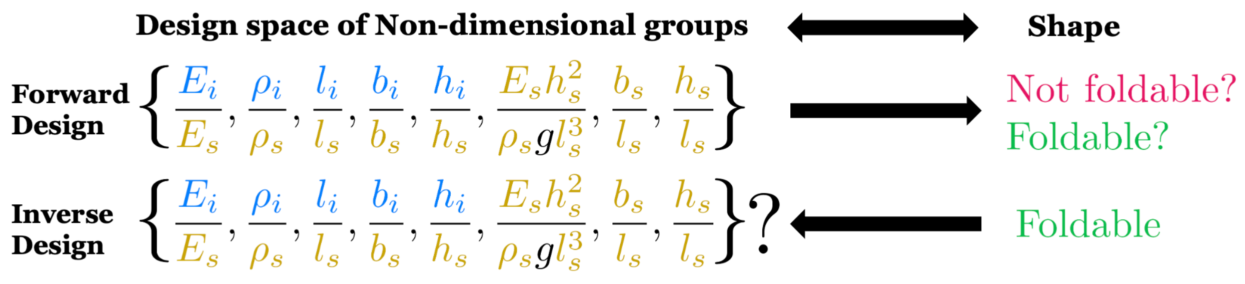

Design space of Non-dimensional groups
Shape
Forward Design
$\left\{\frac{E_i}{E_s}, \frac{\rho_i}{\rho_s}, \frac{l_i}{l_s}, \frac{b_i}{b_s}, \frac{h_i}{h_s}, \frac{E_s h_s^2}{\rho_s g l_s^3}, \frac{b_s}{l_s}, \frac{h_s}{l_s}\right\}$
Not foldable?
Foldable?
Inverse Design
$\left\{\frac{E_i}{E_s}, \frac{\rho_i}{\rho_s}, \frac{l_i}{l_s}, \frac{b_i}{b_s}, \frac{h_i}{h_s}, \frac{E_s h_s^2}{\rho_s g l_s^3}, \frac{b_s}{l_s}, \frac{h_s}{l_s}\right\}?$
Foldable

Figure 4.11: The design problems reframed with the new design space after non-dimensioanlization.

predict the resulting folding behavior through computational methods that follow the framework outlined in Chapter 2. Using our definition of foldability, we translate the final shape produced by the simulation to resulting the foldability parameter. Therefore, in solving the forward design problem we are able to predict what the shape and foldability outcomes are for any given self-folding unit.

Inverse Design

We structure the inverse design problem to present a different challenge, beginning with desired folding behavior and seeking to determine the optimal combination of material properties and geometric parameters that achieve reliable foldability under specified stimulus (Figure 4.11). We design this problem formulation to acknowledge that engineering applications typically begin with functional requirements, such as extent of folding, rather than arbitrary material and geometry specifications. We require the inverse design problem to navigate the complex relationship between design parameters and self-folding unit performance to identify feasible design solutions.

We recognize that the mathematical complexity of the inverse design problem arises from the highly nonlinear (and with gravity, unexplored) dependence of folding behavior on design parameters. Traditional gradient-based optimization approaches are not feasible in the absence of established analytical relations describing folding behavior in the presence of gravity. We therefore solve the inverse design problem with a different strategy that can efficiently explore high-dimensional parameter spaces for self-folding unit design.

To solve the inverse problem, we propose using machine learning methods trained on simulation results to classify regions of true and false foldability in the design space of the non-dimensional groups. With the design space classified, we can for any target foldable behavior provide the set of the non-dimensional parameters which lead to that behavior. Using the practical constraints, we translate the non-dimensional parameters into material properties, geometrical dimensions, and required stimuli-driven dimensional changes that lead to the desired folded self-folding unit. The practical constraints encompass material property bounds reflecting available stimuli-responsive materials, geometric limitations imposed by manufacturing processes, and self-folding unit level requirements such as total weight or volume restrictions.

Solving the inverse problem helps support future manufacture of self-folded units by guiding the material selection, geometrical design, and stimuli-responsive behavior that leads to target self-folding behavior. Outlining the specific application, it can help determine the values of some of these parameters when other parameters are already selected, i.e determining the geometrical dimensions when the bilayer material and active layer changes are known.

Integrated computational framework

We develop the computational framework that enables both forward and inverse design capabilities through integration of three research components, each addressing different aspects of the overall design challenge (Figure 4.12).

1. **Simulations**: We employ continuum mechanics modeling to provide the fundamental physics-based foundation for predicting self-folding unit behavior from first principles, incorporating detailed treatment of material constitutive relations, large deformation kinematics, and complex boundary conditions. We ensure this mechanistic approach

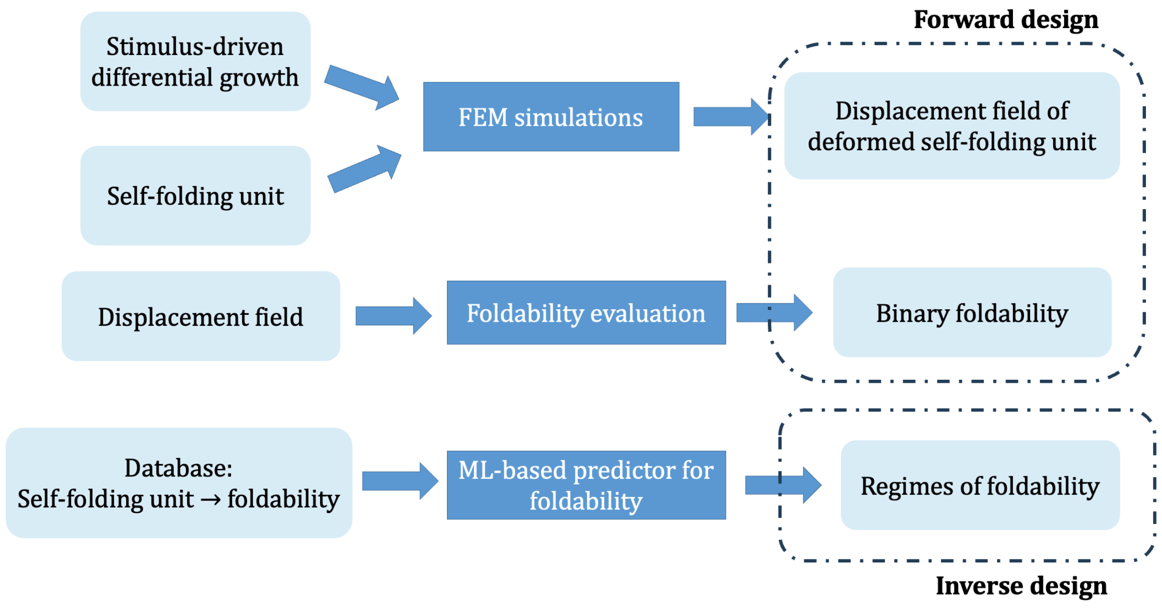

Forward design
Stimulus-driven differential growth
Self-folding unit
FEM simulations
Displacement field of deformed self-folding unit
Displacement field
Foldability evaluation
Binary foldability
Database: Self-folding unit → foldability
ML-based predictor for foldability
Regimes of foldability
Inverse design

Figure 4.12: Overview of the computational framework, with 3 components of FEM simulations for deformation of self-folding units, foldability evaluation algorithms, and ML-based predictors to divide the design space into regimes of foldability

maintains physical realism that enables accurate prediction of folding performance across diverse parameter ranges, but it requires significant computational resources for comprehensive parameter space exploration.

2. **Foldability evaluation**: We implement automated geometric analysis algorithms to objectively evaluate folding outcomes and classify foldability success from 3D simulation results. We design these algorithms to translate complex three-dimensional deformation fields into differential geometry metrics that can be systematically compared across different design configurations. The aim is to generate comprehensive design databases that map parameter combinations to folding outcomes, providing the foundation for statistical analysis and machine learning model development.

3. **ML predictor**: We explore machine learning approaches trained on mechanistic model results to discover relationships between design parameters and self-folding unit performance. These data-driven methods can enable real-time design synthesis and

potentially identify complex parameter interactions that drive self-folding.

We create through the combination of these three research components a comprehensive computational framework capable of addressing both forward and inverse design challenges. In the following sections, we establish this framework as the foundation for transitioning self-folding unit from design to practical engineering applications operating under terrestrial conditions.

4.3 Simulating Morphing of Self-Folding Units

The comprehensive analysis of self-folding unit behavior under gravity loading requires sophisticated computational methods capable of capturing large deformation kinematics, multi-material dimension changes, and complex contact phenomena. We develop a simulation using the workflow in Section 2 that enables systematic prediction of stimuli-induced deformation in self-folding units across diverse parameter ranges. This provides the foundation for both forward design analysis and the inverse design strategies. This computational capability represents the first of our three complementary research components, establishing the physics-based foundation upon which algorithms for geometric analysis and machine learning approaches subsequently build.

4.3.1 Computational Domain Setup

The accurate prediction of self-folding unit behavior necessitates meticulous specification of the computational domain encompassing geometric modeling, material property assignment and mesh generation strategies to ensure that numerical simulations faithfully represent the physical system under investigation. Based on the foundations covered in Section 2.1, we describe the construction of the computational domain for a self-folding unit. We start with constructing a 3D geometry which is later meshed in 3D and separate domains are created to represent different material properties and behavior.

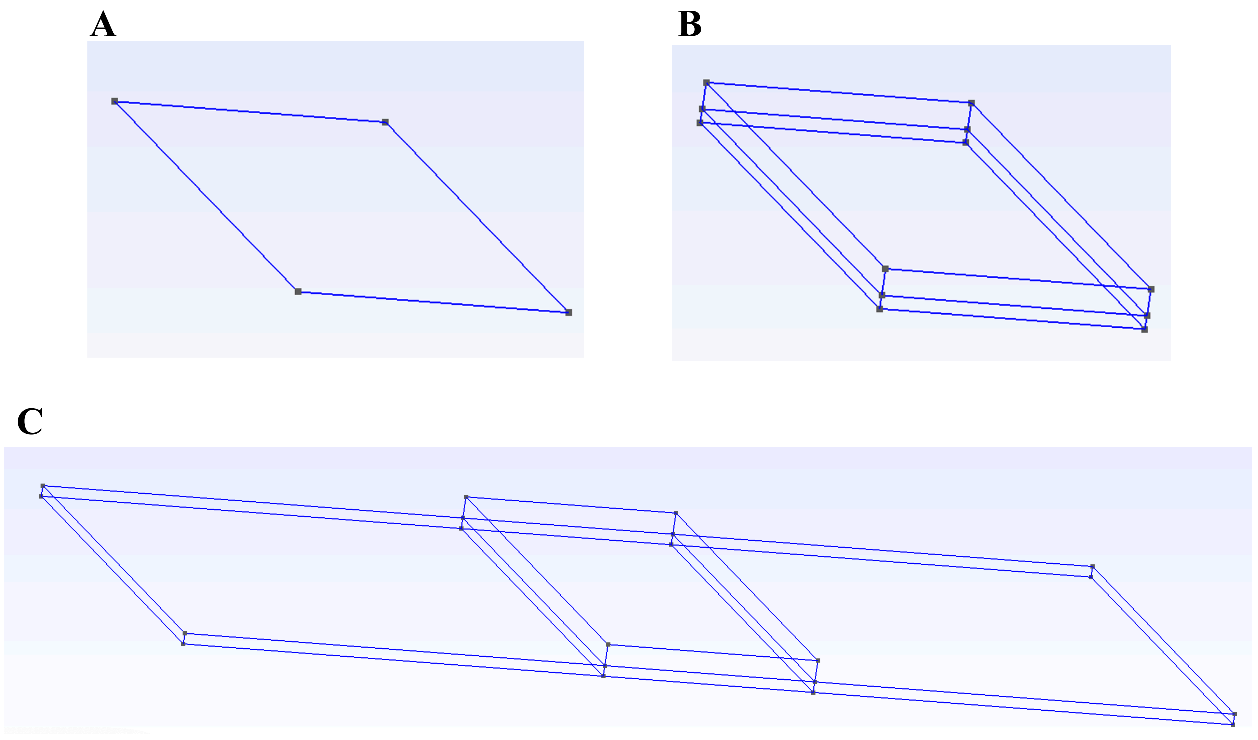


Figure 4.13: Constructing the domain geometry of self-folding unit with material contact between bilayers in the hinge and between panel and hinge. **(A)** Rectangular contact surface of the bilayer. **(B)** Distinct volumes extruded from bilayer contact surface for active and passive layer. **(C)** Distinct volumes for panels extruded on either side from the boundary of the passive layer volume.

Domain geometry

We construct the 3D geometric model of the self-folding unit through a systematic extrusion-based approach (Figure 4.13). This begins with defining the contact interface between the active and passive layers as a rectangular surface of dimensions $l_i \times b_i$. From this rectangular interface, we generate the bilayer hinge geometry by extruding the contact surface outward in both perpendicular directions by distances h_s and h_i respectively, creating the passive and active layer volumes that comprise the bilayer assembly. We identify the two lateral faces of the passive layer prism that are oriented perpendicular to the hinge length direction, and extrude these surfaces outward by distance l_s to form the panel volumes. The resulting panel geometries possess dimensions $l_s \times b_s \times h_s$, where the panel thickness h_s directly inherits the passive layer thickness, the panel width b_s typically equals the hinge width b_i for geometric consistency, and the panel length l_s is determined by design requirements for folding ratio and structural performance.

Contact interfaces

We choose to construct the geometry by extruding the bilayer contact surface to ensure that the active and passive layers share a perfectly conformal interface without geometric discontinuities or gaps that could compromise force transmission or introduce numerical instabilities in the computational analysis. Our geometric construction also ensures that the panels represent direct continuations of the passive layer material, maintaining perfect structural continuity and eliminating the need for discrete interface modeling between hinge and panel components for force or moment transfer.

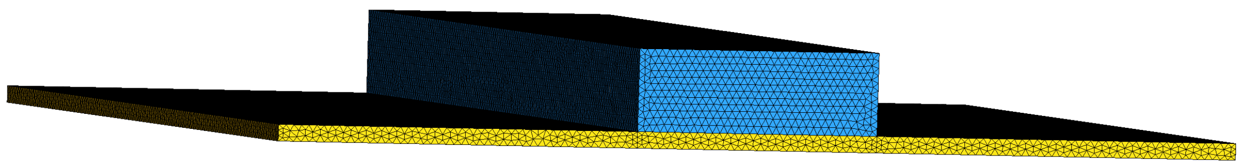
Meshing

Our mesh generation strategies implements spatial discretization techniques to capture multi-material properties. The discretization strategy must balance computational efficiency with accuracy requirements, ensuring adequate resolution of stress gradients near material interfaces while maintaining reasonable computational costs for simulations which scale with number of vertices. We use *Gmsh* to generate 3D unstructured meshes, as described in 2.1, to divide all the volume components of the self-folding unit into tetrahedral elements.

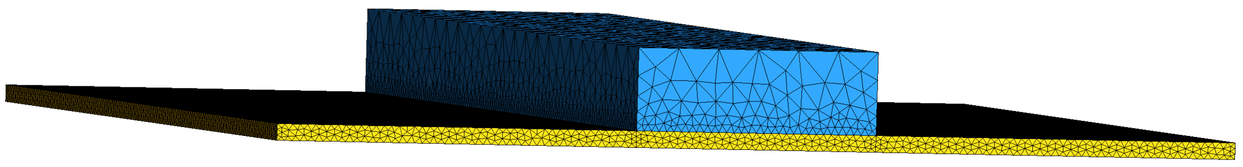
From our construction of the self-folding unit geometry, the contact surface which is defined first is extruded to form the bilayer hinge. Following this construction the volumes of the active and passive layer in the bilayer hinge have a shared volume boundary of the contact surface which is a 2D rectangle. In the process of meshing the volume, the rectangular contact surface is meshed first with points,line edges, and 2D triangular elements. Consequently, the meshes of the active and passive layers share mesh elements, excluding volume elements, at this contact surface establishing perfect contact since the material coordinates at the contact surface always remain common. On the other hand, since only 1D and 2D elements are common at the contact surface, we can easily define the different material properties with markers associated to each 3D volume element because they never overlap. Thus, by geometric construction we ensure the resulting 3D mesh has perfect contact by sharing 2D elements on the contact surface and can allocate different properties to the mesh elements that exactly follows material boundaries.

As elaborated in Section 2.1.3, in consideration of continuum assumptions we are required to always have the element size of the mesh lesser than smallest dimension. For the self-folding unit, the smallest dimension is the thickness, which changes across the geometry. Furthermore, since the volume components of the active and passive layer are separate with the shared boundary of the contact surface , they are meshed in 3D separately. This makes the smallest dimension of the meshed volume the local material thickness. So, to ensure sufficient refinement and reduce the number of nodes we set the mesh size as a function of the local material thickness $c \times h$, where c is a specified constant and h is the local material thickness. The functional dependence makes the element size adapt at different locations within the mesh to reduce computational load by reducing the number of mesh nodes and volume elements (Figure 4.14).

A



B



	r_{i}	r_{s}	Nodes	Elements
Mesh A	0.008	0.008	80870	463133
Mesh B	0.042	0.008	52832	295037

Figure 4.14: Meshes for the self-folding unit. For each mesh, the table contains: the maximum element dimension(relative to self-folding unit length) within each material, r_{i} for the active layer and r_{s} for the passive layer, the number of nodes(vertices) and volume elements in the mesh. **(A)** Mesh generated for the self-folding unit with a uniform mesh element size that satisfies continuity considerations. **(B)** Mesh generated for the self-folding unit with a adaptive mesh element size that satisfies continuity considerations. Here, the maximum element size is set to be $0.49h$, where h is the local material thickness.

4.3.2 Modeling Deformation of Self-Folding Units

We model deformation of the constructed mesh to capture the essential physics governing morphing behavior of self-folding units, integrating large deformation kinematics with stimuli-responsive material behavior and weight effects. Through deforming the mesh we are able to predict the changing shape of any self-folding unit driven be internal growth or contractions that are triggered using stimuli.

We follow the methods in Section 2.2 to use energy minimization to solve for deformation fields. The total potential energy functional below will be minimized to find the displacement field.

$$\Pi(\boldsymbol{u}, \boldsymbol{\lambda}) = \int_{\Omega} \Psi(\mathbf{F}_{\mathrm{e}}(\boldsymbol{u}, \mathbf{F}_{\mathrm{g}}), \boldsymbol{X})\, dV - W_{\text{ext}}(\boldsymbol{u}) - W_{\mathrm{c}}(\boldsymbol{u}) \tag{4.1}$$

We include the following components while formulating the total potential energy functional:

- **Gravity**: The presence of gravity is incorporated as potential energy from a body force distributed through the volume of the self-folding unit using the local density with the following term:

$$\int_{\Omega} \rho(\mathbf{X})\mathbf{g} \cdot \mathbf{x}\, \mathrm{d}V = \int_{\Omega} \rho(\mathbf{X}) g z\, \mathrm{d}V \tag{4.2}$$

  where the terms are non-dimensional and ρ is the local density, $\boldsymbol{g} = g\hat{\boldsymbol{g}} = g\hat{\boldsymbol{e}}_{\boldsymbol{z}}$ the acceleration due to gravity, and $z = \mathbf{x} \cdot \hat{\boldsymbol{e}}_{\boldsymbol{z}}$ the component of current position along the direction of gravity.

- **Elastic energy**: We model the active layer, passive layer, and panels as Neo-hookean elastic solids. So the elastic energy density for the self-folding unit is given by:

$$\Psi(\mathbf{F}_{\mathrm{e}}, \boldsymbol{X}) = \frac{\mu(\boldsymbol{X})}{2}(I_1 - 3 - 2\ln J_{\mathrm{e}}) + \frac{\lambda(\boldsymbol{X})}{2}(\ln J_{\mathrm{e}})^2 \tag{4.3}$$

  where $I_1 = \mathrm{tr}(\mathbf{F}_{\mathrm{e}}^T \mathbf{F}_{\mathrm{e}})$ is the first invariant of the elastic right Cauchy-Green deformation tensor, and $J_\mathrm{e} = \det(\mathbf{F}_\mathrm{e})$ quantifies volumetric changes in the elastic deformation. Lame constants, $\mu(\boldsymbol{X})$ and $\lambda(\boldsymbol{X})$, are determined from the Young's modulus, $E(\boldsymbol{X})$, and Poisson ratio, $\nu(\boldsymbol{X})$. These properties are a function of position $\boldsymbol{X}$ in material coordinates because they vary in the domain of the self-folding unit, which is a composite of multiple materials.

- **Growth**: For our simulations of the self-folding unit, we say the active layer undergoes contraction and the passive layer with the panels remains the same. The simulations by default have isotropic contraction but we have also included the option for anisotropic contraction which can be chosen. The growth tensor for active and passive layers usually looks like:

$$\mathbf{F}_{\mathrm{g,i}} = (1+g)\mathbf{I}, \quad \mathbf{F}_{\mathrm{g,s}} = \mathbf{I} \tag{4.4}$$

  where $g < 0$ is the growth strain from contraction, $\mathbf{I}$ is the identity tensor in 3D, $\mathbf{F}_{\mathrm{g,i}}$ and $\mathbf{F}_{\mathrm{g,s}}$ is the growth tensor in the active and passive layers.

- **Ground constraint**: The presence of the ground during deformation is included in the total potential energy by creating a term called the ground barrier potential energy. We first define the ground to be at $z = 0$ and $z > 0$ as above ground. The ground barrier potential energy drastically penalizes the presence of any mesh point below the ground, thereby having the effect of supporting the self-folding structure on the ground:

$$\int_\Omega b e^{-az} \mathrm{d}V \tag{4.5}$$

  where $z = \boldsymbol{x} \cdot \hat{\boldsymbol{e}}_{\boldsymbol{z}}$ is the z-coordinate of the current position of the self-folding units, $b > 0$ and $a > 0$ are constants that are tuned to make the ground barrier potential energy large enough to prevent any $z < 0$ without making the other terms in the energy formulation negligible.

- **Rigid body motion**: We constrain rigid body translation along all the three axes and rotation along the normal to the plane of the initial flat configuration. Using Lagrange multipliers to implement these constraints, we subtract the following terms from the total energy functional:

$$\lambda_{\text{rot, z}} C^{\text{fixed}}_{\text{rot,z}}(\boldsymbol{u}) + \sum_i \lambda_{\text{trans, i}} C_{\text{trans, i}}(\boldsymbol{u}), \quad \text{i} \in \{x, y, z\} \tag{4.6}$$

$$C_{\text{trans, i}}(\boldsymbol{u}) = \int_\Omega \boldsymbol{u}(\boldsymbol{X}) \cdot \boldsymbol{e}_i \, \text{d}V, \quad C^{\text{fixed}}_{\text{rot,z}}(\boldsymbol{u}) = \int_\Omega \boldsymbol{X} \times \boldsymbol{u} \cdot \boldsymbol{e}_z \, \text{d}V \tag{4.7}$$

where $\lambda_{\text{trans, i}}(\boldsymbol{u})$ and $\lambda_{\text{rot,z}}(\boldsymbol{u})$ are the Lagrange multipliers to constrain translations and rotation, $\boldsymbol{u}(\boldsymbol{X})$ is the displacement field, $\boldsymbol{X}$ is the reference material coordinates, and $\boldsymbol{e}_i$ is the unit vector along each coordinate axis.

Combining all the components, we formulate the total potential energy functional to be minimized as follows:

$$\Pi(\boldsymbol{u}, \boldsymbol{\lambda}) = \int_\Omega \psi(\boldsymbol{u}, \boldsymbol{X}) \, dV + \int_\Omega \rho(\mathbf{X}) g z \, \text{d}V + \int_\Omega b e^{-az} \text{d}V - \lambda_{\text{rot, z}} C^{\text{fixed}}_{\text{rot,z}}(\boldsymbol{u}) - \sum_{i=1}^{3} \lambda_{\text{trans, i}} C_{\text{trans, i}}(\boldsymbol{u}) \tag{4.8}$$

4.3.3 Implementing simulations

We implement the computational framework through a simulation approach that gradually applies stimuli-induced contractions. This section details the numerical implementation, including the evolution scheme for growth parameters, solver selection, and parameter tuning strategies required for robust simulation of large-deformation self-folding behavior.

Evolving equilibrium

Using the approach of quasi-static analysis, we implement solving for energy-minimization where stimuli-induced dimensional changes are gradually applied through a series of incremental steps, allowing the self-folding unit to reach mechanical equilibrium at each stage of the folding process.

The stimuli-induced dimensional changes are captured by the growth strain in the active layer:

$$\mathbf{F}_{\text{g,i}} = (1+g)\mathbf{I}, \quad g < 0 \tag{4.9}$$

where to capture incremental changes we set g using an evolution parameter $e : 0 \to 1$ that represents the progression of simulation:

$$g = g_{\text{init}} + e \cdot (g_{\text{final}} - g_{\text{init}}) \tag{4.10}$$

where g_{init} and g_{final} are the prescribed initial and final growth strains in the active layer induced by stimuli. We set $g_{\text{init}} = 0$ for our self-folding simulations because we want to identify the ability to fold starting from a flat reference state with no initial strain. We also typically use $g_{\text{final}} = -0.40$ to drive the dimensional change of the active layer.

Choice of solver

Initially, we used the Newton's solver approach as detailed in Section 2.4.2 to solve for the displacement field at equilibrium by minimizing Equation (4.8). The convergence failed before $e = 1$ for many self-folding units that were tried even after tuning any simulation related parameter including making the increments of e as small as $\Delta e = 10^{-5}$. We observed behavior of large deformations including warping and buckling during the shape change of the self-folding units. So, we decided to construct a solver based on Arc-length method as detailed in Section 2.4.3 to solve for the entire solution trajectory $e : 0 \to 1$ through regions of sudden shape change and instability. After implementing the arc-length method to solve for equilibrium, we were able to trace the entire solution trajectory $e : 0 \to 1$. So, we chose to use the Arc-length method as the solver strategy for all simulations of self-folding.

Parameter tuning and perturbations

While implementing the simulations for the self-folding units, a lot of fine-tuning of numerical quantities was done and we detail here how some important numerical quantities were tuned and their effects.

- **Arc length parameters**: The arc-length solver (Section 2.4.3) which minimizes the energy functional from Equation 4.8 for the next equilibrium point on the trajectory below:

$$f_{\text{arc}}(\boldsymbol{v}, e) = \frac{\|\boldsymbol{v} - \boldsymbol{v}^{\text{converged}}\|^2}{a_{\text{arc}}^2} + \frac{(e - e_{\text{converged}})^2}{b_{\text{arc}}^2} - 1 = 0 \tag{4.11}$$

  where $b_{\text{arc}} = 0.01$ and $a_{\text{arc}} = ||\boldsymbol{v}(e = e^1) - \boldsymbol{v}^{\text{converged}}(e = 0)||$ is initialized after using the Newton solver to solve for the displacements after the first evolution step $\boldsymbol{v}(e = e^1), e^1 = 0.01$. The values for a_{arc} and b_{arc} are adapted through the evolution steps using the all the enhancements described in Section 2.4.3 for optimizing simulation runtime.

- **Ground constraint**: As described in Section 4.3.2,the constraint to hold the self-folding structure on the ground during deformation is incorporated within the energy functional as a barrier penalty term as:

$$\int_{\Omega} b e^{-az(\boldsymbol{X})} \mathrm{d}V \tag{4.12}$$

  where $z(\boldsymbol{X})$ is the position coordinate along the z axis, and this restricts the presence of any points with $z(\boldsymbol{X}) < 0$ as we consider the plane $z = 0$ as "ground". Here, $b > 0$ and $a > 0$ are tuned so that the penalty term is high enough for it to be an effective constraint, but also not too high which would render the remaining terms in the energy functional unimportant in the energy minimization. Over multiple iterations of changing a, b and observing the effectiveness of penalty term as well as the simulations retaining all the mechanical behavior, we choose $b = 0.001, a = 1000$ as optimal values for our range of self-folding unit simulations.

- **Perturbations for stability**: We add a noise function to the density and displacement

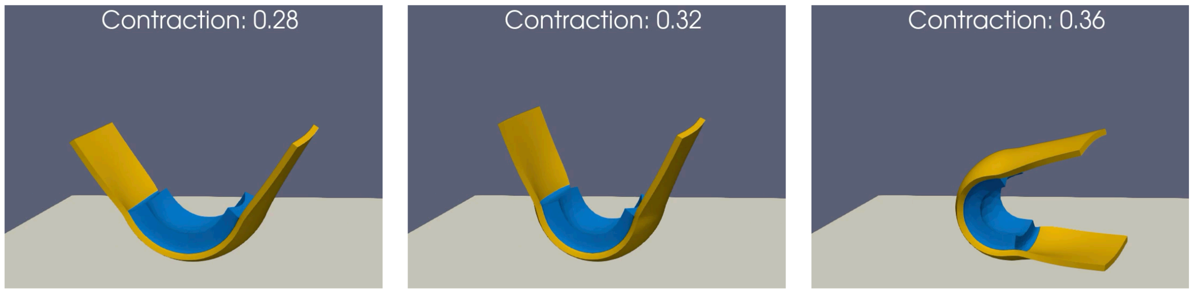


Figure 4.15: The self-folding unit deforming as contraction of the active layer(blue), g_i, increases in magnitude, left to right. The intermediate equilibrium state achieved at $g_i = -0.32$ maybe unstable and by adding perturbation in displacement before evolving the next steps we see that the structure topples and consequent equilibrium states are stable.

functions with noise amplitude of $\epsilon \in (10^{-3}, 10^{-2})$ relative to the perturbed function. This mimics realism by breaking the symmetry in the simulation, perturbing the structure, and capturing not perfectly uniform material density. The perturbations to the self-folding unit, added in each evolution step, ensure that the final shape achieved is stable, because an unstable equilibrium state would change if the perturbation is added and only a stable structure would persist (Figure 4.15).

Mesh sensitivity

An important parameter affecting simulation accuracy is the mesh element size, set as $c \times h$ where $c > 0$ is a constant and h is the local material thickness (Section 4.3.1). To balance accuracy with computational efficiency, we conduct convergence analysis by systematically reducing c from an initial value $c_0 = 1$. For each value $c_n = c_{n-1}/2$, we run identical simulations and compare final solution vectors until:

$$||\boldsymbol{v}^{\text{converged},n} - \boldsymbol{v}^{\text{converged},n-1}|| < \varepsilon, \quad \text{where } \varepsilon = 10^{-10} \tag{4.13}$$

Based on this convergence analysis across multiple self-folding unit geometries, we set $c = 0.48$ for all simulations, ensuring adequate resolution while maintaining reasonable computational cost.

We have established a computational framework for simulating self-folding unit behavior that integrates large deformation mechanics, stimuli-responsive materials, and gravity. The implementation using arc-length continuation methods enables robust tracking of folding trajectories through instabilities, while adaptive meshing ensures computational efficiency. This simulation capability provides the physics-based foundation for generating the datasets needed to develop both geometric analysis algorithms and machine learning models for self-folding design.

4.4 Simulation Results and Validation

4.4.1 Validating Simulation Results Without Gravity

We validate our finite element simulation framework against established analytical solutions for bilayer bending behavior, ensuring that our computational approach accurately captures the physics governing self-folding unit deformation before incorporating gravity.

Analytical benchmark from Timoshenko bilayer theory

The classical Timoshenko analysis of bilayer strips provides exact analytical solutions for curvature development under differential thermal strains, serving as the theoretical foundation for validation of our simulation framework [69]. For a bilayer comprising active and passive layers with thicknesses h_i and h_s, elastic moduli E_i and E_s, and differential strain g_i in the active layer, the developed curvature expressed with our non-dimensionalized parameters is as follows:

$$\kappa = \frac{6g_i(1 + h_i/h_s)}{h_s\left[3(1 + h_i/h_s)^2 + (1 + h_i/h_s \cdot E_i/E_s)(h_i^2/h_s^2 + 1/(h_i/h_s \cdot E_i/E_s))\right]} \tag{4.14}$$

This analytical relation assumes small strain deformation, perfect bonding between layers, and uniaxial strain along bilayer length, providing exact predictions for curvature as functions of material property ratios and geometric parameters.

For analytical prediction of folding angle, we note that curvature is negligible along the panel regions so we integrate the curvature over the hinge length to determine the total accumulated folding angle:

$$\theta_{\text{analytical}} = \int_0^{l_i} \kappa \, ds = \kappa \cdot l_i \tag{4.15}$$

where the curvature κ from Equation (4.14) remains constant along the hinge length under the assumptions of uniform material properties and strain distribution [69].

From Equations (4.14) and (4.15), we note that the folding angle between the panels, and consequently foldability, depends only on 3 non-dimensional parameters of the self-folding unit $\{E_i/E_s, h_i/h_s, h_s/l_i\}$. We also see that the folding angle is independent of the panel lengths, implying that in the absence of the weight of panels the bilayer bending determines the folding of perfectly attached panels.

Gravity-free simulations

We modify our simulation framework by reducing the system to pure elastic deformation driven by stimuli-induced dimensional changes. The simplified energy functional from Equation (4.8) becomes:

$$\Pi(\boldsymbol{u}, \boldsymbol{\lambda}) = \int_\Omega \Psi(\mathbf{F}_{\text{e}}(\boldsymbol{u}, \mathbf{F}_{\text{g}}), \boldsymbol{X}) \, dV - \lambda_{\text{rot, z}} C^{\text{fixed}}_{\text{rot,z}}(\boldsymbol{u}) - \sum_{i=1}^{3} \lambda_{\text{trans, i}} C_{\text{trans, i}}(\boldsymbol{u}) \tag{4.16}$$

This formulation eliminates both potential energy due to gravity and ground barrier terms while retaining the elastic energy density and rigid body motion constraints. The removal of gravity effects lets us directly compare results with analytical bilayer theory predictions.

Anisotropic contraction for validation

While our self-folding unit simulations employ isotropic contraction in the active layer by default, validation against Timoshenko theory requires anisotropic dimensional changes that match the theoretical assumptions of linear expansion along length of bilayer neglecting changes in the width. We implement uniaxial contraction aligned with the folding direction (here y-axis) by specifying the growth tensor as:

$$\mathbf{F}_{\text{g,i}} = \text{diag}(1, 1 + g_{\text{i}}, 1), \quad \mathbf{F}_{\text{g,s}} = \mathbf{I} \tag{4.17}$$

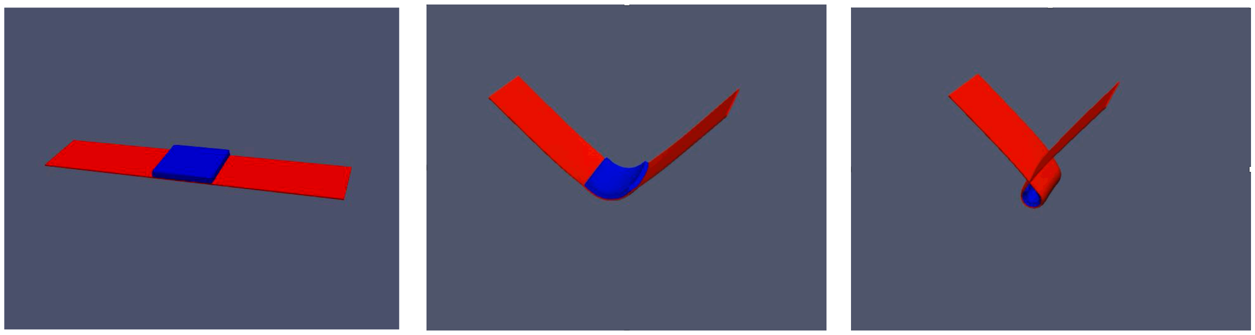
where $\mathbf{F}_{\text{g,i}}$ and $\mathbf{F}_{\text{g,s}}$ are the growth tensors in the active and passive layers, $g_{\text{i}} < 0$ represents contraction strain in the folding direction while maintaining zero strain in the transverse directions. This anisotropic growth tensor ensures that the simulation conditions match the analytical validation benchmark.

Comparative analysis of folding angles

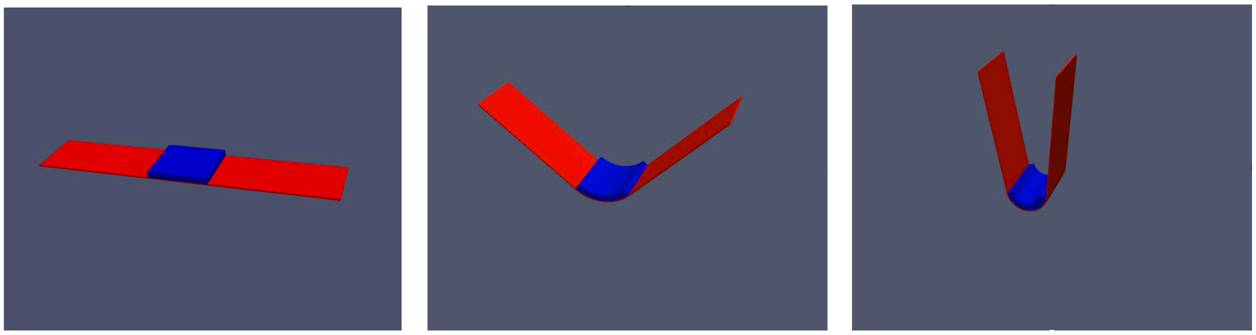
We assess simulation accuracy by comparing folding angles measured from finite element simulation results against angles predicted through integration of analytical curvature expressions. From simulation results, we extract folding angles by measuring the relative orientation between the two panels in their final deformed configurations. The simulation-based folding angle θ_{sim} quantifies the total geometric change achieved through the bilayer hinge deformation. For a prescribed set of parameters we simulate the self-folding unit in the absence of gravity and plot θ_{sim} for different contraction strains g_i (Figure 4.16).

We conduct systematic validation across multiple parameter combinations spanning realistic ranges of thickness ratios h_i/h_s, modulus ratios h_i/h_s, and contraction strains g_{i} following the example shown. From the plot we obtain $\theta_{\text{sim}} = 3$ as the value of the angle accumulated using the values of the corresponding parameter combination in Table 4.1.
The value of $\theta_{\text{analytical}} = 3.63$ is calculated from plugging in the parameter values in Equations (4.14) and (4.15). The validation process demonstrates agreement between θ_{sim} and $\theta_{\text{analytical}}$ within acceptable deviations due to the presence of large deformations, which are treated with

A



B



C

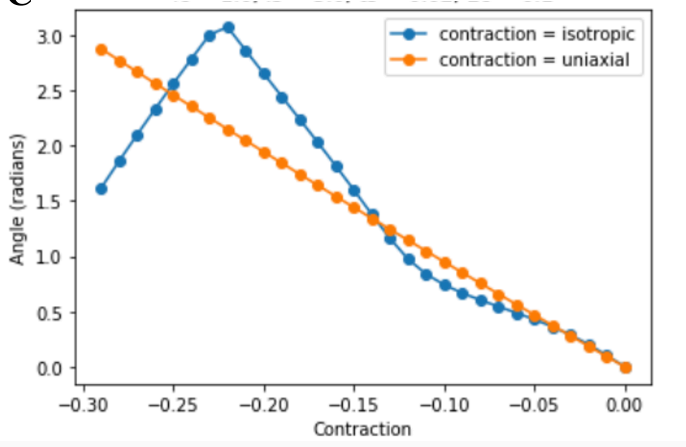


Figure 4.16: Simulation results of self-folding unit in the absence of gravity, generated with non-dimensional parameters in Table 4.1. The active layer(subscript "i") is blue and passive layer is red(subscript "s"). **(A)** Active layer contracting isotropically, frames are for $g_\mathrm{i} = \{0, 0.15, 0.3\}$. **(B)** Active layer contracting uniaxially along folding direction, frames are for $g_\mathrm{i} = \{0, 0.15, 0.3\}$. **(C)** Plot showing folding angle between the two panels as a function of active layer contraction g_i. For isotropic contraction, decreasing angles with contraction are a numerical result of the panels overlapping after they fully close.

$E_{\mathrm{i}}/E_{\mathrm{s}}$	$h_{\mathrm{i}}/h_{\mathrm{s}}$	$l_{\mathrm{i}}/l_{\mathrm{s}}$	$b_{\mathrm{s}}/l_{\mathrm{s}}$	$h_{\mathrm{s}}/l_{\mathrm{s}}$	g_{i}
10^{-1}	5	0.2	0.2	4×10^{-3}	0.3

Table 4.1: Values of non-dimensional parameters. h_i/h_s is the relative thickness, E_i/E_s is the relative stiffness, h_s/l_s is non-dimensionalized passive layer thickness, b_s/l_s is the aspect ratio of the passive sheet, l_i/l_s is the ratio of bilayer length to total length, and g_{i} is the active contractile strain.

finite strain theory in the simulations and using linear elasticity in the analytical relation. The validation process was similarly repeated across different results in the tested parameter space, establishing confidence in our computational framework for subsequent investigations with gravity.

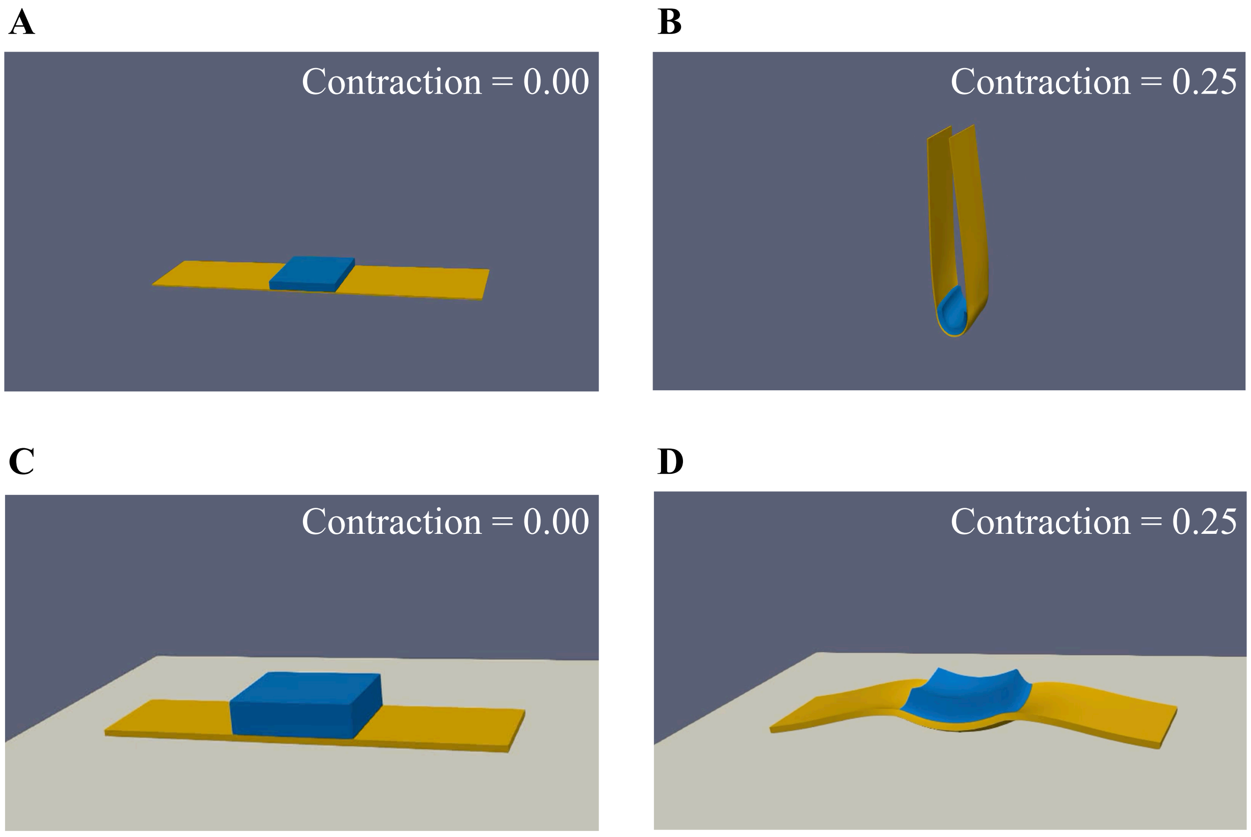
4.4.2 Simulation results with gravity

Effect of adding gravity

We run simulations of self-folding units in the presence of gravity by minimizing the energy functional in Equation (4.8) and visualize the deformation from increasing contraction of the bilayer. We can see the effect of gravity from comparing two simulation results with and without gravity, where the common non-dimensional parameters are identical and the contraction is isotropic (Figure 4.17). In the simulation without gravity, the self-folding unit deforms through active layer contraction causing bending of the hinge that leads to overall folding of the panels. Whereas with gravity, the active layer contraction causes bending in the hinge but there is no folding of the panels. We also increase the active contraction in the simulation with gravity beyond that of the corresponding result without gravity, but this also led to no folding. Since the presence of gravity is what has been added, the bending of the hinge was not strong enough to lift the panels to be folded against their weight.

Additional Design Parameters

With the addition of gravity in the simulations, we see that self-folding units that folded without gravity have failed to fold. This motivated us to explore additional non-dimensional parameters,



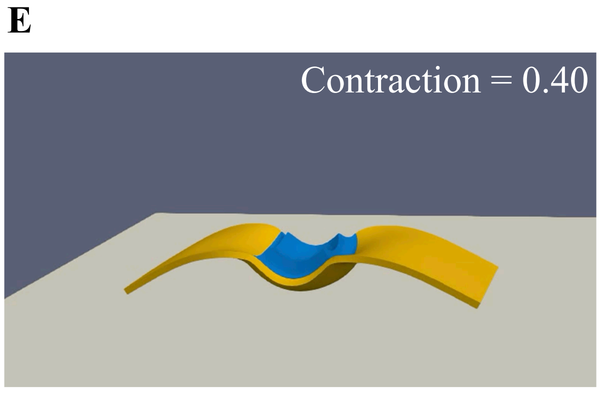


Figure 4.17: States of different self-folding units with identical non-dimensional parameters for the bilayer. **(A, B)** Initial and final states of self-folding unit folding without gravity, panels are fully folded at active contraction of 0.25. **(C)** Initial state of self-folding unit under gravity resting on ground (light gray). **(D)** Deformed state of self-folding unit from (C), panels don't fold at active contraction of 0.25 which corresponds to state in (B) without gravity. **(E)** Deformed self-folding unit under gravity with increased active contraction of 0.4.

not included in the analysis without gravity, which are now important in predicting folding behavior and consequently to design for foldability. To explore successful folding behavior, we adjust the non-dimensional parameter $(E_s h_s^2)/(\rho_s g l_s^3)$, which has now been introduced with the addition of gravity. We had identified this parameter as the fundamental competition in the panels between remaining straight by resisting bending and bending under their weight. We see that increasing the value of this parameter in the simulation results in translating the hinge bending to overall folding as the panels are able to lift off (Figure 4.18 A-B).

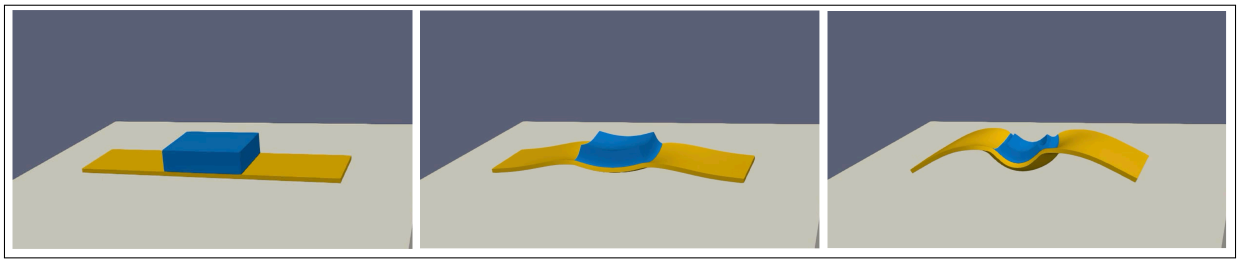
We were also curious to explore the effects of the overall aspect ratio of the self-folding unit, which is not addressed in any way in the analytical equations without gravity. For a self-folding unit that can be successfully folded (Figure 4.18 B), increasing the aspect ratio can result in losing foldability because the direction of bending changes leading to no folding in the expected folding direction (Figure 4.18 C).

We have successfully built and validated the simulations for deforming self-folding units under differential contractions. During validation we removed gravity and made contraction uniaxial to compare with results from existing theoretical framework. We explored the effects adding gravity to the simulation and the importance of 2 non-dimensional groups that get added to the system physics in influencing foldability.

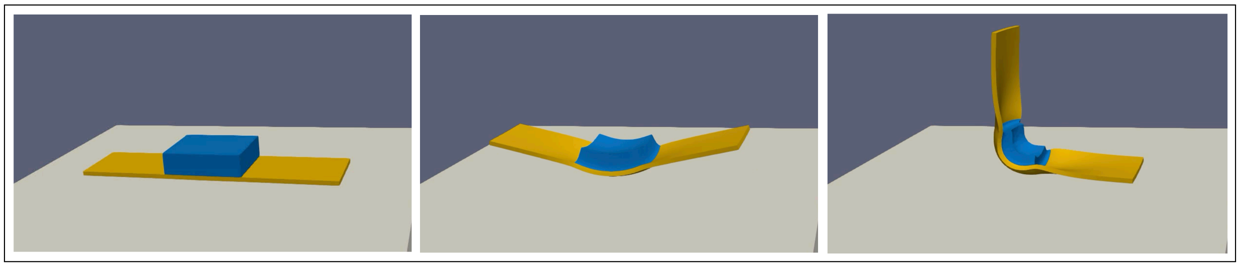
4.4.3 Exploration of design space

Previously, we performed some preliminary comparisons between foldability outcomes of self-folding units when gravity is added and some key non-dimensional parameters are changed. We observed the foldability outcomes by visualizing the simulation results in *ParaView* [49] by feeding it the 3D deformation field from each simulation. For more exhaustive analysis across hundreds of simulations when changing multiple parameters, we require an automated way to extract foldability outcomes which is implemented as a foldability evaluation algorithm discussed

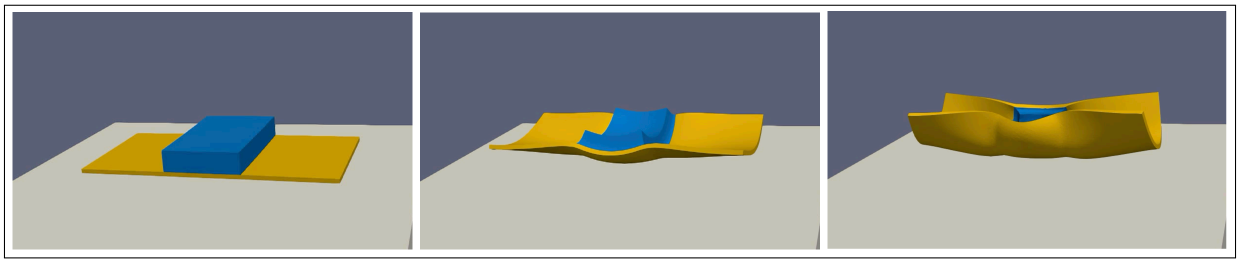
A



B



C



Self-folding unit	$E_s h_s^2/\rho_s g l_s^3$	b_s/l_s
A	0.04	0.25
B	0.4	0.25
C	0.4	0.5

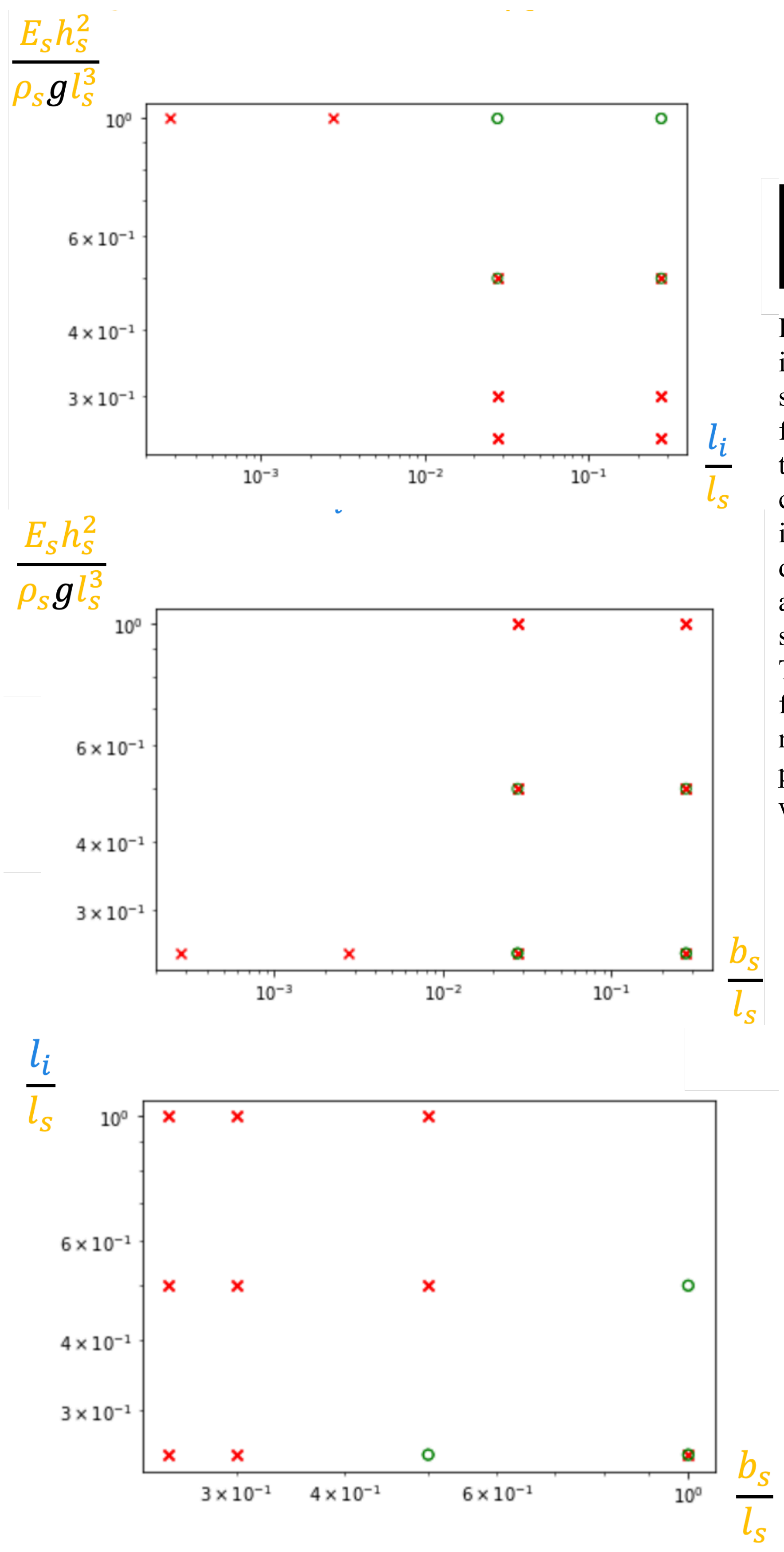
Figure 4.18: Shapes of different self-folding units deforming under gravity and active contractions with magnitude of 0.00(left), 0.20(middle), and 0.40(right). The parameters differentiating the self-folding units are given in the table above and the remaining non-dimensional parameters of the self-folding units are identical.

in Section 4.5. This helps us assign a binary for foldability success based on running an analysis on simulation results. Using a combined workflow of the simulation and the foldability evaluation algorithm, we were able to run batches of hundreds of simulations with varying non-dimensional parameters and understand the foldability outcome for each combination of non-dimensional parameters. We captured the results across 900+ simulations in a *pandas* database for further analysis in python, formatted as shown in Table 4.2.

Index	E_i/E_s	ρ_i/ρ_s	l_i/l_s	b_i/b_s	h_i/h_s	$E_s h_s^2/\rho_s g l_s^3$	b_s/l_s	h_s/l_s	g_i	F
1	0.1	0.1	0.25	1	0.2	0.4	0.2	0.004	0.4	*True*
2	...	...	...	...	...	...	...	...	...	*True*
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 4.2: Dataset of simulation results created as a *pandas* database by collecting results of individual simulations of the self-folding unit with gravity and active layer contraction g_i, each specified by its non-dimensional parameters and corresponding foldability outcome from the simulation.

From the simulation results, we plot foldability outcomes as a function of different non-dimensional groups to explore the influence of different design parameters on foldability. From the simulations without gravity which are validated with the analytical relations in Section 4.4.1, we know the general trends of the effect of relative thickness h_i/h_s and relative stiffness E_i/E_s on folding behavior which is that folding increases with increase in both these parameters. So, to understand the effect of other important parameters when gravity is added, we plot the foldability outcomes of simulations with the 2 non-dimensional groups on the axes. These plots help us understand the effect of each of the 2 non-dimensional parameters (Figure 4.19). In our plots we understand that all of the parameters in the design space are important to the foldability outcomes because the outcomes observed are not independent to any of the non-dimensional parameter. We can also see the possible existence of regimes of foldability because successful foldability outcomes exist exclusively in select regions of the plots. We did not generate simulation results to manually explore the effect of parameters ρ_i/ρ_s and h_s/l_s on foldability outcomes because the density of active layer ρ_i would likely have lesser effect on the potential



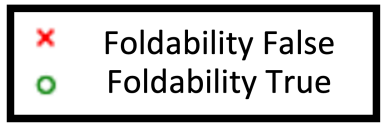


Figure 4.19: Plots showing foldability outcomes from simulations of different self-folding units. The simulation results are plotted by locating the point corresponding to the value of the non-dimensional parameters used and using a marker to represent the foldability outcome. The 3 plots visualize the effect of 3 non-dimensional parameters on foldability, these parameters become important when gravity is added.

energy, with the active layer always being closer to the ground surface, and the effect of h_s/l_s might be captured within the non-dimensional parameter, $(E_s h_s^2)/(\rho_s g l_s^3)$, that is already manually explored.

We understand that given the influence of at least 5 non-dimensional parameters on the foldability outcomes, exploring design space to identify regimes of foldability through plotting becomes ineffective, motivating us to seek more sophisticated ways for finding regimes of foldability from simulation results.

4.5 Automated foldability assessment

Our simulations take a self-folding unit defined by a set of material and geometric properties as input and generate the final deformed state under stimuli. The result of the simulations is a 3D displacement field on a 3D mesh which is visualized. Based on the final deformed shapes we may visually determine if the self-folding unit has deformed to fit into our definition of foldability from Section 4.2.3 and consequently assign the binary foldability parameter (Figure 4.20). Manual assessment of simulation results becomes impractical for exploring design spaces with thousands of parameter combinations, necessitating the development of algorithmic approaches that can reliably distinguish successful from failed folding attempts. We develop automated geometric analysis algorithms that transform complex three-dimensional deformation fields into the foldability parameter, enabling systematic analysis of large parameter datasets and providing the foundation for machine learning model development. This computational capability represents the second part of our computational framework outlined in Section 4.2.3, bridging the physics-based simulation framework with data-driven optimization approaches.

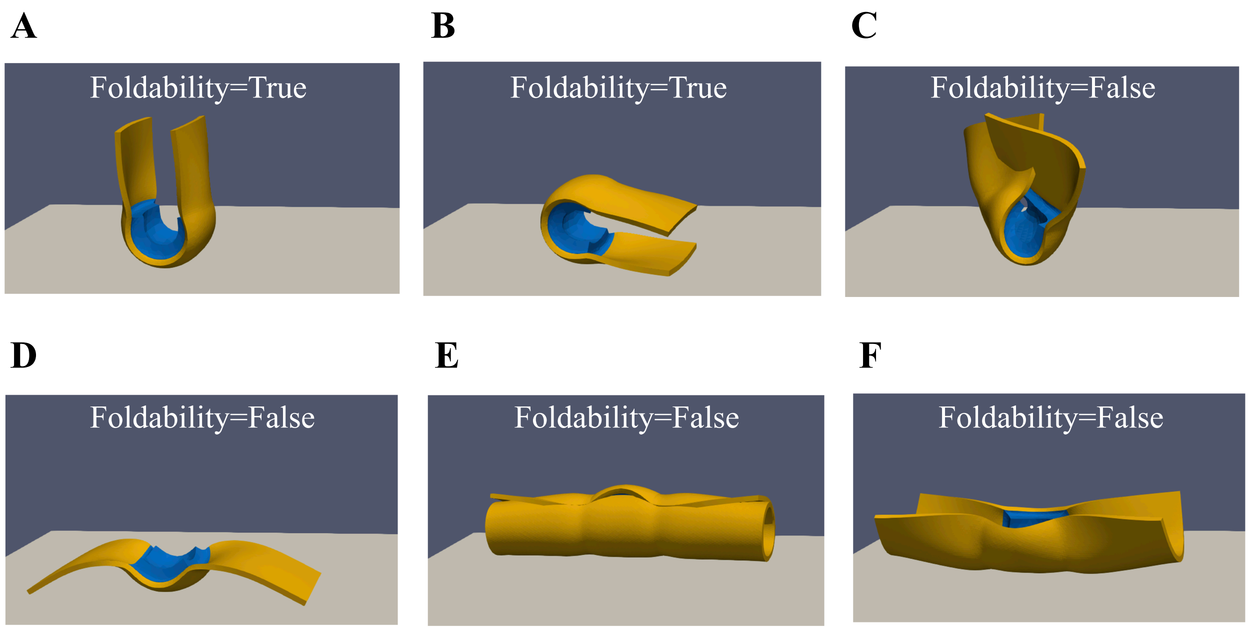


Figure 4.20: The deformed states of different self-folding units(initially flat) under gravity and 40% contraction in the active layer(blue). We can visually identify that self-folding units in **(A)** and **(B)** have desirable foldability (Section 4.2.3) and the other self-folding units have local bending that doesn't satisfy foldability. **(C)** has the panels lifted off the ground but the panels are now warped thus unreliable for folding. **(D)** The differential contraction and mechanics don't translate the bilayer bending to lift the panels against gravity. **(E)** The bending of the bilayer is concentrated in the direction perpendicular to the folding direction and there is curvature developed without folding. **(F)** Similar to (E) and the panels are curved and don't remain plane.

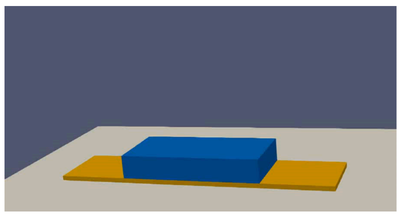
4.5.1 Geometric Analysis for Folded Configuration Characterization

We develop comprehensive geometric analysis algorithms that extract quantitative measures of folding success from complex three-dimensional deformed configurations produced by finite element simulations. These algorithms must operate reliably across diverse folding patterns, geometric scales, and deformation magnitudes while providing physically meaningful metrics that correlate with engineering performance objectives. The algorithmic framework transforms 3D deformation fields on a mesh with more than 10^5 nodes into a binary metric, enabling systematic comparison of foldability across different design configurations and parameter combinations.

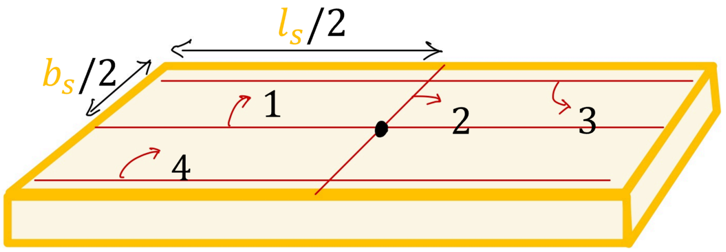
The differential geometry approaches utilize computational techniques that focus on the deformation analysis of four critical curves identified in the material coordinate system of the self-folding unit. We identify these four curves based on their geometric significance for characterizing folding behavior: the midline along the folding direction (C_1), the midline along the folding edge (C_2), and the two lateral hinge edges parallel to the folding direction (C_3 and C_4). These curves in material coordinates provide a representation of deformation for evaluating folding success by capturing the primary deformation of hinge bending and the secondary deformation patterns that indicate proper mechanical coupling between hinge and panel components.

We sample points at regular intervals along these curves and map them to deformed positions, creating corresponding spatial curves that represent the actual geometric state achieved after deformation. The material-to-spatial coordinate mapping utilizes the displacement field from the FEM simulation to transform the regular material coordinate sampling into 3D spatial curves embedded in the deformed self-folding unit (Figure 4.21). The spatial sampling strategy collects 1000 points along each curve, ensures adequate resolution to capture local curvature variations and geometric irregularities while maintaining computational efficiency for large-scale parametric studies.

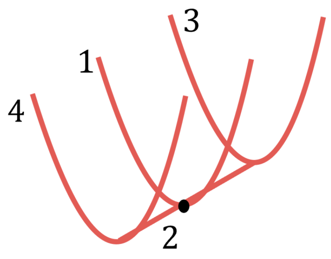
A



B



C



Curve	Equation in material coordinates
1	$x = 0,\ y \in (-l_s/2, l_s/2),\ z = 0$
2	$x \in (-b_s/2, b_s/2),\ y = 0,\ z = 0$
3	$x = -b_s/2,\ y \in (-l_s/2, l_s/2),\ z = 0$
4	$x = b_s/2,\ y \in (-l_s/2, l_s/2),\ z = 0$

Figure 4.21: **(A)** The initial configuration of the self-folding unit which is also the reference configuration because of the absence of any residual stress. **(B)** The mid-plane of the passive layer from the reference configuration of the self-folding unit. The material points are uniformly sampled in the reference configuration from the indexed curves in red, $\{C_1, C_2, C_3, C_4\}$ with coordinates given in the table. **(C)** The curves of the sampled material points change in spatial configuration as the passive layer of the self-folding unit deforms in the process of active contraction of the blue layer.

We compute four distinct geometric measures from these discrete spatial curves (Figure 4.21) that collectively characterize the folding outcome and assess foldability:

- **Curvature measure**: Total accumulated curvature along the folding direction to quantify the overall bending deformation achieved, computed as the magnitude of line integral of the curvature:

$$m_1 = \left| \int_{C_1} \kappa_1 \, \mathrm{d}s \right| \tag{4.18}$$

  where κ_1 is the curvature along C_1, measured by calculating the numerical double derivative of the points on the spatial curve.

- **Straightness measure**: Total accumulated curvature along the folding edge to quantify the lack of bending deformation along this direction, which enables a more defined folded state. This is computed similarly to m_1:

$$m_2 = \left| \int_{C_2} \kappa_2 \, \mathrm{d}s \right| \tag{4.19}$$

  where κ_2 is the curvature along C_2, measured by calculating the numerical double derivative of the points on the spatial curve.

- **Symmetry measures**: We evaluate symmetry and coordination of folding along the length of the folding edge. We choose to compare the curves at the edges of the self-folding unit and the midline running along the folding direction. To measure symmetry between two curves we use a metric that compares the directions of the curves at the same material points:

$$\begin{aligned} m_3 &= \frac{1}{l_s} \int_{C_1, C_3} \frac{\boldsymbol{t_1} \cdot \boldsymbol{t_3}}{|\boldsymbol{t_1}||\boldsymbol{t_3}|} \\ m_4 &= \frac{1}{l_s} \int_{C_1, C_4} \frac{\boldsymbol{t_1} \cdot \boldsymbol{t_4}}{|\boldsymbol{t_1}||\boldsymbol{t_4}|} \end{aligned} \tag{4.20}$$

where m_3 and m_4 is the symmetry metric between spatial curves $\{C_1, C_3\}$ and $\{C_1, C_4\}$ respectively, c_i is the curve in the material coordinates, $\boldsymbol{t_i}$ is the tangent vector along c_i numerically calculated by taking the double derivative of the final coordinates $\boldsymbol{x}$.

These four measures provide complementary geometric information from 3D displacement fields that enables the tagging of folding outcomes across diverse parameter ranges and configuration types. The curvature and straightness measures capture the primary folding characteristics, which requires bending to be concentrated along the folding direction keeping the folding edge straight. While the symmetry measures provide quality assurance that distinguish successful folding about the folding edge from warping or complex shapes. The presence of two symmetry measures in an ideal system is redundant but is included to make the collection of measures more robust. The systematic evaluation of these four measures through automated algorithms eliminates subjective assessment variability and provides standardized metrics suitable for statistical analysis and machine learning model development.

4.5.2 Foldability tagging algorithm

We implement a tagging algorithm that combines multiple geometric criteria to provide robust and reliable assessment of folding success across diverse configuration types and parameter ranges. The tagging system must operate consistently across varying geometric scales, material property ratios, and loading conditions while maintaining sufficient sensitivity to distinguish between successful and marginally successful folding attempts. This approach automates processing of large simulation datasets and provides standardized training data for machine learning methods.

Boolean conditions from geometric criteria

The Boolean logic framework combines geometric criteria including curved edges, straight folding lines, and symmetry conditions through logical operations that reflect the physical requirements for successful self-folding unit operation.

We establish primary criteria for curved edge formation by requiring that the integrated curvature along the hinge centerline exceeds a minimum threshold value, indicating that sufficient bending deformation has occurred to achieve meaningful shape change. The curved edge criterion utilizes the first measure from Section 4.5.1, comparing the computed curvature magnitude against established thresholds that distinguish successful folding from incomplete or failed attempts α_1. We can formulate this as:

$$F_1 = (m_1 > \alpha_1)\,, \quad \alpha_1 = -g_i\pi \tag{4.21}$$

where $g_i < 0$ is the contractile strain in the active layer at the final deformed state. This is formulated to identify that foldability depends on the stimuli-induced contraction, making the foldability classification robust to include all the self-folding units that can potentially achieve folded states.

We implement straight folding line criteria through edge straightness measures, m_2, that ensure the hinge line remains straight throughout the folding process. The straightness criterion requires that curvature represented in m_2 remains below specified values, to fail the folding attempts with undesirable local bending perpendicular to the folding direction.

$$F_2 = (m_2 < \alpha_2)\,, \quad \alpha_2 > 0 \tag{4.22}$$

The first two criteria ensures that both folding magnitude and folding quality meet acceptable standards for engineering applications.

The symmetry criteria utilize symmetry measures m_3 and m_4 to verify that the folded configuration exhibits the expected geometric characteristic of successful self-folding unit operation. For foldability we expect the curves being compared in the calculation of m_3 and m_4 in Equation 4.20 to be entirely parallel. So the values of the two measures approach 1 and the individual symmetry criteria arising from each measure is :

$$F_3 = (|m_3 - 1| < \alpha_3)\,, \quad \alpha_3 \to 0$$
$$F_4 = (|m_4 - 1| < \alpha_4)\,, \quad \alpha_4 \to 0 \tag{4.23}$$

We combine these individual geometric criteria through Boolean logic “AND” operation requiring that curved edge, straight folding line, and symmetry criteria are simultaneously satisfied, ensuring that all essential aspects of folding success are verified before positive classification. The foldability parameter for a simulation result is calculated as:

$$F = F_1 \cap F_2 \cap F_3 \cap F_4 \tag{4.24}$$

where F is the binary foldability parameter and F_i are the binary variables representing the individual criteria (Figure 4.22).

Tuning parameters and implementation

To implement foldability criteria, we set the values of the threshold parameters $\{\alpha_2, \alpha_3, \alpha_4\}$ used in Equations 4.22 and 4.23 to determine binary values for $\{F_1, F_2, F_3, F_4\}$. The values of threshold parameters are tuned using representative simulation datasets that span the expected range of geometric scales and material property ratios. The tuning process examines classification accuracy for threshold parameter values, identifying the right combination of threshold parameters that maximize correct classification rates.

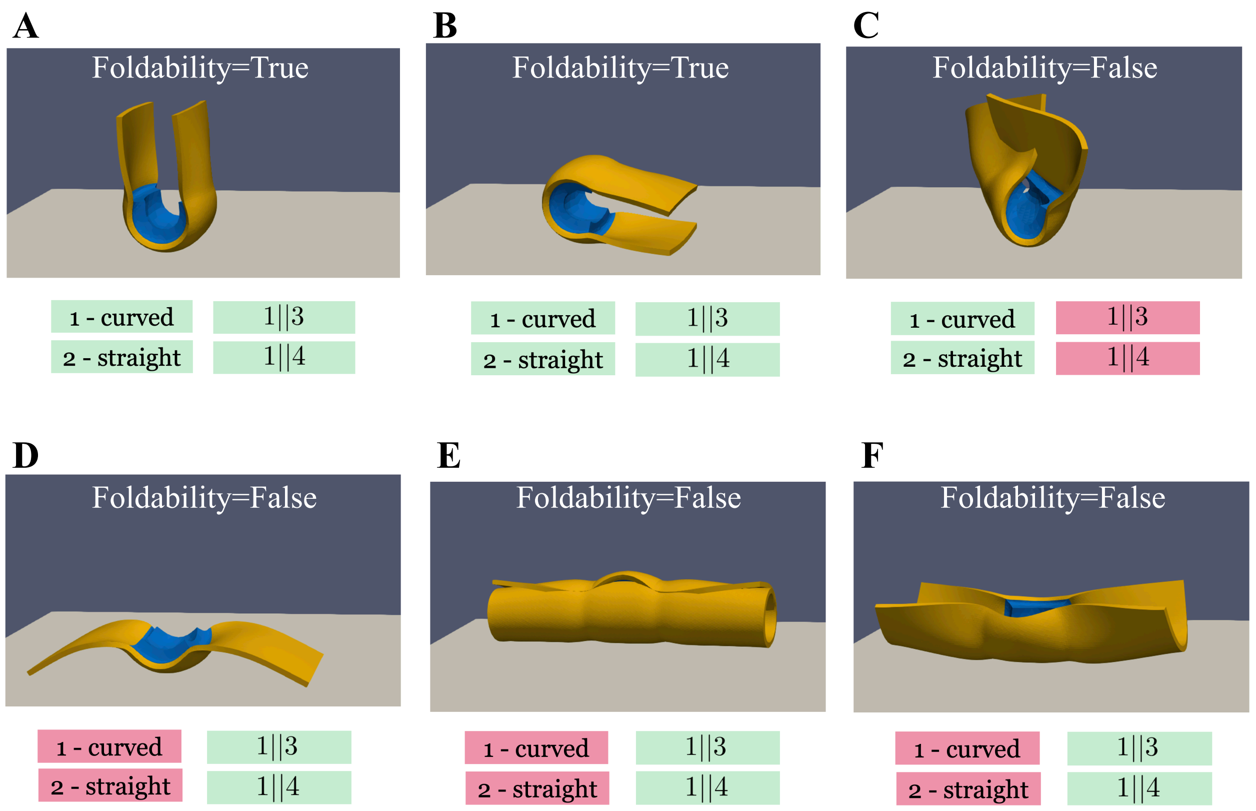


Figure 4.22: Final shapes of different self-folding units from Figure 4.20 deforming under gravity and contraction of the active layer(blue). The self-folding units are classified using the algorithm in Section 4.5.2 into true and false foldability based on their shapes. The individual geometric criteria corresponding to $\{F_1, F_2, F_3, F_4\}$ are shown below the shapes summarized as {“1-curved”, “2-straight”, “1||3”, “1||4”}, colored in green if they are satisfied and in red if they are not.

The threshold parameter values for tagging folding are:

$$\alpha_2 = \alpha_1/2$$
$$\alpha_3 = 0.1$$
$$\alpha_4 = 0.1$$

where α_2 depends on α_1 from Equation 4.21, incorporate that successful foldability the curvature of C_1 always dominates that of C_2, even if the latter is non-zero. Since our simulation is mimicking realistic effects of differential contraction, finite deformation, and gravity we expect deviations from ideal values, in Equations 4.22 and 4.23, even for successful foldability. The deviations are encoded in the values of the threshold parameters and are set so that their corresponding criteria do not show false positives but incorporate all of the successful outcomes. During tuning each threshold parameter is initially set at its ideal value and moved away to ensure perfect accuracy for over 200 simulation results that were manually tagged for their foldability.

Our algorithm for foldability evaluation through geometric analysis provides objective assessment of self-folding unit performance. This supports machine learning model development by enabling the generation of large training datasets from simulation results, in the format represented by Table 4.2.

4.6 Machine Learning methods for Predictive Design

Simulating and automated tagging of self-folding unit behavior across different parameter ranges can generate substantial datasets to train machine learning approaches for rapid design prediction. We develop data-driven models that discover regimes of foldability in the design parameter space, facilitating efficient exploration of design space and prediction for manufacturing. This computational approach represents the third of our three-part computational framework, to enable systematic design synthesis and parameter optimization for self-folding systems.

4.6.1 Overview of Machine Learning Methods

We survey supervised learning approaches applicable to self-folding unit design, focussing on methods for predicting foldability success. The binary nature of the foldability assessment problem, distinguishing successful from failed attempts, naturally aligns with classification methodologies that can learn decision boundaries in high-dimensional parameter spaces. We examine three approaches: symbolic regression for discovering interpretable analytical relationships [70], random forests for foldability classification [71], and neural networks for predicting binary foldability [72].

Symbolic regression can offer unique advantages for engineering applications by discovering explicit mathematical relationships that provide physical insight into the underlying design principles [73]. Unlike black-box machine learning approaches, it generates human-interpretable expressions that engineers can analyze, validate against physical understanding, and apply confidently across diverse operating conditions. Random forests provide robust classification performance through ensemble learning, combining many individual decision trees. They also offer some interpretability through "decisions" used in the classifier which are booleans of the form $(x_i \lesseqgtr a)$ where x_i is a design parameter and a is a tuned constant [74]. Neural networks excel at capturing complex nonlinear relationships and parameter interactions but function as black-box models requiring careful validation and uncertainty quantification for engineering applications [75].

We formulate a tractable problem of foldability prediction and try solving it with the the three machine learning approaches which have different interpretability and accuracy outcomes. Interpretable models enable engineers to understand why certain parameter combinations lead to successful folding, facilitating design insight development and fostering confidence in automated design recommendations. However, interpretable models may sacrifice some predictive accuracy compared to complex black-box approaches, particularly when the underlying physical relationships involve high-order parameter interactions or nonlinear coupling effects. We choose the neural network method to establish a workflow for foldability prediction and further explore improving foldability predictions by selectively choosing training data.

4.6.2 Foldability Prediction in Reduced Design Space

Training datasets for machine learning methods require comprehensive coverage of the design space for reliable model development. The full dimensionless parameter space from Section 4.2.2 includes geometric ratios, material property ratios, and deflection related parameters, creating an eight-dimensional space requiring tens of thousands of simulations for adequate sampling/training. Individual simulations described in Section 4.3 involving gravity, contact constraints, and large deformation analysis demand substantial computational resources on the order of 32-240 GB memory for 10 hours - 6 days to perform a single simulation, making the necessary comprehensive parameter space exploration unfeasible for machine learning model development, with the time and compute resources currently available.

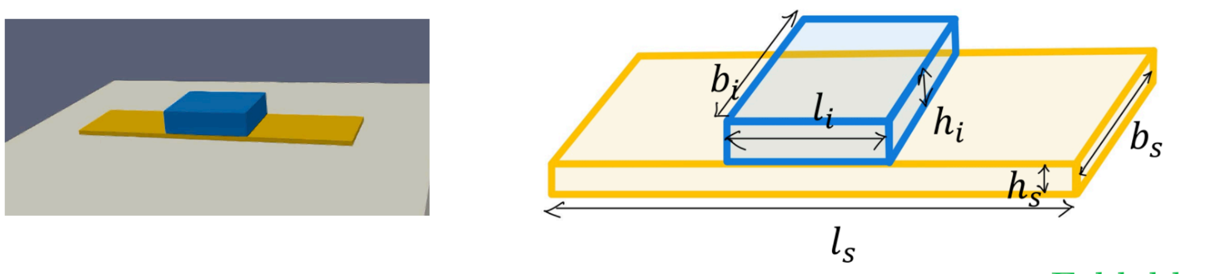
Given these limitations, we explore pathways for data-driven foldability prediction by reducing the design space, which makes sampling datasets approachable by operating in a reduced dimensional space instead of the original eight-dimensional parameter space. We eliminate gravity, which removes all design parameters containing density or gravity terms from the dimensionless design space. Removal of gravity also removes the contribution of panels to folding behavior, because now they remain just the extrapolation of the edges of the bilayer hinge. This reduces the dependance of foldability to pure bilayer hinge effects without the complications of body forces, contact, and panel mass contributions.

We recognize that the gravity-free model corresponds closely to analytical relations that were used to validate the simulations in Section 4.4.1. We therefore choose to use analytical relations to generate extensive training datasets for creating the first machine learning-based workflow, to have more flexibility over ML-model development because of instant training data generation.

Reducing the model to three design parameters for bilayer specification provides a tractable design space for systematic analysis while preserving the fundamental design relationships governing bilayer hinge performance (Figure 4.23). We retain the elastic moduli ratio E_i/E_s that controls the relative stiffness contributions of active and passive layers, the thickness ratio h_i/h_s that determines neutral axis position and bending moment distribution, and the aspect ratio h_s/l_s that characterizes the passive layer thickness using the length scale of l_s. These three parameters capture the essential design trade-offs in bilayer hinge systems while enabling complete parameter space sampling with reasonable computational effort. Another modification to note is that the dimensional changes in active layer are through uniaxial contractions directed in the folding direction, in contrast to isotropic contractions used by default in the simulations. Additionally, this reduced model contributes insight into identifying which self-folding units definitely cannot fold under gravity, because if the self-folding unit cannot fold without gravity we can reasonably assume it will not fold with gravity. Conversely, if a self-folding unit can fold in the absence of gravity it may or may not fold with gravity.

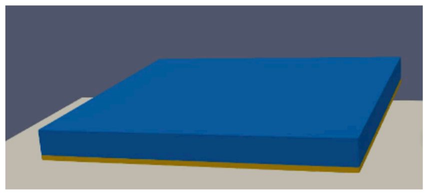
We thus establish a reduced three-dimensional design space and the analytical framework that enables extensive dataset generation across the reduced parameter space. We recognize that reducing the design problem complexity by eliminating gravity decreases the realism of the foldability predictions made by ML approaches, but this simplified design space enables us to build a workflow for inverse design of foldability with our current capabilities. This workflow serves to demonstrate the feasibility of data-driven foldability prediction while providing a pathway for future implementation in more comprehensive design spaces when the limitations around computational resources are addressed.

A



Inverse Design $\left\{\frac{E_i}{E_s}, \frac{\rho_i}{\rho_s}, \frac{l_i}{l_s}, \frac{b_i}{b_s}, \frac{h_i}{h_s}, \frac{E_s h_s^2}{\rho_s g l_s^3}, \frac{b_s}{l_s}, \frac{h_s}{l_s}\right\}? \longleftarrow$ Foldable 1

B



$$g = 0, l_i = l_s, b_i = b_s$$

Inverse Design $\left\{\frac{E_i}{E_s}, \frac{h_i}{h_s}, \frac{h_s}{l_s}\right\}? \longleftarrow$ Foldable 1

Figure 4.23: **(A)** The self-folding unit with the geometric parameters labelled and the inverse design problem for foldability defined using the non-dimensional parameters. **(B)** Reducing the system from a self-folding unit to the bilayer after removing gravity and the panels, and the corresponding inverse design problem for foldability with reduced set of non-dimensional parameters controlling foldability.

4.6.3 Workflow for ML-based Prediction Approaches

We build a workflow to predict foldability using ML approaches in the previously established reduced three-dimensional design space. The workflow establishes protocols for defining inputs, data generation, and performance evaluation to ensure comparison of predictive models built using different ML approaches.

Model Input and Design space

We define the workflow inputs as the three dimensionless design parameters identified in the reduced parameter space: the elastic moduli ratio E_i/E_s, the thickness ratio h_i/h_s, and the non-dimensionalized passive layer thickness h_s/l_s. These input parameters provide complete specification of the bilayer hinge configuration within the simplified physics framework. We establish the design space boundaries where limits on input parameter ranges reflect realistic material property combinations, manufacturing feasibility, and simulation limits. We establish the design space boundaries in powers of 10 along all three axes of inputs. The limits for all three design parameters are as follows:

$$10^{-3} \leq E_i/E_s \leq 10^{3}$$
$$10^{-1} \leq h_i/h_s \leq 10^{1}$$
$$10^{-3} \leq h_i/l_s \leq 10^{-1}$$

Generating inputs for training data

The data generation process employs sampling within the established input parameter limits to ensure comprehensive coverage of the design space without systematic bias toward particular parameter combinations. We implement Latin hypercube sampling of the three-dimensional parameter space, generating datasets with predetermined sample sizes to give thousands of parameter combinations depending on the training requirements of the model. To generate the datasets each parameter is independently sampled from a uniform distribution in the logarithmic

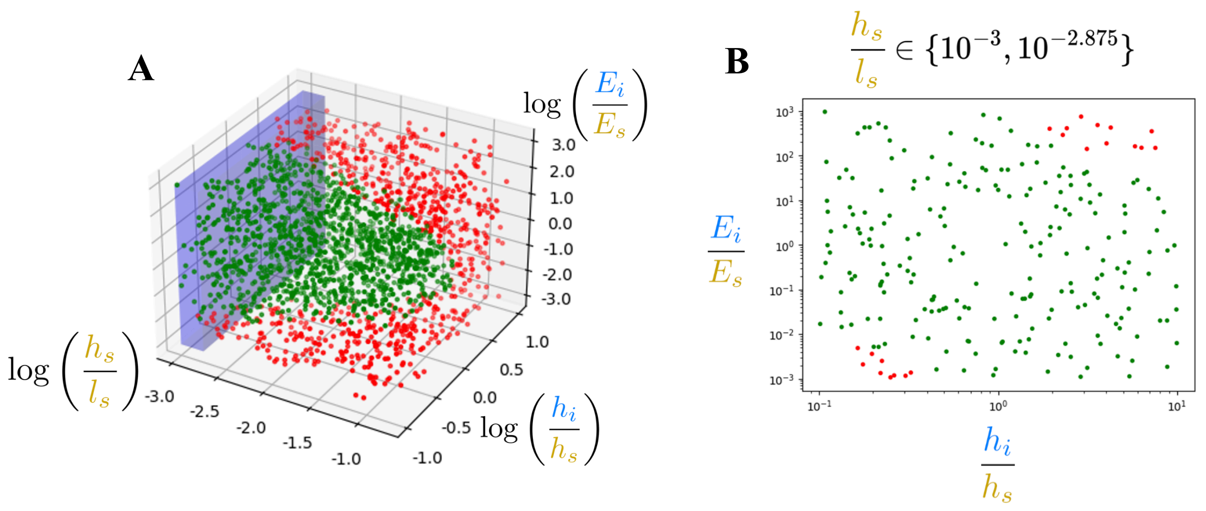


Figure 4.24: **(A)** Plot of the distribution of 2000 generated inputs, with each point representing one input data located based on the input parameters and marked based on its foldability target, green if foldability is true or else false. **(B)** 2D plot showing the generated inputs lying within the blue slice in (A).

space (Figure 4.24).

Generating targets of training data

We apply analytical relations derived from Timoshenko's bi-metallic strip [62] used in Section 4.4.1 to tag any generated input data with binary foldability classifications (Figure 4.24). The analytical approach eliminates the computational overhead of finite element simulations for dataset generation while maintaining physical accuracy within the assumptions of the reduced parameter space. We integrate the analytical relations for curvature over the length of the bilayer l_s, and derive the following equation for angle accumuluated in the bilayer normalized by active strain:

$$\frac{\kappa l_s}{g_i} = \frac{6\left(1+\frac{h_i}{h_s}\right)}{\frac{h_s}{l_s}\left(3\left(1+\frac{h_i}{h_s}\right)^2 + \left(1+\frac{h_i}{h_s}\frac{E_i}{E_s}\right)\left(\left(\frac{h_i}{h_s}\right)^2 + \frac{1}{\frac{h_i}{h_s}\frac{E_i}{E_s}}\right)\right)} \tag{4.25}$$

where κ is the curvature developed in the bilayer hinge for growth strain g_i in the active layer. The foldability criterion is applied as a threshold on this result to produce the binary target of foldability $F\left(\frac{h_i}{h_s}, \frac{E_i}{E_s}, \frac{h_s}{l_s}\right)$ for any input to generate the training data:

$$F\left(\frac{h_i}{h_s}, \frac{E_i}{E_s}, \frac{h_s}{l_s}\right) = \left\{\frac{\kappa l_s}{g_i} \geq 2\pi? \quad \textit{True}(1) \quad \text{else} \quad \textit{False}(0)\right\} \tag{4.26}$$

where the result of the condition on the scalar value $\kappa l_s/g_i$ generates the binary value $F\left(\frac{h_i}{h_s}, \frac{E_i}{E_s}, \frac{h_s}{l_s}\right)$. The training data is now a list of the design parameters and the corresponding foldability target as:

$$\left\{\left\{\frac{h_i}{h_s}, \frac{E_i}{E_s}, \frac{h_s}{l_s}\right\}, F\left(\frac{h_i}{h_s}, \frac{E_i}{E_s}, \frac{h_s}{l_s}\right)\right\}_{1,2,3...M} \tag{4.27}$$

where M is the number of training data points used.

Model Output

Using the training data generated with the foldability targets, we train any ML model to classify the inputs based on foldability. The ML model outputs consist of a binary prediction indicating whether each parameter combination results in successful folding behavior. If the ML model doesn't directly produce a binary number, a threshold is assigned to map the predicted float into binary:

$$F^{\mathrm{p}}\left(\frac{h_i}{h_s}, \frac{E_i}{E_s}, \frac{h_s}{l_s}\right) = \left\{p\left(\frac{h_i}{h_s}, \frac{E_i}{E_s}, \frac{h_s}{l_s}\right) \geq \alpha_F\,? \quad \textit{True}(1) \quad \text{else} \quad \textit{False}(0)\right\} \tag{4.28}$$

where F^{p} is the predicted binary foldability mapped from $p\left(\frac{h_i}{h_s}, \frac{E_i}{E_s}, \frac{h_s}{l_s}\right)$ which is the float value predicted by the ML-model using the scalar threshold α_F which is model dependent.

Defining measure of accuracy

We define the accuracy of foldability prediction by comparing the model predictions of foldability with actual foldability on a set of test inputs. The testing data is independent of the training data used to develop the ML model and is used to see the capability for the model to predict foldability on "unseen" parameter combinations. The accuracy for classification is the number of correct classifications over the total number of classifications. To measure this for testing data, we calculate the following:

$$\text{testing accuracy} = \frac{\sum_{j=1}^{M_\text{test}} F_j^\text{p} == F_j}{M_\text{test}} \tag{4.29}$$

where M_test is the number of testing data points and for each point in testing data j the binary values for predicted foldability F_j^p and actual foldability F_j from Equation 4.26. The comparison between the binary values is converted to float to maintain precision when calculating the accuracy:

$$(F_\text{p} == F) = \begin{cases} 1.0 & \text{if} \quad F_\text{p} = F \\ 0.0 & \text{otherwise} \end{cases}$$

Using this outlined workflow, we build and train 3 machine learning models to predict foldability and compare their prediction accuracy. While benchmarking all the models we use the same 6000 training data points and 1000 testing data points. We divide the parameter space into regimes of foldability based on the results of the ML model with the highest prediction accuracy and discuss additional modifications to develop the model further.

4.7 Preliminary Results from Machine Learning Approaches

We investigate complementary machine learning methodologies to provide comparative assessment of prediction accuracy and interpretability across different approaches, establishing the relative advantages and limitations of each method for self-folding unit design. These preliminary investigations inform the selection of optimal machine learning strategies and provide backup approaches for scenarios where neural network methods prove inadequate or inappropriate.

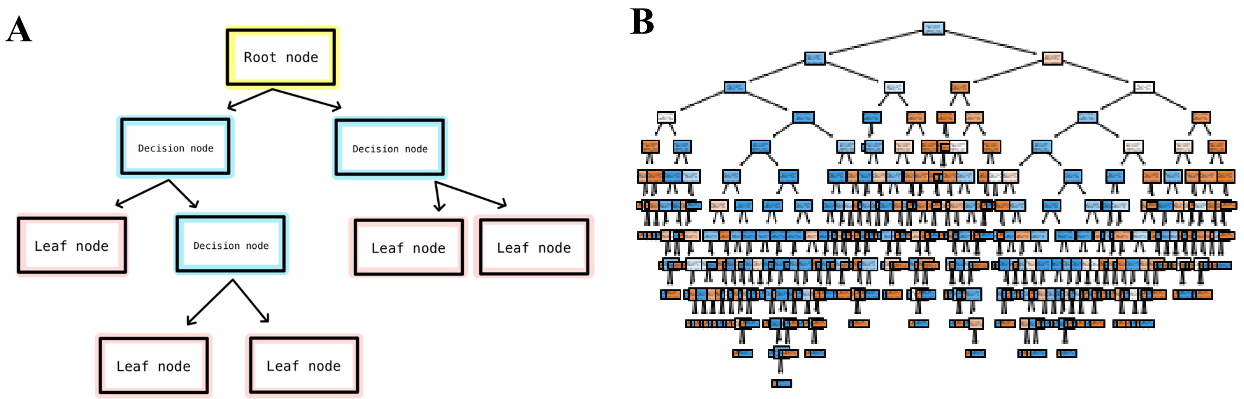


Figure 4.25: **(A)** Schematic of a decision tree classifier. Each decision nodes is a condition based on inputs, where the decision flows to two nodes below based on whether the condition is satisfied or not. The root node is the first decision node. In a classifier, the leaf nodes, which are terminal, are true/false outputs. **(B)** The trained decision tree for classifying foldability on the reduced design space.

4.7.1 Decision Tree Classifier

Decision trees provide interpretable rules and parameter thresholds through hierarchical splitting criteria that explicitly reveal the decision logic underlying foldability classification [76]. We implement decision tree algorithms that systematically partition the parameter space based on threshold values for each dimensionless design parameter, creating explicit rules of the form "if $E_i/E_s >$ threshold$_1$ and $h_i/h_s <$ threshold$_2$, then folding succeeds" (Figure 4.25 A). The resulting decision trees provide immediate engineering insight into critical parameter ranges and design constraints while maintaining some interpretability through the partitions and thresholds. However, decision tree approaches may struggle with complex parameter interactions and may exhibit reduced accuracy compared to other machine learning methods. We used the workflow in Section 4.6.3 to train a Decision Tree classifier [77] in *Python* which led to the accuracy of predicted foldability being 0.962. The interpretability from the partitions was limited because of the large number of partitions in the decision tree which aligns with the general understanding of the utility of decision trees (Figure 4.25 B).

4.7.2 Symbolic Regression

We implement symbolic regression using PySR (Python Symbolic Regression) to discover analytical relationships between design parameters and foldability outcomes, aiming to generate explicit mathematical expressions providing both prediction capability and physical insight [70]. The symbolic regression approach searches through combinations of mathematical operators and parameter relationships to identify expressions that accurately predict foldability outcomes while maintaining mathematical simplicity and interpretability (Figure 4.26).

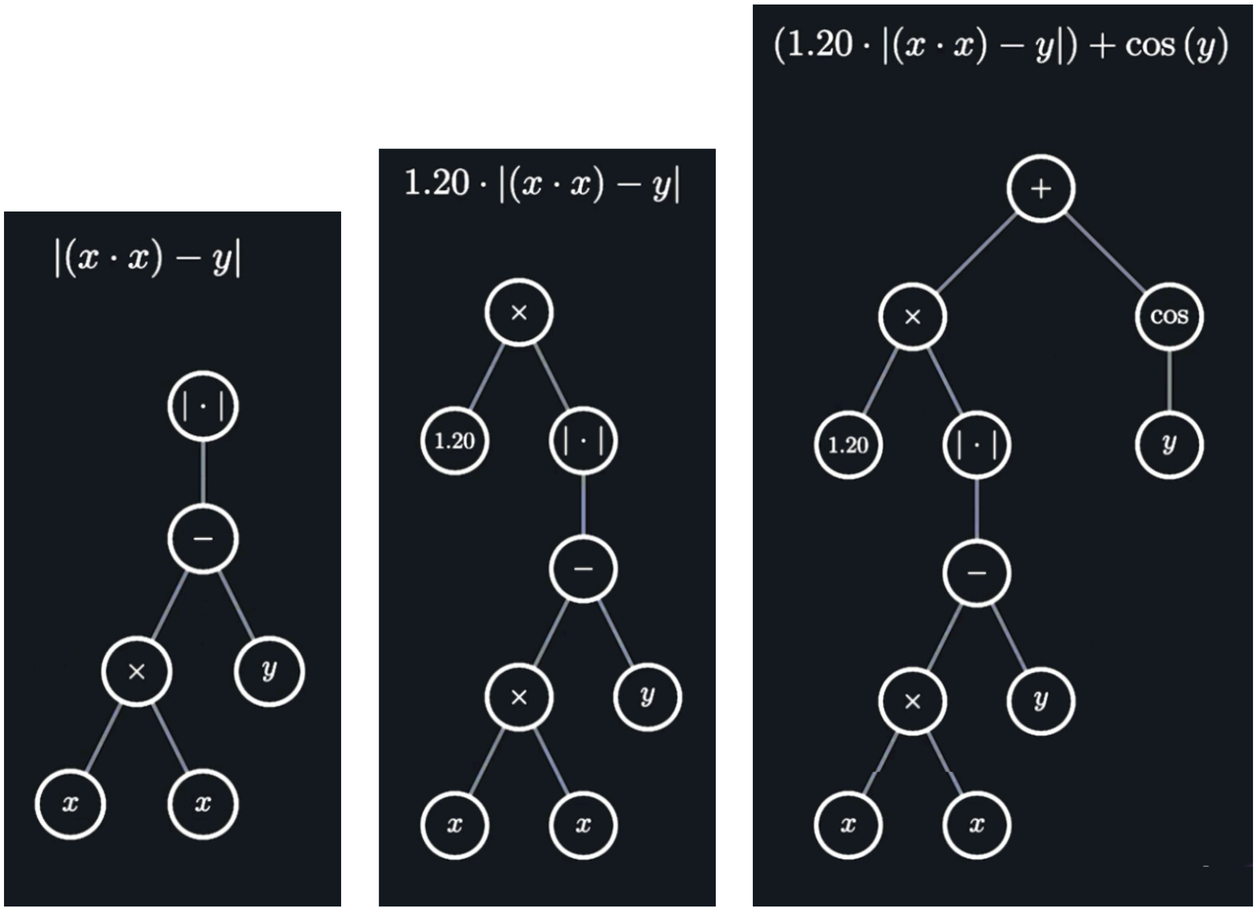


Figure 4.26: Example of PySR evolving a symbolic expression, left to right, by adding complexity through constants, variables, and operators in each step. The symbolic expressions of each evolution step is shown with its corresponding node tree structure. In each step, many possible node additions or mutations are sampled and the best performing change is finally implemented.

We configure PySR with operator sets to include typical engineering relationships such as power laws and rational expressions, ensuring that discovered relationships align with expected physical behavior. The allowable binary operators are $+, -, \times, /$ and the allowable unary operator is log, which is further constrained to avoid any cascading of the log operator in the expression. Furthermore, in training the PySR model we use the scalar target of $\kappa l_s/g_i$ from Equation 4.25 since we want to make the model predict an analytical expression, which is suited to train using scalar target rather than boolean foldability. Additionally, we constrained the depth of the expression tree to 7 to bias for simpler expressions which would be more interpretable. For measuring accuracy of foldability prediction, the output of the analytical expression was thresholded based on Equation 4.26 to generate a binary predicted foldability F_{p}. After multiple attempts of tuning the ML model and training, our best results so far were not very interpretable as expressions and also did not achieve high prediction accuracy, 0.76. The resulting expression from the symbolic regression model predicting $\kappa l_s/g_i$ was:

$$\frac{1.19}{\frac{h_s}{l_s} \cdot \frac{h_i}{h_s}\left(\left(\frac{E_i}{E_s}\right)^2 \frac{h_i}{h_s} + 2.23\right) + 5.12\frac{h_s}{l_s} + \frac{\frac{h_s}{l_s}}{\frac{h_i}{h_s}\left(\frac{h_i}{h_s}+0.00521\right)}} \tag{4.30}$$

The only possible advantage of further developing a symbolic regression model for foldability prediction is that to train the model we only needed 2000 datasets and using more datasets did not significantly alter model results.

4.7.3 Neural Networks

We implement a feedforward neural network architecture in Python using PyTorch [78] designed for binary classification of foldability outcomes in the reduced three-dimensional parameter space. We ran a few investigations on network depth and width that examine classification performance as functions of hidden layer count and neuron density, identifying the best configurations that balance predictive accuracy with computational efficiency and overfitting resistance.

The network architecture (Figure 4.27) consists of an input layer with three neurons

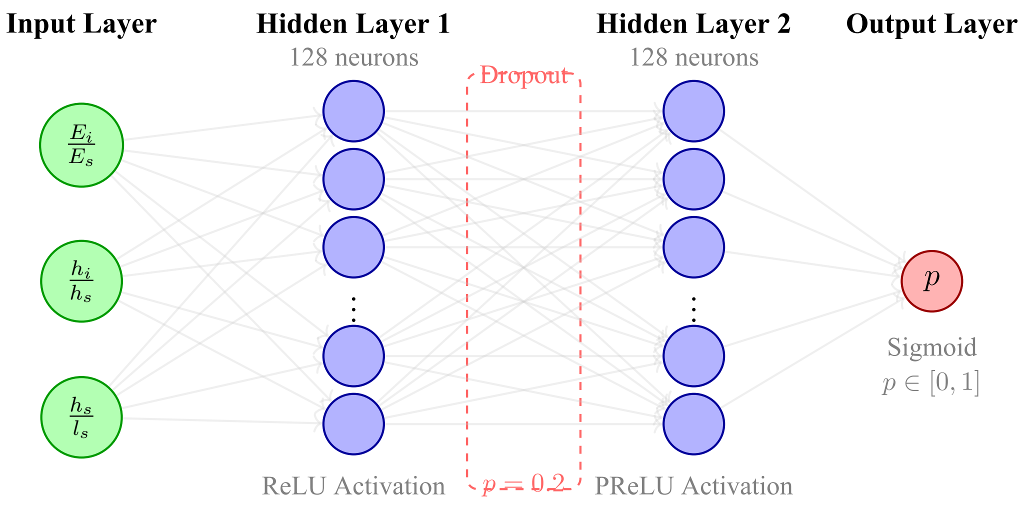


Figure 4.27: Neural network architecture for foldability prediction.

corresponding to the dimensionless design parameters $(E_i/E_s, h_i/h_s, h_s/l_s)$, two hidden layers each containing 128 neurons to capture complex nonlinear parameter interactions, and a single output neuron providing probability of foldability success. The first hidden layer employs ReLU activation functions to introduce nonlinearity while maintaining computational efficiency, followed by a dropout layer with 0.2 probability to prevent overfitting during training. The second hidden layer utilizes Parametric ReLU (PReLU) activation to allow adaptive learning of the activation slope, enabling the network to optimize its representational capacity for the specific characteristics of the foldability classification problem. The output layer implements a sigmoid activation function to produce probability values between 0 and 1. Thus the output of this neural network is a scalar $p(E_i/E_s, h_i/h_s, h_s/l_s) \in [0, 1]$, which represents the probability of foldability success. To map the probability of foldability to a binary foldability we use Equation 4.28 and use a threshold of 0.5:

$$F_{\mathrm{p}}\left(\frac{h_i}{h_s}, \frac{E_i}{E_s}, \frac{h_s}{l_s}\right) = \left\{ \quad p\left(\frac{h_i}{h_s}, \frac{E_i}{E_s}, \frac{h_s}{l_s}\right) \geq 0.5\,? \quad \mathit{True}(1) \quad \text{else} \quad \mathit{False}(0) \right\} \tag{4.31}$$

We train the network using the Adam optimizer [79, 80] with a learning rate of 0.001 and default momentum parameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$) to ensure stable convergence, while employing binary cross-entropy loss to optimize classification accuracy for the foldability prediction task.

The accuracy of the neural network on the testing data was measured from the predicted binary foldability using Equation 4.29 and we found it to be the most promising achieving up to 97.7%. We note that the neural network though most accurate, is essentially a blackbox when it comes to the reasoning of how the foldability classification is being made.

Comparing the preliminary implementations of the different ML models validates the distinct advantages and trade-offs between the approaches. Neural networks demonstrate superior predictive accuracy for complex parameter interactions but provide limited interpretability for design insight development. Decision trees offer transparency of classification but may sacrifice accuracy for complex nonlinear relationships because of direct partitioning. Symbolic regression provides complete transparency by generating interpretable analytical expressions but the approach struggles with complex parameter interactions that resist compact mathematical representation. Given these specific limitations and advantages of each approach for self-folding unit design optimization applications, we see that neural networks would likely excel in scenarios requiring maximum predictive accuracy and can handle arbitrary complexity in parameter relationships, making them suitable for application in predicting foldability in the extended design space with gravity and potential application to guide manufacturing where prediction reliability is paramount.

4.7.4 Regime prediction

We predict foldability regimes using the trained machine learning model that is most promising, the neural network, to identify distinct regions in the reduced parameter space corresponding to successful and unsuccessful folding behavior. When applied to the extended design space, the regimes of foldability provide guidelines and parameter selection strategies for inverse design of self-folding units.

Foldability regimes in reduced design

To map the regimes in the reduced design space, we densely sampled 100,000 points of design parameter combinations $(E_i/E_s, h_i/h_s, h_s/l_s)$. For each of these points, using the selected ML model we predict the foldability using which the parameter combinations are tagged as foldable or non-foldable. Using the tags, we generate heat maps of foldability in the design space which help visualize the decision boundaries that illustrate the transitions between foldable and non-foldable parameter combinations, to identify promising design regions and avoid parameter combinations likely to result in folding failure (Figure 4.28).

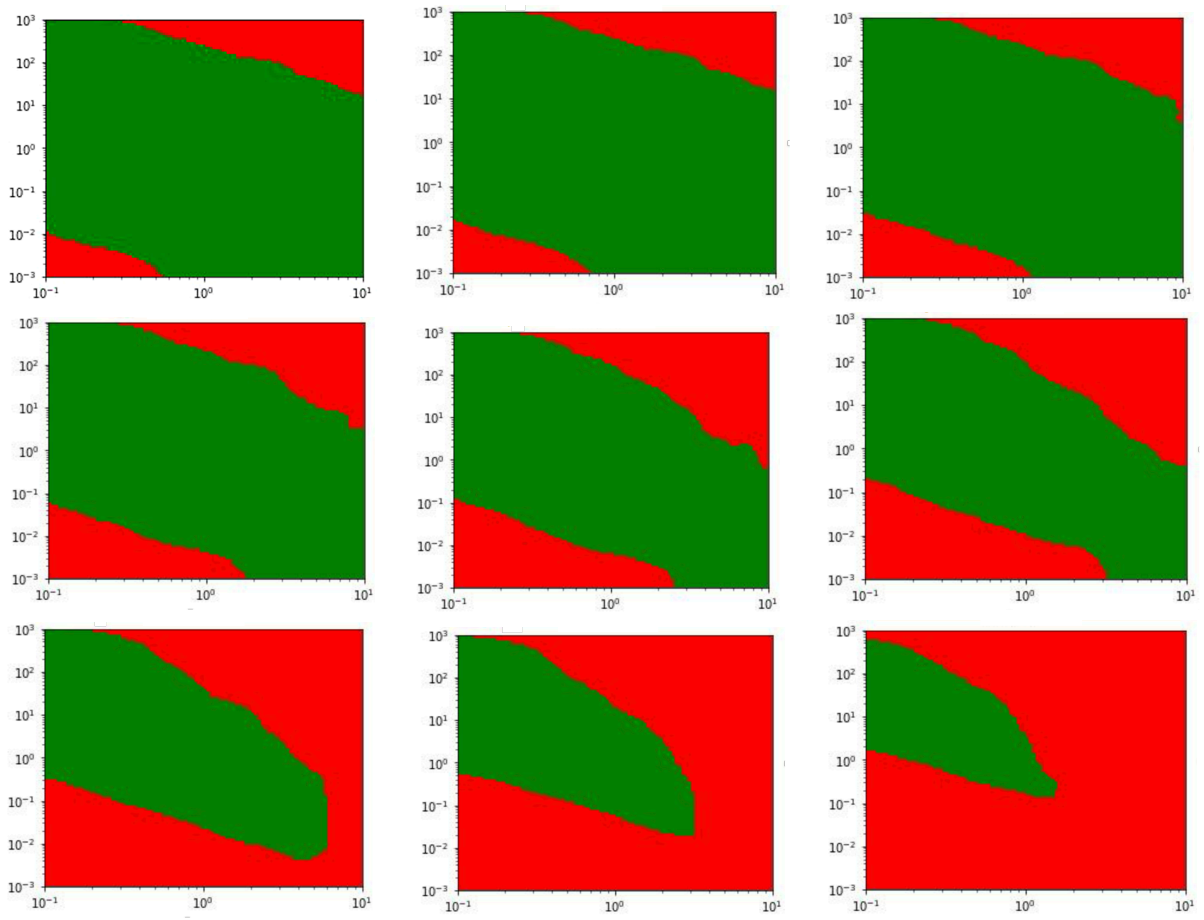


Figure 4.28: Plotting regimes of predicted foldability where green is where foldability is predicted true and red is false. The x and y axes are $\{E_i/E_s, h_i/h_s\}$. Each 2D plot represents a slice along the axis of h_s/l_s. For each plot we have the values of h_s/l_s as $\{10^{-3}, 10^{-2.75}, 10^{-2.5}, 10^{-2.25}, 10^{-2}, 10^{-1.75}, 10^{-1.5}, 10^{-1.25}, 10^{-1}\}$, left to right and top to bottom.

Validating the foldability regimes

For the reduced design space, we can also compare the predicted foldability regimes to the actual foldability regimes, calculated like the training data targets using Equations 4.25 and 4.26, to validate our regimes and identify the location of false positives and false negatives of the predicted regimes (Figure 4.29). This helps us gain a better understanding on where the errors in foldability are located and guide potential model developments.

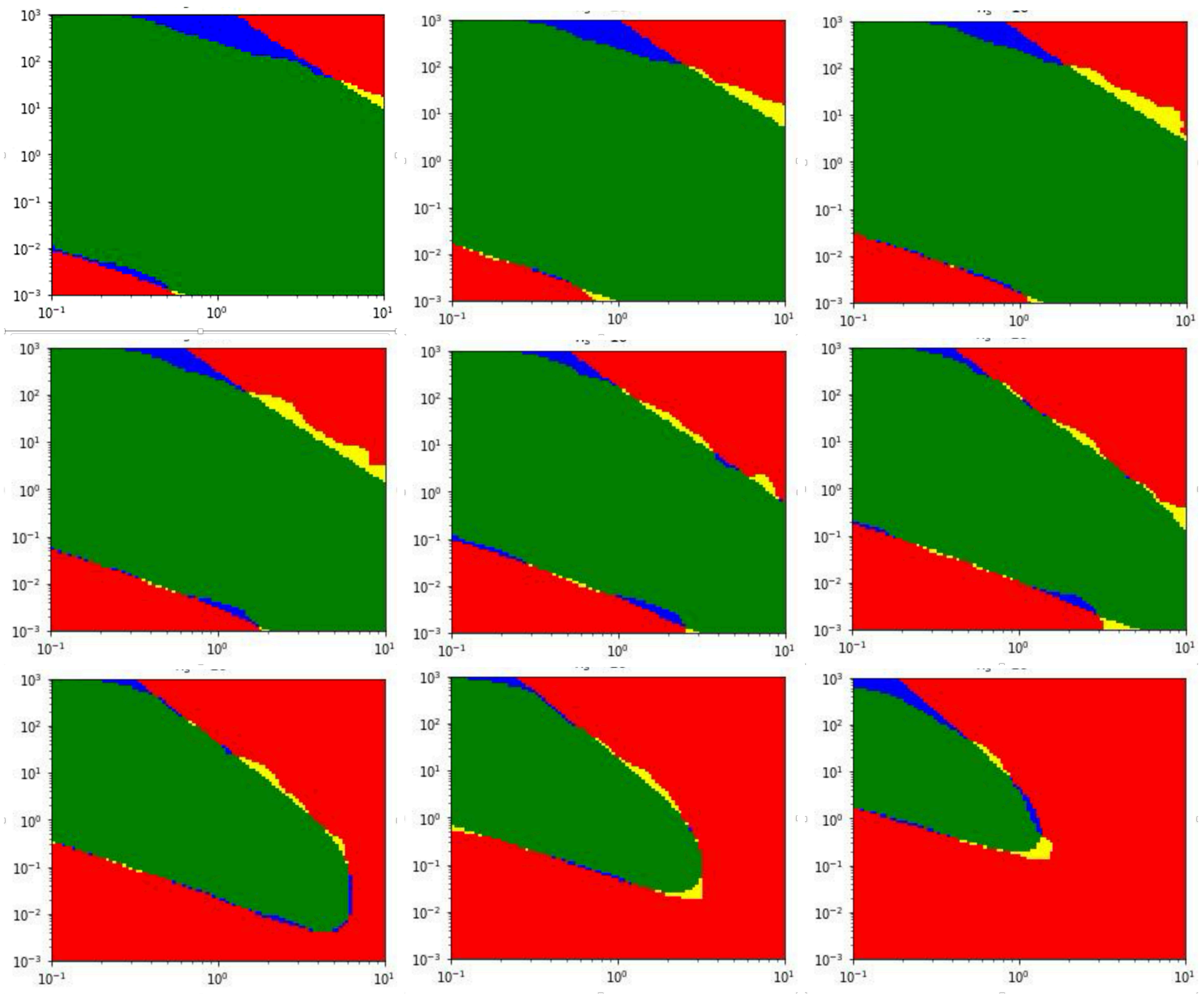


Figure 4.29: Plotting regimes of predicted foldability validated against analytical relations in Equation 4.26. Green and red is where foldability is predicted as true and false accurately. Blue is where the prediction is a false negative and yellow is false positive. The x and y axes are $\{E_i/E_s, h_i/h_s\}$. Each 2D plot represents a slice along the axis of h_s/l_s. For each plot we have the values of h_s/l_s as $\{10^{-3}, 10^{-2.75}, 10^{-2.5}, 10^{-2.25}, 10^{-2}, 10^{-1.75}, 10^{-1.5}, 10^{-1.25}, 10^{-1}\}$, left to right and top to bottom.

Reducing false positives

For our intended application of the foldability prediction to enable inverse design of self-folding units, it is crucial to reliably predict foldability while avoiding false predictions of success. The implication of this when developing data-driven prediction is to reduce false positives, where the predicted value is true but actual value is false. This may sacrifice on accuracy of foldability prediction and overall shrinking of the regime for successful foldability, but increases the reliability of predictions when used in engineering applications. We propose the following two ways of reducing false positives in our model:

- **Biasing folding probability**: We define the predicted binary foldability from the neural network prediction of probability of foldability success based on Equation 4.31. We evenly divide the output probability into failure and success using the probability threshold 0.5. Increasing this threshold by small amounts shifts the binary prediction to be true only in regions where the probability is stronger.
- **Biasing the loss function**: While developing the neural network as detailed in Section 4.7.3, we use BCELoss (pytorch link) to formulate a loss function which is equivalent to “errors” that is minimized as the network trains to predict foldability. By default the BCELoss function penalizes false positives and false negatives equally and by minimizing the total penalty the model predictions becomes more accurate. But there exists an option to add bias to the BCELoss function by assigning different weights for the penalty on false positives and false negatives. Using this option, we can tune these weights so that false positives are more heavily penalized than false negatives.

4.7.5 Recursive Generation of Training Data

The foldability classification problem exhibits distinct regimes separated by decision boundaries that indicate transitions between successful and unsuccessful folding behavior. Within each regime in the design space, foldability outcomes remain consistent with minimal variation, while the boundaries between regimes represent critical transitions where small parameter changes can dramatically alter folding success. These decision boundaries especially require precision to ensure accurate classification performance. We also recognize that decision boundaries need enhanced prediction accuracy, because the misclassifications may be higher near the critical transitions, like Figure 4.29, and can lead to unsuccessful design recommendations or missed opportunities for successful folding configurations.

When sampling training data, we can strategically include more points near decision boundaries to increase prediction accuracy without drastically increasing the total number of training points, maintaining computational efficiency while improving model performance in the most critical regions of the parameter space. This targeted sampling approach becomes particularly important for extended design spaces where generating training data from simulations represents the primary computational bottleneck and is the limitation for machine learning model development.

We implement a recursive training data generation strategy that iteratively augments the dataset by identifying and densifying regions near predicted decision boundaries. To implement this in neural networks, we assume the complete training process utilizes $10n$ total epochs, which are passes through the network, where we can approximate $10n = 10,000$ for comprehensive model development. The training epochs are organized into iterative cycles that progressively improve boundary resolution through targeted data augmentation summarized in Algorithm 5.
We first conduct n epochs of training on the current training dataset using standard backpropagation and the Adam optimizer, allowing the neural network to learn the existing data patterns and establish preliminary decision boundaries. We collect data points around locations with predicted outputs near 0.5, specifically targeting the range of 0.4 to 0.6 where classification uncertainty is highest and decision boundaries are most likely to occur. The selection range can be tuned based on the desired number of training points to be added in each training cycle. For each identified uncertain prediction, we generate foldability targets for the input parameters evaluate

Algorithm 5 Recursive Training Data Augmentation

```
Require: Initial training dataset $D_0$, total epochs $10n$, cycle length $n$, uncertainty range $[0.4, 0.6]$
 1: Initialize $D \leftarrow D_0$, cycle $\leftarrow 1$, untrained neural network in presrcibed architecture
 2: while total epochs $< 10n$ do
 3:     Train network for $n$ epochs on current dataset $D$
 4:     Sample a dataset of input parameters across design space and with their predicted outputs
 5:     Select the inputs with predictions in range $[0.4, 0.6]$
 6:     Create dataset $D_{\text{new}}$ from selected inputs
 7:     Assign foldability targets for $D_{\text{new}}$ using analytical relations/simulations
 8:     Augment training dataset: $D \leftarrow D \cup D_{\text{new}}$
 9:     Increment cycle counter
10: end while
        return Trained neural network
```

using the analytical relations, creating training data D_{new} specifically targeting decision boundary regions. We create an augmented training dataset by combining the original training data with the newly generated boundary-focused samples, ensuring that subsequent training cycles benefit from enhanced resolution in the most critical regions of the parameter space. This recursive process repeats for the specified number of cycles, progressively refining the decision boundaries and improving classification accuracy in regions where predictions are most important to increase the overall prediction accuracy.

The recursive generation approach provides several advantages for self-folding unit design applications. The method automatically identifies and targets the most critical regions of the parameter space without requiring prior knowledge of boundary locations. The approach concentrates the overall training data generation to where they provide the greatest impact on model performance. This approach enables the neural network to progressively learn accurate decision boundaries with lesser epochs/passes and lesser training data than compared to neural networks without this modification, which reduces compute resources for training the neural network as well as for generating training datasets.

4.8 Computational Framework and Limitations

We have developed a computational framework that enables both forward and inverse design of self-folding units through the integration of three complementary research projects: physics-based simulation, automated foldability evaluation, and machine learning prediction. This framework provides methods for predicting folding outcomes and identifying parameter combinations that achieve target foldability objectives.

4.8.1 Forward Design Capabilities

The forward design framework predicts foldability outcomes and final shapes from specified self-folding unit parameters through finite element simulations that capture the complete three-dimensional deformation behavior. We implement FEM simulations (Section 4.3) that predict final 3D shapes through displacement fields on meshes with 100,000+ degrees of freedom, incorporating large deformation, gravity, and contact constraints to accurately represent the folding process from initial flat configuration to final folded state. We employ automated foldability evaluation algorithms (Section 4.5) that utilize differential geometry techniques to extract four key features from complex displacement fields and assign a binary foldability to the simulation result. This algorithm eliminates subjective assessment variability while enabling quantitative analysis of a large set of simulations.

4.8.2 Inverse Design Methodology

The inverse design capability identifies parameter ranges and combinations that generate target folding behaviors, specifically focusing on achieving foldable configurations from self-folding units through systematic design space exploration. We partition the entire design space into foldable and non-foldable regimes using predictions from machine learning-based classifiers trained on comprehensive simulation datasets with foldability targets assigned through the automated foldability evaluation algorithms.

We established a complete workflow for developing ML-based predictors trained on thousands of simulation results (Section 4.6.3), enabling identification of design regimes and parameter combinations that achieve successful folding outcomes. Current computational resource limitations for generating simulation datasets compromise our goal of inverse design, but we developed a pathway for inverse design through regime identification for a reduced design problem. In the reduced design problem the self-folding unit is simplified to bilayer folding without gravity, where preliminary model development utilizes flexible training data generation through analytical relations.

We evaluate three machine learning approaches (Section 4.7), identifying neural networks as the most promising method based on classification performance. Using neural network predictions, we delineate foldable and non-foldable regimes in the reduced design space and validate the regimes. We propose a recursive training data augmentation strategy (Section 4.7.5) to improve prediction accuracy efficiently by targeting data generation at decision boundaries for maximum impact, potentially requiring fewer training samples when implementing inverse design in extended design spaces.

4.8.3 Future directions

Training data generation

The extension of inverse design capabilities to the complete design space of the self-folding unit faces significant computational challenges related to training dataset generation from finite element simulations. Full-scale simulations require substantial CPU resources and extended computation times (on average, 128GB over 3 days for a single simulation), limiting the feasibility of generating comprehensive datasets for machine learning model training. These limitations can be overcome in the future by extending access to CPU resources as well as through parallelized FEM implementations on GPU architectures. We demonstrated $\approx 3x$ speedup from parallelization of test simulations on $\sim 10^6$ degrees of freedom systems for a single step. This has the potential to be scaled with additional GPU resources to self-folding unit simulations with $\sim 10^6$ degrees of freedom over multiple evolution steps. (Table 4.3)

DOF	CPU time(s)	GPU time(s)	Speedup	CPU node memory	GPU node memory
4×10^4	44.6	33.7	1.3	16 GB	640 GB
10^6	1135.4	331.5	3.4	32 GB	640 GB

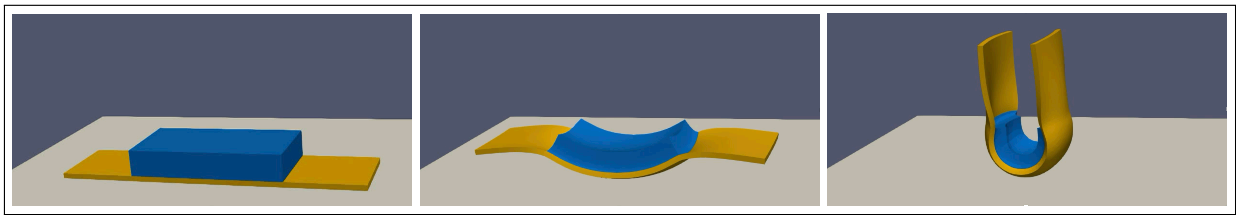
Table 4.3: Results from benchmarking the performance of a single-step FEM simulation in FEniCS executed on CPU and GPU. The execution time on GPU is shorter than that on CPU, but requires access to larger compute memory simultaneously.

Patterned bilayer

A patterned bilayer design moves away from a fully filled active layer on top of the passive layer toward an active layer present as discrete strips on the passive layer. This approach optimizes manufacturing by using less material and aligns folding in the preferred direction of the active layer strips, making the design more robust. Even with local isotropic active layer contraction, the patterned design achieves the same effect as uniaxial contraction due to the directionality of the strip pattern.

We compared folding outcomes between non-patterned and patterned bilayers with identical self-folding parameters and contraction values (Figure 4.30). We see similar successful folding outcomes for the self-folding unit with and without the patterned active layer. We also simulated bilayer bending for bilayers with solid and patterned active layers, using identical material properties, geometrical dimensions, and isotropic contraction in the active layer (Figure 4.31). The bending in the solid bilayer occurs along a diagonal axis, which may not translate to effective and stable folding when used as a bilayer hinge. In contrast, the patterned bilayer bends with curvature concentrated along a single direction, with the pattern controlling the direction of effective contraction and, consequently, bending and folding. The width of strips and spacing between strips would be additional design parameters, if the design problem extends to self-folding units with patterned active layer.

A



B

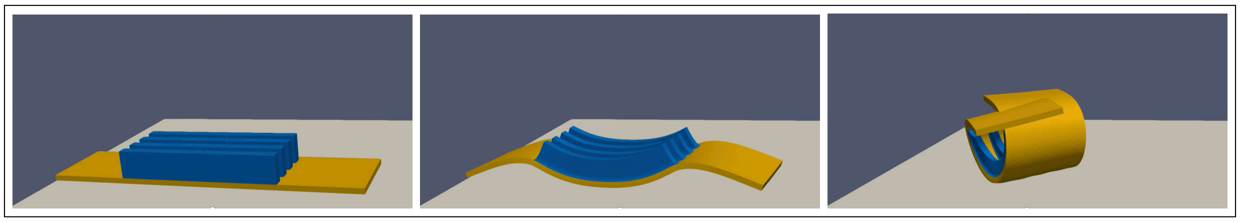
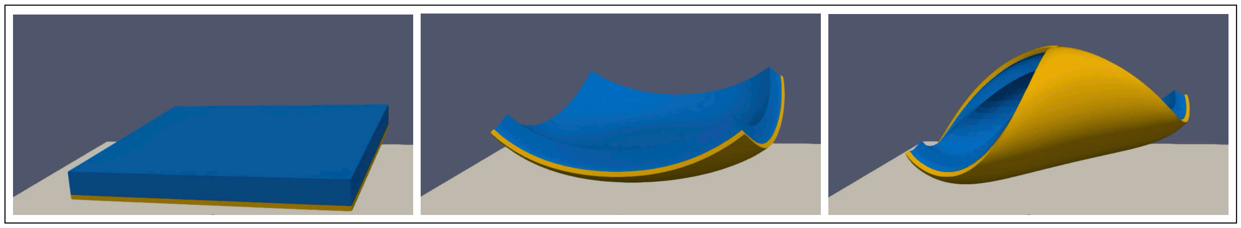


Figure 4.30: Shapes of self-folding units deforming under gravity and contraction of the active (blue) layer, with contractions = $\{0, 0.2, 0.4\}$ for both the left, middle, and right configurations. **(A)** Self-folding unit deforming from a flat to a folded state. **(B)** Self-folding unit made from a patterned bilayer hinge, with all material and geometric properties identical to (A) except the presence of cuts in the bilayer hinge.

A



B

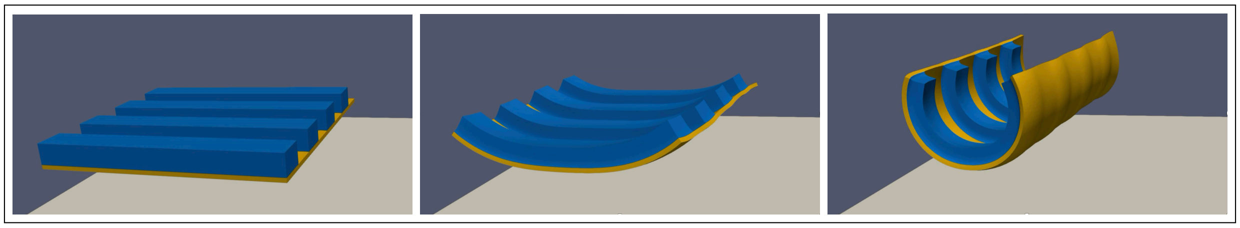


Figure 4.31: Shapes of bilayers deforming under gravity and contraction of the active (blue) layer, with contractions = $\{0, 0.12, 0.24\}$ for both the left, middle, and right configurations. **(A)** Bilayer bending from a flat state to a curved state with curvature in the final state concentrated along the diagonal. **(B)** Bilayer made from a patterned bilayer hinge, with all material and geometric properties identical to (A) except the presence of cuts in the bilayer hinge.

4.8.4 Limitations in the computational framework

The computational framework demonstrates significant capabilities for self-folding unit design, but relies on a few idealized assumptions that may not fully capture real-world folding behavior.

Perfect contact surfaces

Our simulation framework adopts idealized assumptions that represent the optimal extreme for foldability prediction, excluding potential mechanical losses through imperfect contact behavior between material interfaces. These assumptions cause our models to predict best-case scenarios with gravity rather than more realistic observations. The characteristics of contact surfaces fundamentally affect foldability through their influence on force transmission, stress concentration, and deformation compatibility throughout the folding process. Imperfect bonding at the bilayer interface can lead to delamination under differential strains induced by stimuli activation, effectively decoupling active and passive layers and partially or fully eliminating bending moments necessary for folding. Similarly, inadequate attachment between hinges and panels can result in localized deformation rather than coordinated folding motion, where hinges bend locally without effectively rotating attached panels through desired folding sequences. Additionally, we are also ignoring the presence of any minor dissipative friction from the contact between the self-folding unit and the “ground” surface.

Static material properties

The current simulation framework does not include changes in material properties during stimuli-induced dimensional changes in active layers, which experimental studies demonstrate can result in varying material stiffness and density based on the specific stimulus used to trigger strains [81]. These material property variations can significantly influence the folding kinematics through energy functionals, potentially affecting both the magnitude of achievable curvature and the stability of folded configurations. We developed one variation of the model that changes material properties as a function of the evolution parameter as the simulation progresses, which accounts for some of these coupled material behavior effects to improve any future experimental validation and design.

Despite these limitations, the developed computational framework provides a foundation for self-folding unit design, establishing modeling approaches that can accommodate increased physical complexity as computational capabilities advance and material characterization data becomes available.

Chapter 5

Viscoelastic morphing

The viscoelastic morphing results in this chapter are generated for "Patterns of Mitochondrial ATP Presage and Predict Apical Constriction-Driven Morphogenesis"(under review). The experimental image data and domain geometries shown in Section 5.5 are contributed by Dr. Bezia Lemma at Tissue Morphodynamics Laboratory, Princeton University.

5.1 Introduction and Motivation

The computational framework developed in Chapter 2 provides a robust foundation for modeling growth-induced deformation under the assumption of purely elastic material behavior and quasi-static evolution of growth states. While this framework successfully captures many morphogenesis phenomena, biological tissues can exhibit time-dependent mechanical response that influence developmental processes. This chapter extends the morphoelastic theory to incorporate viscoelastic constitutive relations, enabling more exhaustive modeling of morphogenesis across diverse systems.

5.1.1 Limitations of Purely Elastic Models in Biological Tissues

Biological tissues are complex, hierarchical materials composed of cells embedded within extracellular matrices (ECM), permeated by interstitial fluids, and subject to continuous remodeling [82]. This structural complexity manifests in mechanical behaviors that can deviate from the instantaneous, reversible responses assumed in elastic models. Three primary time-dependent phenomena characterize tissue mechanics: stress relaxation, creep, and hysteresis [83]. The elastic framework presented in Chapter 2 cannot capture these temporal dependencies, limiting the application of models in predicting morphologies when mechanical relaxation timescales are comparable to growth timescales.

Consider epithelial morphogenesis, where tissue folding occurs through coordinated cell shape changes and rearrangements [84]. Experimental measurements reveal that epithelial tissues exhibit viscoelastic properties with relaxation times ranging from minutes to hours [85]. During this period, cells undergo neighbor exchanges and apical constriction that dissipate mechanical stress and alter the tissue's reference configuration [86]. The quasi-static elastic approach in Section 2.2 assumes mechanical equilibrium at each growth increment, neglecting the effect of relaxation processes that may occur between or during growth steps.

5.1.2 The Need for Viscoelastic Modeling in Growth Mechanics

The incorporation of viscoelasticity into growth mechanics addresses two fundamental limitations of purely elastic models: the energy dissipation during tissue deformation and the coupling between growth and mechanical relaxation.

First, the energy dissipation during viscoelastic deformation influences the mechanical work required for morphogenesis. The variational formulation in Section 2.2 minimizes the total potential energy $\Pi(\mathbf{u})$ at each growth increment. For viscoelastic materials, mechanical work is partially dissipated rather than perfectly stored as elastic energy. This dissipation affects the energy landscape thereby altering the configurations achieved and changing the morphogenesis trajectory.

Second, morphogenesis processes occur across a spectrum of timescales that often overlap with material relaxation times. The growth timescale τ_g, defined as the characteristic time for significant shape change due to interal growth, varies from hours in rapidly developing embryonic tissues [87] to weeks in slowly expanding plant organs [88]. When τ_g is comparable to the mechanical relaxation time, the assumptions of quasi-static growth and instantaneous elastic response break down. The evolution parameter e introduced in Chapter 2 as the driving parameter of deformation, implicitly represents a quasi-static progression through growth states because we assume in each evolution step equilibrium is always reached. However, the physical time associated with each increment Δe determines whether mechanical relaxation significantly alters the stress field before subsequent growth occurs.

Different experiments have quantified viscoelastic properties in developing tissues across multiple model systems. In *Drosophila* embryos, experiments reveal viscous relaxation times of 10-60 seconds in epithelial tissues [89]. Zebrafish blastoderm exhibits power-law rheology with time-dependent shear moduli spanning several orders of magnitude [90]. Plant meristems show viscoelastic behavior with relaxation times of 1-10 hours, comparable to cell division timescales [88]. These measurements underscore the necessity of incorporating time-dependent material properties into computational models of morphogenesis.

5.1.3 Overview

The mathematical framework developed in this chapter addresses limitations of morphoelastic models and builds upon the existing framework from Chapter 2 by adding dynamics. Using the finite element formulation from Section 2.3 and the numerical solvers of Section 2.4, we provide a unified approach to simulate viscoelastic growth mechanics applicable across the spectrum of biological timescales.

The remainder of this chapter develops a comprehensive viscoelastic framework. Section 5.2 introduces time dependence in deformation to include viscous dissipative effects and derives the corresponding variational principles to solve for displacement. Section 5.3 presents standard rheological models, Kelvin-Voigt, Maxwell, and the Standard Linear Solid model, which are simulated using time-dependent mechanics and FEM, with the results validated against their known behavior. Section 5.4 addresses incorporating growth in viscoelastic modeling, providing an algorithm to evolve time-dependent mechanics in morphing systems. Finally, Section 5.5 demonstrates the framework through biologically relevant applications, incorporating viscoelasticity in growth-induced morphologies.

5.2 Theoretical Framework for Viscoelastic Deformation

The quasi-static framework developed in Chapter 2 provides a robust foundation for modeling growth-induced deformation under the assumption of instantaneous material response. However, biological tissues exhibit inherently time-dependent mechanical behavior that necessitates a fundamentally different theoretical approach. This section develops a comprehensive viscoelastic framework that extends the continuum mechanics formulation to incorporate rate-dependent constitutive relations, internal variables representing microstructural evolution, and thermodynamically consistent dissipation mechanisms.

5.2.1 Incorporating Time into the Deformation Framework

The morphing of the system is represented with the deformation gradient which is related to the displacement field as:

$$\mathbf{F} = \mathbf{I} + \nabla \boldsymbol{u}(\boldsymbol{X}) \tag{5.1}$$

where $\boldsymbol{u}(\boldsymbol{X})$ is the displacement at the material or reference coordinate $\boldsymbol{X}$. The multiplicative decomposition of the deformation gradient introduced in Section 2.2 for growth mechanics is analogous to the framework for incorporating viscoelastic effects. While the elastic-growth decomposition $\mathbf{F} = \mathbf{F}_e \cdot \mathbf{F}_g$ captures the separation between reversible elastic deformation and irreversible growth, viscoelastic materials require an additional mechanism to represent time-dependent, dissipative deformation. This multiplicative decomposition, rather than additive, preserves frame invariance and captures the finite deformation kinematics inherent in morphing systems. We therefore include a viscous deformation gradient, generally through multiplicative decomposition:

$$\mathbf{F} = \mathbf{F}_e \cdot \mathbf{F}_v \tag{5.2}$$

where $\mathbf{F}_v$ represents the viscous component of deformation that evolves according to constitutive relations governing the material's time-dependent response [10]. This decomposition enables the separation of elastic and viscous contributions to deformation, facilitating the formulation of constitutive relations that distinguish between energy storage and dissipation mechanisms. Unlike the growth tensor $\mathbf{F}_g$ which is prescribed based on biological processes, the viscous deformation $\mathbf{F}_v$ emerges from the material's constitutive response to applied stresses and must be determined as part of the solution.

The symmetric part of the viscous velocity gradient($\mathbf{L}_v$) defines the viscous stretching tensor($\mathbf{D}_v$) which plays a central role in formulating energy dissipation:

$$\begin{aligned} \mathbf{L}_v &= \dot{\mathbf{F}}_v \mathbf{F}_v^{-1} \\ \mathbf{D}_v &= \frac{1}{2}(\mathbf{L}_v + \mathbf{L}_v^T) \end{aligned} \tag{5.3}$$

where $\dot{\mathbf{F}}_v$ is the rate of change of viscous deformation tensor with time.

5.2.2 Variational Principles for Viscoelastic Materials

The energy minimization principle that governs equilibrium in elastic materials must be generalized to account for rate-dependent behavior in viscoelastic systems. Instead of minimizing a potential energy functional, viscoelastic materials satisfy a principle of virtual power that balances internal and external power expenditures [91]. For a body occupying domain Ω in the reference configuration, the principle of virtual power states that for any kinematically admissible evolution of internal variable:

$$\delta\mathcal{P}_{int} = \delta\mathcal{P}_{ext} \tag{5.4}$$

where $\delta\mathcal{P}_{int}$ and $\delta\mathcal{P}_{ext}$ represent the internal and external virtual power.

In general, we would need to solve dynamic equilibrium equations from viscoelasticity, which introduces time dependent variables into the system. But, with the use of virtual power, which in our case consists of elastic and viscous parts, we are able to pose the problem as an energy-rate minimization problem. We construct a energy-rate functional $\mathcal{P}$ which is composed of viscous and elastic contributions [92]:

$$\mathcal{P} = \dot{\Pi}(\mathbf{F}_\mathrm{e}, \boldsymbol{u}) + \int_\Omega \phi(\mathbf{D}_v)\, dV \tag{5.5}$$

where $\phi(\mathbf{D}_v)$ is the rate of viscous dissipation and $\dot{\Pi}(\boldsymbol{u})$ is the rate of change of the energy functional from Equation 2.35 with contributions from internal elastic energy and external work:

$$\Pi(\boldsymbol{u}) = \int_\Omega \Psi(\mathbf{F}_\mathrm{e}, \boldsymbol{X})\, dV - W_\mathrm{ext}(\boldsymbol{u}) - W_\mathrm{c}(\boldsymbol{u}) \tag{5.6}$$

The energy rate functional can be expanded as:

$$\mathcal{P} = \int_\Omega \dot{\Psi}(\mathbf{F}_\mathrm{e}, \boldsymbol{X})\, dV + \int_\Omega \phi(\mathbf{D}_v)\, dV - \dot{W}_\mathrm{ext}(\boldsymbol{u}) - \dot{W}_\mathrm{c}(\boldsymbol{u}) \tag{5.7}$$

by taking time derivatives of all the terms contributing to the energy functional, $\dot{\Psi}, \dot{W}_{\text{ext}}, \dot{W}_{\text{c}}$. The constructed minimization problem:

$$\delta \mathcal{P} = 0 \tag{5.8}$$

provides a thermodynamically consistent equation for the evolution of the internal variables, consisting of $\{\boldsymbol{u}, \mathbf{D}_v\}$ or more variables depending on the complexity of the rheological model used for viscoelastic response [93, 94, 92]. This approach extends the quasi-static energy minimization framework Section 2.3.1 to accommodate time-dependent material behavior while maintaining the ability to be solved using numerical methods.

5.2.3 Modular elements for rheological models

The constitutive relations for specific viscoelastic models with complex material behaviors can be systematically constructed from elementary mechanical components, spring and dashpot, that represent fundamental deformation mechanisms. The spring element represents elastic response, storing mechanical energy reversibly, and the dashpot element models purely viscous behavior, dissipating mechanical energy. We specifically define the element properties for eventual application to morphing of biological systems through the framework of energy-rate minimization.

Spring element

We define the spring element with Neo-Hookean constitutive relations to represent elastic response, storing mechanical energy reversibly according to:

$$\Psi(\mathbf{C}_e) := \frac{\mu}{2}(I_1(\mathbf{C}_e) - 3 - 2\ln J_e) + \frac{\lambda}{2}(\ln J_e)^2 \tag{5.9}$$

where $\mathbf{C}_e = \mathbf{F}_e^{\mathrm{T}}\mathbf{F}_e$ is the right Cauchy-Green strain tensor, $I_1(\mathbf{C}_e) = \text{trace}(\mathbf{C}_e)$, $J_e = \det(\mathbf{F}_e) = \sqrt{\det(\mathbf{C}_e)}$, and μ, λ are the Lamé parameters. These parameters relate to the material properties describing elastic response, of Young's modulus E and Poisson's ratio ν

through:

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \quad \mu = \frac{E}{2(1+\nu)}$$

Dashpot element

The dashpot element models purely viscous response, dissipating mechanical energy through internal friction. The rate of viscous dissipation is given by:

$$\phi(\mathbf{D}_v) := \frac{\eta}{2}\mathbf{D}_v : \mathbf{D}_v \tag{5.10}$$

where η is the viscosity parameter and $\mathbf{D}_v$ represents the viscous stretch tensor defined in Equation 5.2.1. In the absence of rigid body rotations, we can express the viscous dissipation rate from Equation 5.10 as:

$$\begin{aligned} &\phi(\mathbf{C}_v, \dot{\mathbf{C}}_v) = \frac{\eta}{2}\mathbf{l}_v : \mathbf{l}_v \\ &\mathbf{l}_v = \frac{1}{2}\dot{\mathbf{C}}_v \cdot \mathbf{C}_v^{-1} \\ &\mathbf{C}_v = \mathbf{F}_v^{\mathrm{T}}\mathbf{F}_v \end{aligned} \tag{5.11}$$

where $\mathbf{F}_v$ is the viscous component of deformation.

Combining elements

The spring and dashpot elements combine through parallel and series arrangements to generate diverse viscoelastic responses. We note that for any type of arrangement, the energy contribution from the elements always add up and the conditions on the deformation contributions from individual elements can vary.

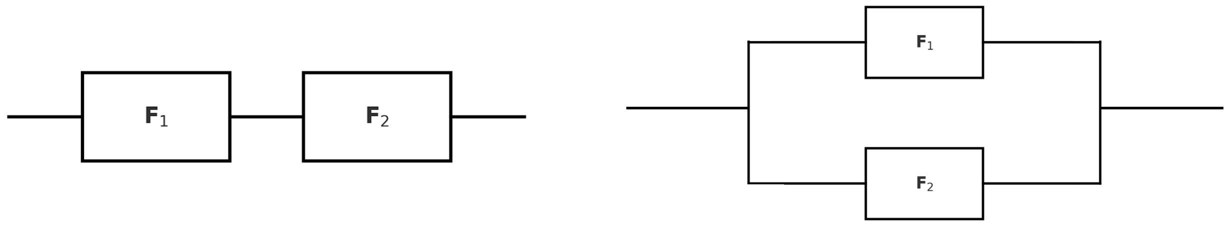


Figure 5.1: Combining rheological elements. **Left**: Series. **Right**: Parallel.

In parallel arrangements, the elements experience identical deformation:

$$\mathbf{F}(\boldsymbol{u}) = \mathbf{F_1} = \mathbf{F_2} \tag{5.12}$$

$$\mathbf{C}(\boldsymbol{u}) = \mathbf{C_1} = \mathbf{C_2} \tag{5.13}$$

where $\boldsymbol{u}$ is the total displacement field, **F** is the deformation tensor, and $\mathbf{C}(\boldsymbol{u})$ is the Cauchy Green tensor. In series arrangements, for large deformations, the deformation from the elements combine:

$$\mathbf{F}(\boldsymbol{u}) = \mathbf{F}_1 \cdot \mathbf{F}_2 \tag{5.14}$$

We note that for series arrangements, the multiplicative decomposition naturally extends from the viscoelastic decomposition established in Section 5.2.1, where $\mathbf{F}_1$ and $\mathbf{F}_2$ represent deformation contributions from the elements combined in series.

The conditions of element combination along with the constitutive relations for individual elements, yield the governing energy equations for composite rheological models. Since we assemble viscoelastic rheologies by combining the spring and dashpot elements, whose energy contributions are defined in Equation 5.9 and 5.10, we can reframe the energy rate functional from Equation 5.7 as:

$$\mathcal{P} = \int_{\Omega} \dot{\Psi}(\mathbf{C}_{\mathrm{e}})\, dV + \int_{\Omega} \phi(\mathbf{C}_v, \dot{\mathbf{C}}_v)\, dV - \dot{W}_{\text{ext}}(\boldsymbol{u}) - \dot{W}_{\mathrm{c}}(\boldsymbol{u}) \tag{5.15}$$

where the internal variables determining the contributing terms are $\{\boldsymbol{u}, \mathbf{C}_{\text{e}}, \mathbf{C}_v, \dot{\mathbf{C}}_v\}$ and we note that additional terms of $\dot{\Psi}$ and ϕ are possible if there exist multiple modular elements:

$$\mathcal{P} = \sum_{i=1}^{n_{\text{spring}}} \int_\Omega \dot{\Psi}_i(\mathbf{C}_{\text{e},i})\, dV + \sum_{j=1}^{n_{\text{dashpot}}} \int_\Omega \phi_j(\mathbf{C}_{\text{v},j}, \dot{\mathbf{C}}_{\text{v},j})\, dV - \dot{W}_{\text{ext}}(\boldsymbol{u}) - \dot{W}_{\text{c}}(\boldsymbol{u}) \tag{5.16}$$

where n_{dashpot} and n_{spring} is the number of dashpot and spring elements respectively. The functions describing the elastic energy and dissipation contributions, Ψ_i and ϕ_j, are also different because they are governed by the material properties of the individual rheological elements.

We simulate visco-elastic deformation by solving for the internal variables that minimize $\mathcal{P}$, and these variables are further controlled by the equations relating them, based on how the spring or dashpot elements are combined, and determining their evolution with time.

5.2.4 Time Integration Framework

The coupled system of equilibrium equations which drives the evolution of internal variables necessitates a robust time integration strategy that maintains numerical stability while accurately capturing the material's time-dependent response.

We assume all our internal variables are evolving with time for a visco-elastic material where the stress response is time dependent. We use the Finite Element Method as described in Section 2.3 to solve for the internal variables by minimizing the energy rate functional, instead of total energy that is used for elastic models. Since the internal variables are dynamic, we discretize the energy rate functional from Equation 5.15 over time and solve for the unknown internal variables at each time step. For the state at the "n-th" time step after updating time $t^n = t^{n-1} + \Delta t$, we have the energy rate to be minimized:

$$\mathcal{P} = \int_\Omega \dot{\Psi}^n\, dV + \int_\Omega \phi^n\, dV - \dot{W}^n_{\text{ext}} - \dot{W}^n_{\text{c}} \tag{5.17}$$

where, $\{\mathbf{C}_{\mathrm{e}}^n, \mathbf{C}_{\mathrm{v}}^n, \dot{\mathbf{C}}_{\mathrm{v}}^n, \boldsymbol{u}^n\}$ are values of the internal variables at the current timestep and $\{\dot{\Psi}^n, \phi^n, \dot{W}_{\mathrm{ext}}^n, \dot{W}_{\mathrm{c}}^n\}$ are the terms evaluated at the current time step using backward Euler scheme [48]:

$$\begin{aligned}
\dot{\Psi}^n &= \frac{1}{\Delta t}(\Psi(\mathbf{C}_e^n) - \Psi(\mathbf{C}_e^{n-1})) \\
\phi^n &= \phi(\mathbf{C}_{\mathrm{v}}^n, \dot{\mathbf{C}}_{\mathrm{v}}^n), \quad \dot{\mathbf{C}}_{\mathrm{v}}^n = \frac{1}{\Delta t}(\mathbf{C}_{\mathrm{v}}^n - \mathbf{C}_{\mathrm{v}}^{n-1}) \\
\dot{W}_{\mathrm{ext}}^n &= \frac{1}{\Delta t}(W_{\mathrm{ext}}(\boldsymbol{u}^n) - W_{\mathrm{ext}}(\boldsymbol{u}^{n-1})) \\
\dot{W}_{\mathrm{c}}^n &= \frac{1}{\Delta t}(W_{\mathrm{c}}(\boldsymbol{u}^n) - W_{\mathrm{c}}(\boldsymbol{u}^{n-1}))
\end{aligned} \tag{5.18}$$

where $\{\mathbf{C}_{\mathrm{e}}^{n-1}, \mathbf{C}_{\mathrm{v}}^{n-1}, \dot{\mathbf{C}}_{\mathrm{v}}^{n-1}, \boldsymbol{u}^{n-1}\}$ are known values from the solution at the previous step. Combining the system of equations from time discretization(Equations 5.18) and energy-rate minimization Equation 5.8, we pose the variational problem to be numerically solved using FEM at time t^n:

$$\delta\mathcal{P}(\mathbf{C}_{\mathrm{e}}^n, \mathbf{C}_{\mathrm{v}}^n, \boldsymbol{u}^n) = 0 \tag{5.19}$$

where the internal variables required to describe the current state $\{\boldsymbol{u}^n, \mathbf{C}_{\mathrm{v}}^n, \mathbf{C}_{\mathrm{e}}^n\}$ are solved for. The resulting nonlinear system of equations is solved using the solver algorithms from Section 2.4 while incrementing time as shown in Algorithm 6. The selection of time step Δt involves

Algorithm 6 Time Integration for Viscoelastic Deformation

```
1:  **Input:** Initial displacement $\mathbf{u}^0$, time step $\Delta t$, final time $t_{final}$, tolerance $\epsilon$
2:  **Initialize:** $t^0 = 0, n = 0$
3:  Compute initial deformation and other internal variables: $\mathbf{F}^0 = \mathbf{I} + \nabla\mathbf{u}^0$, $\mathbf{C}_e^0$, $\mathbf{C}_v^0 = \mathbf{I}$
4:  Store initial state: $\{\mathbf{u}^0, \mathbf{C}_e^0, \mathbf{C}_v^0\}$
5:  **while** $t^n < t_{\text{final}}$ **do**
6:      $n = n + 1, t^n = t^{n-1} + \Delta t$
7:      Initialize displacement and other internal variables: $\mathbf{u}^{n,0} = \mathbf{u}^{n-1}$, $\mathbf{C}_v^{n,0} = \mathbf{C}_v^{n-1}$
8:      Solve nonlinear system $\delta\mathcal{P}(\mathbf{u}^n, \mathbf{C}_v^n) = 0$ to obtain converged $\{\mathbf{u}^n, \mathbf{C}_v^n\}$
9:      Update other variables based on the rheological model: $\mathbf{F}^n = \mathbf{I} + \nabla\mathbf{u}^n$, $\mathbf{C}$, $\mathbf{C}_e^n$, $\dot{\mathbf{C}}_v^n$
10:     **Post-process and store results**
11: **end while**
12: **Output:** Time history of displacement $\{\mathbf{u}^n, t^n\}$
```

balancing accuracy and computational efficiency. The ratio of the time step to the characteristic relaxation time, $\Delta t/\tau$, determines the resolution of the transient response for the viscoelastic material.

In summary, this theoretical framework extends the elastic formulation of Chapter 2 to viscoelastic materials through: (i) multiplicative decomposition incorporating viscous deformation, (ii) energy rate minimization replacing energy minimization, and (iii) time discretization of evolution equations. We introduced constitutive relations for spring and dashpot elements used and their modular assembly provides flexibility in representing diverse material behaviors. The following section demonstrates the application of this framework to classical rheological models, showing how specific arrangements of these elements capture viscoelastic phenomena.

5.3 Rheological Models for Viscoelastic Materials

The modular components introduced in Section 5.2.3 provide the building blocks for constructing rheological models that capture the diverse viscoelastic behaviors observed in biological tissues. This section presents three fundamental models: Kelvin-Voigt, Maxwell, and Standard Linear Solid, each representing different arrangements of springs and dashpots that yield distinct mechanical responses. These models serve as the foundation for understanding complex tissue behaviors and provide analytical benchmarks for validating our computational framework for viscoelasticity. We demonstrate the implementation of each model through uniaxial creep tests, which provide a standard benchmark for characterizing viscoelastic material response and validating our finite deformation framework against established analytical solutions.

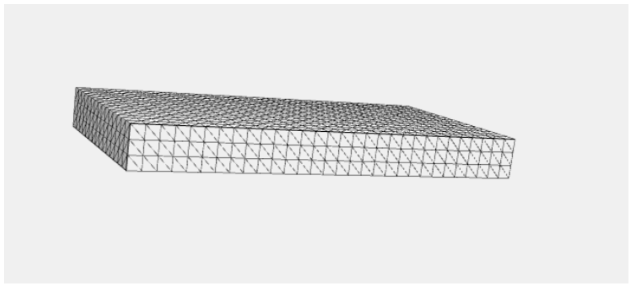


Figure 5.2: 3D mesh for cuboidal rod with x-axis along its length, which is the largest dimension.

5.3.1 Creep Test

In a uniaxial creep test, a specimen is subjected to a constant stress while the resulting strain is measured over time. This fundamental characterization reveals the time-dependent deformation behavior unique to each rheological model. For small strains where $\|\nabla\mathbf{u}\| \ll 1$, the nonlinear finite deformation framework converges with linear viscoelasticity, enabling direct comparison with classical analytical solutions derived under assumptions of infinitesimal strains and linear constitutive relations.

For simulating the uniaxial creep test, we assume the object is a cuboidal rod which is stretched by externally applied stress along its length (Figure 5.2). We model the object using 3D mesh and 3D displacements, using the modular rheological elements to determine the material response. We setup the problem to minimize the general energy-rate functional:

$$\mathcal{P} = \sum_{i=1}^{n_{\text{spring}}} \int_{\Omega} \dot{\Psi}_i(\mathbf{C}_{\text{e},i})\, dV + \sum_{j=1}^{n_{\text{dashpot}}} \int_{\Omega} \phi_j(\mathbf{C}_{\text{v},j}, \dot{\mathbf{C}}_{\text{v},j})\, dV - \dot{W}_{\text{ext}}(\boldsymbol{u}) - \dot{W}_{\text{c}}(\boldsymbol{u}) \tag{5.20}$$

where n_{spring} and n_{dashpot} denote the number of spring and dashpot elements in the rheological model, respectively. The creep test also includes external traction acting on the boundary (Section 2.2.2) and we use Lagrange multipliers to constrain all rigid body motion(Section 2.2.3), so the resulting virtual work terms are as follows:

$$W_{ext}(\boldsymbol{u}) = \int_{\partial\Omega} \boldsymbol{t} \cdot \boldsymbol{u} \, dA \tag{5.21}$$

$$W_c(\boldsymbol{u}) = \sum_{i=1}^{3} \lambda_{\text{trans, i}} \left[\int_{\Omega} \boldsymbol{u}(\boldsymbol{X}) \cdot \boldsymbol{e}_i \, dV \right] + \sum_{i=1}^{3} \lambda_{\text{rot,i}} \int_{\Omega} \boldsymbol{X} \times \boldsymbol{u} \cdot \boldsymbol{e}_i \, dV \tag{5.22}$$

here, $\partial\Omega$ is the boundary of the mesh and for uniaxial external stress we set the tractions to be:

$$\boldsymbol{t} = \boldsymbol{\sigma}_{ext} \cdot \boldsymbol{n} \tag{5.23}$$

$$\boldsymbol{\sigma}_{ext} = \begin{bmatrix} \sigma_{xx} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{5.24}$$

where $\boldsymbol{n}$ is the facet normal on the mesh and this results in a uniform external stress acting along the x-axis on the left and right faces of the bar.

The creep test is conducted by implementing the uniaxial external stress as a step function:

$$\sigma_{xx} = \begin{cases} 0 \text{ Pa}, t \leq 0 \\ 1 \text{ Pa}, 0 < t < 1 \\ 0 \text{ Pa}, t \geq 1 \end{cases} \tag{5.25}$$

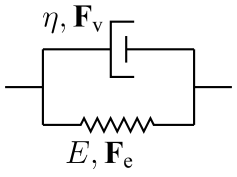
where σ_{xx} is the scalar value of the external stress acting along the x-axis. For stepping through time we use a time increment of $\Delta t = 0.01$ over 200 timesteps which is chosen to provide high-resolution sampling of the time-dependent response. We choose other material parameters of E, ν, η for individual elements so the maximum strain is 1%, thus remaining in the limit where the analytical viscoelastic responses serve as a valid benchmark for our simulation.

5.3.2 Kelvin-Voigt Model

The Kelvin-Voigt model consists of a spring and dashpot arranged in parallel and in this configuration, both elements experience identical deformation while their stresses sum:

$$\mathbf{F} = \mathbf{F}_e = \mathbf{F}_v, \quad \boldsymbol{\sigma} = \boldsymbol{\sigma}_e + \boldsymbol{\sigma}_v \tag{5.26}$$

where $\boldsymbol{\sigma}, \boldsymbol{\sigma}_e, \boldsymbol{\sigma}_v$ are the total stress, stress contribution from the spring and dashpot element respectively.



Implementing the simulation

For numerical implementation within the framework of Section 5.2, the Kelvin-Voigt model requires no additional internal variables along with displacement $\boldsymbol{u}$ since the viscous deformation is directly coupled to the total deformation. So, the deformation variables are:

$$\mathbf{F} = \mathbf{I} + \nabla \boldsymbol{u} \tag{5.27}$$

$$\mathbf{C} = \mathbf{F}^\text{T}\mathbf{F} \tag{5.28}$$

$$\mathbf{C}_\text{e} = \mathbf{C}_\text{v} = \mathbf{C} \tag{5.29}$$

and the energy rate functional from Equation 5.15 simplifies to:

$$\mathcal{P}_\text{kv} = \int_\Omega \dot{\Psi}(\mathbf{C})\, dV + \int_\Omega \phi(\mathbf{C})\, dV - \dot{W}_\text{ext}(\boldsymbol{u}) - \dot{W}_\text{c}(\boldsymbol{u}) \tag{5.30}$$

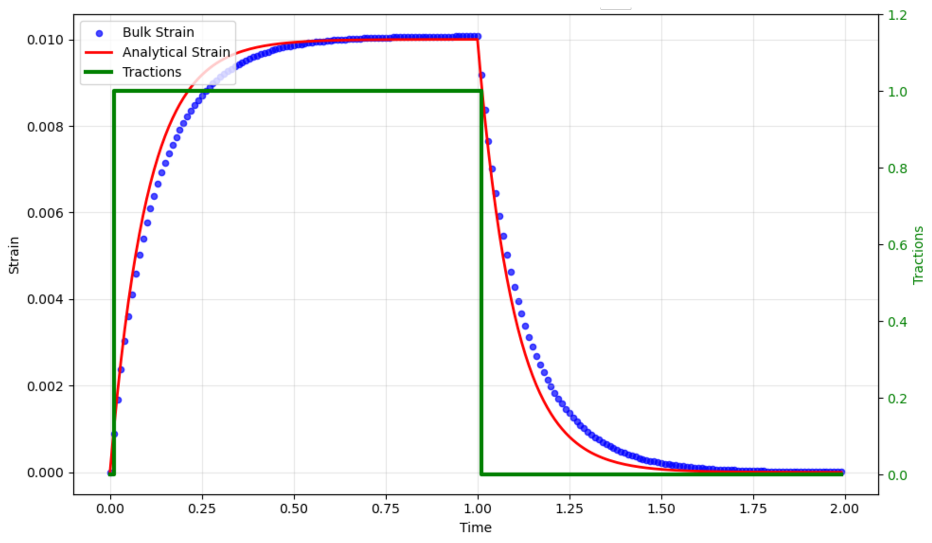


Figure 5.3: **Kelvin-Voigt model:** Plot showing time-dependent response to the creep test with step-function for loading and unloading through tractions along the material length(right on y-axis). The strain response(left y-axis) from the analytical linear model(red) and the simulation(blue) are plotted. The material properties(in SI units) used are $\eta = 10$ as the viscosity constant, $E = 100$ as Young's modulus, and $\nu = 0.3$ as Poisson's ratio(only for the simulation).

Validation with analytical benchmarks

We implement a simulation for the creep test using this rheological model, as described in Section 5.3.1 and use Equation 5.30 to provide the energy-rate functional to be minimized. For the analytical benchmark, we know that under uniaxial creep conditions with constant applied stress σ_0, the strain evolution in Kelvin-Voigt model follows:

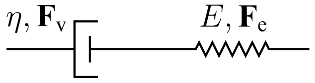
$$\varepsilon(t) = \frac{\sigma_0}{E}\left(1 - e^{-t/\tau}\right) \tag{5.31}$$

where $\tau = \eta/E$ represents the characteristic relaxation time, the timescale over which the material transitions from initial stiffness to its equilibrium response. The model approaches the elastic solution $\varepsilon_\infty = \sigma_0/E$ asymptotically, capturing the delayed elastic behavior observed in many biological tissues [83]. We see from Figure 5.3 that the simulation results agree with the analytical relation for the creep test with 1% strain.

5.3.3 Maxwell Model

The Maxwell model employs a spring and dashpot in series and the series arrangement enforces stress continuity while deformations combine. This multiplicative decomposition aligns with the framework established in Equation 5.2.

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}_e = \boldsymbol{\sigma}_v, \quad \mathbf{F} = \mathbf{F}_e \cdot \mathbf{F}_v \tag{5.32}$$



Implementing the simulation

Numerical implementation requires tracking additional internal variables along with $\boldsymbol{u}$, because we need additional information to complement Equation 5.32 to obtain all the values for $\{\boldsymbol{u}, \mathbf{C}_v, \mathbf{C}_e\}$ required to determine the energy-rate functional for the Maxwell model:

$$\mathcal{P}_\text{maxwell} = \int_\Omega \dot{\Psi}(\mathbf{C}_e)\, dV + \int_\Omega \phi(\mathbf{C}_v)\, dV - \dot{W}_\text{ext}(\boldsymbol{u}) - \dot{W}_\text{c}(\boldsymbol{u}) \tag{5.33}$$

where, we also require the value of $\mathbf{C}_e$ and $\mathbf{C}_v$ to determine the elastic energy contribution and viscous dissipation. Since we have an equation relating the variables $\{\boldsymbol{u}, \mathbf{C}_e, \mathbf{C}_v\}$ from the series combination of deformations $\mathbf{F} = \mathbf{F}_e \cdot \mathbf{F}_v$, we choose to include only one additional internal variable to be tracked and the unknown functions to be solved for are $\{\boldsymbol{u}, \mathbf{C}_v\}$. For determining the elastic energy contribution in Equation 5.33 from the internal variables $\{\boldsymbol{u}, \mathbf{C}_v\}$, we use the following relations:

$$
\begin{aligned}
&\mathbf{F} = \mathbf{I} + \nabla \boldsymbol{u} \\
&\mathbf{C} = \mathbf{F}^{\mathrm{T}}\mathbf{F},\ \mathbf{C}_v = \mathbf{F}_v^{\mathrm{T}}\mathbf{F}_v,\ \mathbf{C}_e = \mathbf{F}_e^{\mathrm{T}}\mathbf{F}_e \\
&\mathbf{F}_e = \mathbf{F} \cdot \mathbf{F}_v^{-1} \\
&\mathbf{C} \cdot \mathbf{C}_v^{-1} = \mathbf{F}^{\mathrm{T}}\mathbf{F} \cdot (\mathbf{F}_v^{\mathrm{T}}\mathbf{F}_v)^{-1} = (\mathbf{F}_e \cdot \mathbf{F}_v)^{\mathrm{T}}(\mathbf{F}_e \cdot \mathbf{F}_v) \cdot (\mathbf{F}_v^{\mathrm{T}}\mathbf{F}_v)^{-1} = \mathbf{F}_v^{\mathrm{T}}\mathbf{C}_e(\mathbf{F}_v^{\mathrm{T}})^{-1} \\
&\mathrm{tr}(\mathbf{C}_e) = \mathrm{tr}(\mathbf{F}_v^{\mathrm{T}}\mathbf{C}_e(\mathbf{F}_v^{\mathrm{T}})^{-1}) = \mathrm{tr}(\mathbf{C} \cdot \mathbf{C}_v^{-1}),\ \det(\mathbf{C}_e) = \det(\mathbf{F}_v^{\mathrm{T}}\mathbf{C}_e(\mathbf{F}_v^{\mathrm{T}})^{-1}) = \det(\mathbf{C} \cdot \mathbf{C}_v^{-1})
\end{aligned}
\tag{5.34}
$$

where tr$(\mathbf{C}_e)$ and det$(\mathbf{C}_e)$ can completely determine the elastic energy contribution through the Neo-Hookean constitutive relation from Equation 5.9 and we thus derive $\Psi(\mathbf{C}_e) = \Psi(\mathbf{C} \cdot \mathbf{C}_v^{-1})$. So, the energy-rate function to be minimized to evolve the internal variables $\{\boldsymbol{u}, \mathbf{C}_v\}$ at each time step is given below.

$$
\mathcal{P}_{\text{maxwell}}(\boldsymbol{u}, \mathbf{C}_v) = \int_\Omega \dot{\Psi}(\mathbf{C} \cdot \mathbf{C}_v^{-1})\, dV + \int_\Omega \phi(\mathbf{C}_v)\, dV - \dot{W}_{\text{ext}}(\boldsymbol{u}) - \dot{W}_{\text{c}}(\boldsymbol{u}) \tag{5.35}
$$

We also note that the unknown functions to be solved for $(\boldsymbol{u}, \mathbf{C}_v)$, include the symmetric tensor $\mathbf{C}_v = \mathbf{F}_v^{\mathrm{T}}\mathbf{F}_v$. To make the simulation computationally efficient we remove redundant terms of the tensor $\mathbf{C}_v$ from the unknown function to reduce the degrees of freedom in the simulation. The resulting unknown function is of the form $(\boldsymbol{u}, C_{v,i})$ where $C_{v,i}$ are scalars corresponding to the unique terms of the symmetric tensor $\mathbf{C}_v$ and while running the simulation the tensor $\mathbf{C}_v$ is re-constructed from $C_{v,i}$.

Validation with analytical benchmarks

The constitutive relation for the Maxwell model is governed by the differential equation:

$$
\boldsymbol{\sigma} + \frac{\eta}{E}\dot{\boldsymbol{\sigma}} = \eta\dot{\boldsymbol{\varepsilon}} \tag{5.36}
$$

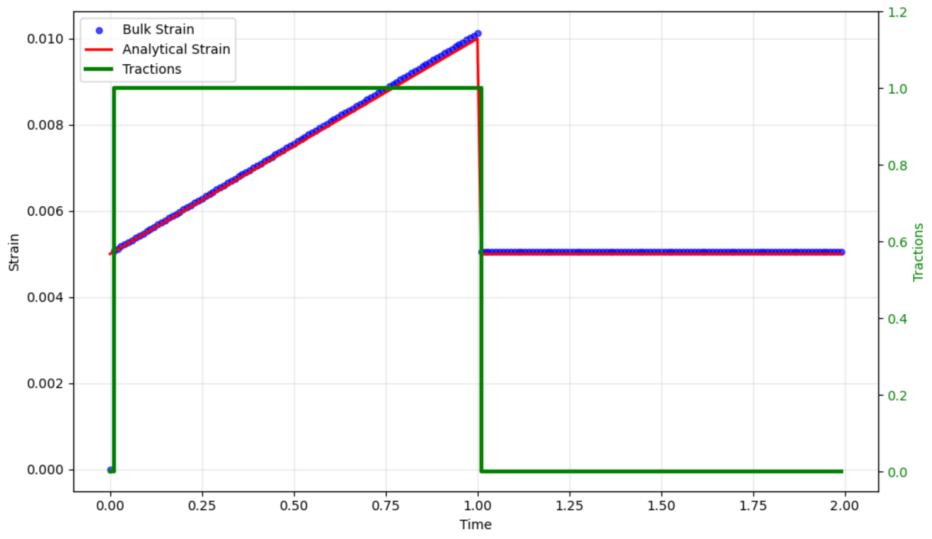


Figure 5.4: **Maxwell model:** Plot showing time-dependent response to the creep test with step-function for loading and unloading through tractions along the material length(right on y-axis). The strain response(left y-axis) from the analytical linear model(red) and the simulation(blue) are plotted. The material properties(in SI units) used are $\eta = 200$ as the viscosity constant, $E = 200$ as Young's modulus, and $\nu = 0.3$ as Poisson's ratio(only for the simulation).

Under creep loading, the Maxwell model exhibits unbounded deformation:

$$\varepsilon(t) = \frac{\sigma_0}{E} + \frac{\sigma_0}{\eta}t \tag{5.37}$$

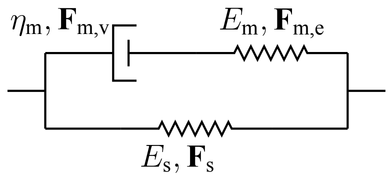
comprising an instantaneous elastic response followed by steady viscous flow [10]. The unbounded creep reflects irreversible flow, exhibiting permanent deformation after unloading, $t > 1$. We implement a simulation for the creep test (Section 5.3.1) using the maxwell model of viscoelasticity and use Equation 5.35 to provide the energy-rate functional to be minimized. We see from Figure 5.4 that the simulation results align with the analytical relation for the creep test.

5.3.4 Standard Linear Solid (SLS) Model

The Standard Linear Solid combines the bounded deformation of the Kelvin-Voigt model with the stress relaxation capability of the Maxwell model through a three-element configuration, a spring in parallel with a Maxwell element. The total stress comprises contributions from the parallel branches of spring and the Maxwell element, but the deformation in the branches is equal :

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}_s + \boldsymbol{\sigma}_m, \quad \mathbf{F} = \mathbf{F}_s = \mathbf{F}_m \tag{5.38}$$

where $\boldsymbol{\sigma}_s$ represents the spring stress response and $\boldsymbol{\sigma}_m$ stress in the Maxwell branch.


$\eta_\mathrm{m}, \mathbf{F}_\mathrm{m,v}$
$E_\mathrm{m}, \mathbf{F}_\mathrm{m,e}$
$E_\mathrm{s}, \mathbf{F}_\mathrm{s}$

Implementing the simulation

For finite deformations, we decompose the total deformation as:

$$\mathbf{F}_m = \mathbf{F}_{m,e} \cdot \mathbf{F}_{m,v} \tag{5.39}$$

where subscript m denotes the Maxwell branch, $\mathbf{F}_{m,e}$ and $\mathbf{F}_{m,v}$ are the elastic and viscous contributions to deformation within the branch, and $\mathbf{F}_s$ is the deformation of the spring element in parallel. The energy rate functional is constructed to include the contributions from the two elastic spring elements and the viscous dashpot element:

$$\mathcal{P}_\text{SLS} = \int_\Omega \left[\dot{\Psi}_s(\mathbf{C}_s) + \dot{\Psi}_{m,e}(\mathbf{C}_{m,e})\right] dV + \int_\Omega \phi(\mathbf{C}_{m,v})\, dV - \dot{W}_\text{ext}(\boldsymbol{u}) - \dot{W}_\text{c}(\boldsymbol{u}) \tag{5.40}$$

where, we can express all the contributing terms from the reduced set of internal variables $\{\boldsymbol{u}, \mathbf{C}_{m,v}\}$ using the deformation from Equations 5.38 and 5.39 by following a derivation similar to that described in Equations 5.34.

$$\mathbf{F} = \mathbf{I} + \nabla \boldsymbol{u},\ \mathbf{F}_m = \mathbf{F},\ \mathbf{F}_s = \mathbf{F} \tag{5.41}$$

$$\mathbf{C} = \mathbf{F}^{\mathrm{T}}\mathbf{F} \tag{5.42}$$

$$\mathbf{C}_s = \mathbf{C} = \mathbf{C}_m \tag{5.43}$$

$$\mathbf{F}_{m,e} = \mathbf{F}_m \cdot \mathbf{F}_{m,v}^{-1} = \mathbf{F} \cdot \mathbf{F}_{m,v}^{-1} \tag{5.44}$$

$$\mathbf{C} \cdot \mathbf{C}_{m,v}^{-1} = \mathbf{C}_m \cdot \mathbf{C}_{m,v}^{-1} = (\mathbf{F}_{m,e} \cdot \mathbf{F}_{m,v})^{\mathrm{T}}(\mathbf{F}_{m,e} \cdot \mathbf{F}_{m,v}) \cdot (\mathbf{F}_{m,v}^{\mathrm{T}}\mathbf{F}_{m,v})^{-1} = \mathbf{F}_{m,v}^{\mathrm{T}}\mathbf{C}_{m,e}(\mathbf{F}_{m,v}^{\mathrm{T}})^{-1} \tag{5.45}$$

$$\mathrm{tr}(\mathbf{C}_{m,e}) = \mathrm{tr}(\mathbf{F}_{m,v}^{\mathrm{T}}\mathbf{C}_{m,e}(\mathbf{F}_{m,v}^{\mathrm{T}})^{-1}) = \mathrm{tr}(\mathbf{C} \cdot \mathbf{C}_{m,v}^{-1}),\ \det(\mathbf{C}_{m,e}) = \det(\mathbf{C} \cdot \mathbf{C}_{m,v}^{-1}) \tag{5.46}$$

$$\Psi_{m,e}(\mathbf{C}_{m,e}) = \Psi_{m,e}(\mathbf{C} \cdot \mathbf{C}_{m,v}^{-1}) \tag{5.47}$$

Here, $\Psi(\mathbf{C}_{m,e})$ is the elastic energy contribution from the spring in the Maxwell element which can be calculated from $\mathbf{C} \cdot \mathbf{C}_{m,v}^{-1}$ since the spring element is Neo-hookean which is calculated using invariants of $\mathbf{C}_{m,e}$. Thus, we can derive the energy rate function from the internal variables as:

$$\mathcal{P}_{\mathrm{SLS}}(\boldsymbol{u}, \mathbf{C}_{m,v}) = \int_\Omega \left[\dot{\Psi}_s(\mathbf{C}) + \dot{\Psi}_{m,e}(\mathbf{C} \cdot \mathbf{C}_{m,v}^{-1})\right] dV + \int_\Omega \phi(\mathbf{C}_{m,v})\, dV - \dot{W}_{\mathrm{ext}}(\boldsymbol{u}) - \dot{W}_{\mathrm{c}}(\boldsymbol{u}) \tag{5.48}$$

where, the unknown function $(\boldsymbol{u}, \mathbf{C}_{m,v})$ when implemented is reduced in dimension by removing the redundant terms of the symmetric tensor $\mathbf{C}_{m,v}$, similar to the simulation of the Maxwell model.

Validation with analytical benchmarks

In the small strain limit, the theoretical constitutive relation for the SLS model simplifies to:

$$\sigma(t) + \frac{\eta_m}{E_m}\frac{d\sigma(t)}{dt} = \eta_m \frac{d\epsilon(t)}{dt} + E_s\epsilon(t) + \frac{\eta_m E_s}{E_m}\frac{d\epsilon(t)}{dt} \tag{5.49}$$

where E_s and E_m are the elastic moduli of the parallel and Maxwell springs, respectively. The analytical creep response of the SLS model exhibits bounded deformation with characteristic two-stage behavior:

$$\varepsilon(t) = \sigma_0 \left[\frac{1}{E_p + E_m} + \frac{E_m}{E_p(E_p + E_m)} \left(1 - e^{-t/\tau}\right) \right] \tag{5.50}$$

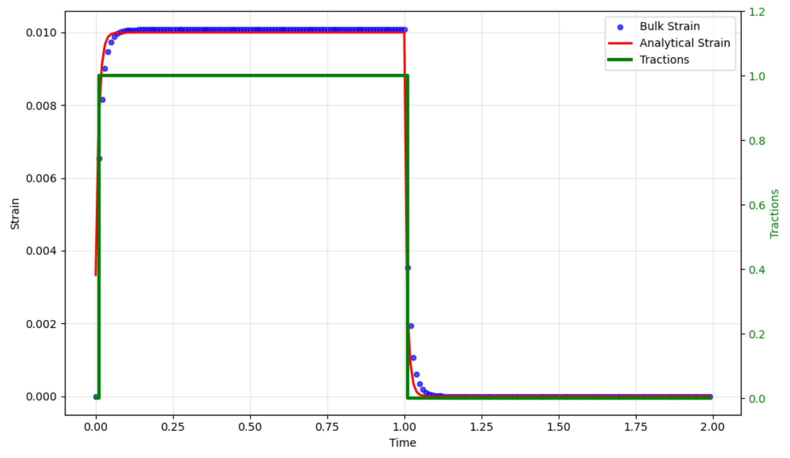
where $\tau = \eta/E_m$ is the relaxation time. The two-stage response captures both instantaneous elastic deformation and time-dependent relaxation to a new equilibrium state. For the creep test, we follow the approach of Section 5.2.4 to solve for both displacement **u** and viscous deformation $\mathbf{C}_{m,v}$ simultaneously:

$$\delta \mathcal{P}_{SLS}(\mathbf{u}^n, \mathbf{C}^n_{m,v}) = 0 \tag{5.51}$$

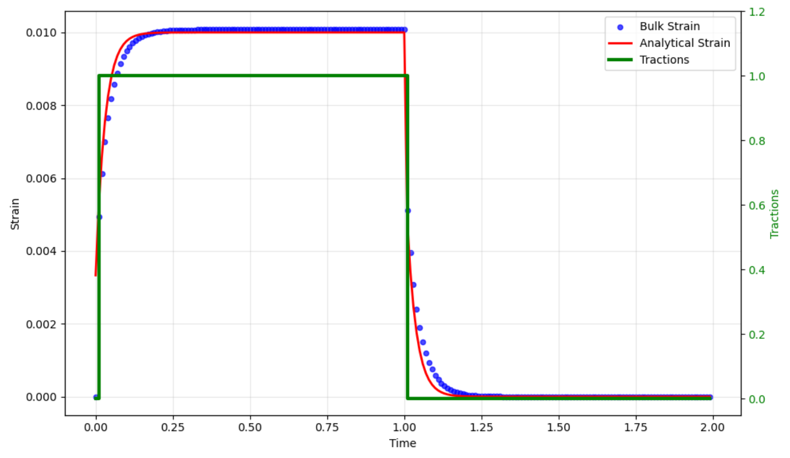
We see from Figure 5.5 that the simulation results align with the analytical relation for the creep test.

The three models presented—Kelvin-Voigt, Maxwell, and SLS—span the spectrum of viscoelastic behaviors from solids to fluids, with the SLS providing a versatile framework for biological tissues. We derive the specific energy-rate minimization problems to be solved numerically for each of the rheologies based on the connections between the elements, in a way that makes modeling large viscoelastic deformations feasible through our finite element framework. Simulation involves iteratively solving the non-linear system with time stepping and choosing Δt based on the relaxation timescales to obtain good resolution of the transient response. The successful validation of the simulation against analytical benchmarks demonstrates that our energy-rate minimization framework captures the physics of viscoelastic deformation while maintaining the generality needed for finite deformations and complex geometries. Our computational model capturing viscoelastic dynamics enables us to incorporate growth in the following section, examining how combining viscoelastic relaxation and morphogenetic driving forces shapes the final tissue architecture.

A



B



C

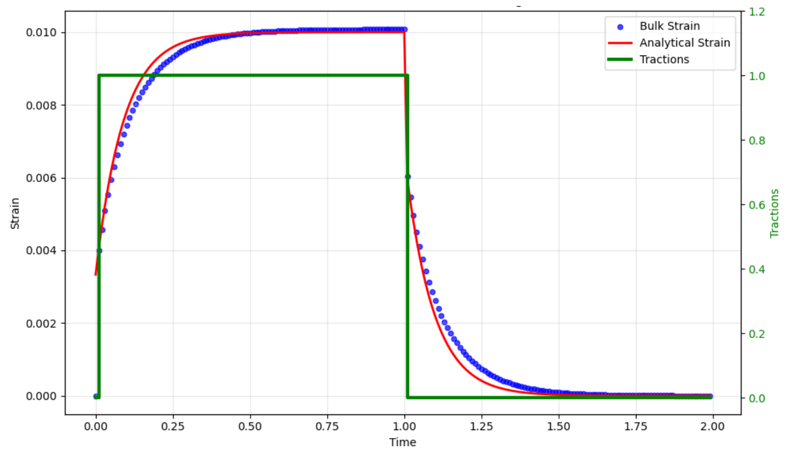


Figure 5.5: **SLS model**: Plots showing creep test responses for SLS models with different relaxation times. The tractions are implemented as a step-function for loading and unloading along the material length(right on y-axis). The strain response(left y-axis) from the analytical linear model(red) and the simulation(blue) are plotted. The common material properties(in SI units) used are $\eta = 10$ as the viscosity constant, $E_\mathrm{m} = 200$, $E_\mathrm{s} = 100$ as Young's modulus of the spring elements, and $\nu_s = \nu_m = 0.3$ as Poisson's ratio(only for the simulation). The relaxation times of the SLS models are increased from A-C by increasing the viscosity coefficient η_m. **(A)** η_m = 2/3, **(B)** η_m = 2, **(C)** η_m = 6.

5.4 Viscoelastic Growth

The frameworks developed for viscoelastic material response (Section 5.2) and morphoelastic growth (Section 2.2) provide the building blocks for modeling growth-induced deformation in tissues with time-dependent mechanical properties. This section develops a unified framework that captures the interplay between growth processes and viscoelastic relaxation, addressing how growth and material relaxation interact in the simulation to determine final morphologies.

5.4.1 Multiplicative Decomposition for Viscoelastic Growth

The morphoelastic framework decomposes the total deformation gradient as:

$$\mathbf{F} = \mathbf{F}_e \mathbf{F}_g \tag{5.52}$$

where $\mathbf{F}_e$ represents elastic deformation and $\mathbf{F}_g$ prescribes growth. For viscoelastic materials, we extend this decomposition by recognizing that the elastic response $\mathbf{F}_e$ must be replaced by the viscoelastic response characterized by the chosen rheological model.

We choose the Standard Linear Solid model, which provides a general framework encompassing both stress relaxation and bounded deformation, and the total deformation becomes:

$$\mathbf{F} = \mathbf{F}_{\text{sls}} \mathbf{F}_g \tag{5.53}$$

where $\mathbf{F}_{\text{sls}}$ represents the viscoelastic deformation that would occur from an intermediate configuration after growth to the current configuration, similar to the morphoelastic theory [3].

Within the SLS framework, the viscoelastic deformation further decomposes according to the rheological model where spring and Maxwell elements are parallel(Section 5.3.4):

$$\mathbf{F}_{\text{sls}} = \mathbf{F}_s = \mathbf{F}_m = \mathbf{F}\mathbf{F}_g^{-1} \tag{5.54}$$

$$\mathbf{F}_m = \mathbf{F}_{m,e}\mathbf{F}_{m,v} \tag{5.55}$$

where the subscripts s and m denote the spring and Maxwell branches respectively. The elastic deformation in the Maxwell branch becomes:

$$\mathbf{F}_{m,e} = \mathbf{F}_m\mathbf{F}_{m,v}^{-1} = \mathbf{F}\mathbf{F}_g^{-1}\mathbf{F}_{m,v}^{-1} \tag{5.56}$$

5.4.2 Energy Rate Functional for Growing Viscoelastic Materials

The energy rate functional must account for growth-induced changes in the reference configuration. Following the modeling for SLS material from Section 5.3.4, we construct the energy rate function to be minimized during growth as:

$$\mathcal{P}_{\text{SLS,g}}(\mathbf{u}, \mathbf{C}_{m,v}) = \int_\Omega \left[\dot{\Psi}_s(\mathbf{C}_{\text{sls}}) + \dot{\Psi}_{m,e}(\mathbf{C}_{\text{sls}}\mathbf{C}_{m,v}^{-1})\right] dV + \int_\Omega \phi(\mathbf{C}_{m,v}, \dot{\mathbf{C}}_{m,v})\, dV - \dot{W}_{\text{ext}}(\mathbf{u}) - \dot{W}_{\text{c}}(\mathbf{u}) \tag{5.57}$$

We derive an expression for $\mathbf{C}_{\text{sls}}$ in terms of the internal variables $\{\mathbf{u}, \mathbf{C}_{m,v}\}$ and the prescribed growth tensor $\mathbf{F}_g$:

$$\mathbf{F} = \mathbf{I} + \nabla\mathbf{u} \tag{5.58}$$

$$\mathbf{C} = \mathbf{F}^T\mathbf{F} \tag{5.59}$$

$$\mathbf{C}_{\text{sls}} = \mathbf{F}_{\text{sls}}^T\mathbf{F}_{\text{sls}} = (\mathbf{F}\mathbf{F}_g^{-1})^T(\mathbf{F}\mathbf{F}_g^{-1}) = \mathbf{F}_g^{-T}\mathbf{C}\mathbf{F}_g^{-1} \tag{5.60}$$

Substituting into Equation 5.57 we have an energy-rate functional which can be solved for

$\{\mathbf{u}, \mathbf{C}_{m,v}\}$ at each timestep with prescribed $\mathbf{F}_g$:

$$\mathcal{P}_{\text{SLS,g}}(\mathbf{u}, \mathbf{C}_{m,v}) = \int_\Omega \left[\dot{\Psi}_s(\mathbf{F}_g^{-T}\mathbf{C}\mathbf{F}_g^{-1}) + \dot{\Psi}_{m,e}(\mathbf{F}_g^{-T}\mathbf{C}\mathbf{F}_g^{-1}\mathbf{C}_{m,v}^{-1})\right] dV + \int_\Omega \phi(\mathbf{C}_{m,v}, \dot{\mathbf{C}}_{m,v})\, dV - \dot{W}_{\text{ext}}(\mathbf{u}) - \dot{W}_{\text{c}}(\mathbf{u}) \quad (5.61)$$

5.4.3 Numerical Implementation

The coupling between growth and viscoelastic relaxation requires solving the time-dependent evolution of both displacement and internal variables while updating the growth tensor. At each time step, we have:

$$\text{Set} \quad \mathbf{F}_g^n = \mathbf{F}_g(t^n)$$
$$\text{Solve} \quad \delta\mathcal{P}_{\text{SLS,g}}(\mathbf{u}^n, \mathbf{C}_{m,v}^n) = 0$$

We extend Algorithm 6 to include evolution of the growth tensor resulting in Algorithm 7 which evolves growth-driven viscoelastic deformation by stepping through time. The selection of time step Δt must resolve both the viscoelastic relaxation time $\tau_v = \eta/E$ and change of growth tensor $\mathbf{F}_g$. We use the following guidelines to choose Δt, N_{steps} value:

Algorithm 7 Viscoelastic Growth Evolution

```
1: Initialize: $\mathbf{u}^0 = \mathbf{0}$, $\mathbf{C}_{m,v}^0 = \mathbf{I}$, $\mathbf{F}_g^0 = \mathbf{I}$
2: for $n = 1$ to $N_{\text{steps}}$ do
3:     Update time: $t^n = t^{n-1} + \Delta t$
4:     Update growth: $\mathbf{F}_g^n = \mathbf{F}_g(t^n)$
5:     Solve: $\delta\mathcal{P}_{\text{SLS,g}}(\mathbf{u}^n, \mathbf{C}_{m,v}^n) = 0$, with initial guess of $\{\mathbf{u}^n, \mathbf{C}_{m,v}^n\} = \{\mathbf{u}^{n-1}, \mathbf{C}_{m,v}^{n-1}\}$
6:     Store: $\{\mathbf{u}^n, \mathbf{C}_{m,v}^n\}$
7: end for
```

- Total time of the simulation: $t_{\text{sim}} = \Delta t \times N_{\text{steps}}$
- Sufficient resolution over visco-elastic transient response: $\Delta t/\tau_v \leq 10^{-1}$

- Smooth transitions of growth strains: $g_{\text{final}}/N_{\text{steps}} \ll 1$, in practice we set $g_{\text{final}}/N_{\text{steps}} \approx 10^{-2}$, where g_{final} is the growth strain corresponding to the value of the growth tensor in the final configuration $\mathbf{F}_g(t_{\text{sim}})$.

The following section demonstrates these principles through specific biological applications.

5.5 Simulations of Viscoelastic Morphogenesis

The theoretical framework developed previously provides a foundation for simulating morphogenetic processes in tissues with time-dependent mechanical properties. This section demonstrates the application of our viscoelastic growth model to epithelial morphogenesis, specifically examining apical constriction-driven tissue folding across two different developmental systems.

5.5.1 Apical Constriction in Epithelial Morphogenesis

Apical constriction represents a fundamental morphogenetic mechanism whereby epithelial cells reduce their apical surface area, generating tissue-scale deformations essential for development. This process drives critical developmental events including ventral furrow formation during *Drosophila* gastrulation[95] and neural tube closure during chordate neurulation [96]. Observations from experimental collaborators reveal a spatial pattern of mitochondrial enrichment at the apical domain of constricting cells Figures 5.6(B), 5.8(B). Immunofluorescence staining for the mitochondrial marker Tom20 demonstrates elevated apical mitochondrial density in regions destined to undergo constriction, preceding visible tissue deformation, suggesting a link between mitochondrial patterning and active tissue contractions.

5.5.2 Model Implementation

In our viscoelastic model, we implement apical constriction through spatially patterned growth tensors that contract the reference configuration of the mesh. We assume growth tensor is isotropic and takes the form:

$$\mathbf{F}_g = \frac{1}{1 - g(t)\beta(\mathbf{X})}\mathbf{I} \tag{5.62}$$

where $g(t)$ is active contractile strains representing the temporal evolution of constriction and $\beta(\mathbf{X}) \in [0, 1]$ defines the spatial distribution of contractions guided by experimentally observed mitochondrial patterns. The temporal evolution follows:

$$g(t) = g_{\text{fin}} \frac{t}{t_{\text{fin}}} \tag{5.63}$$

where g_{fin} denotes the maximum magnitude of contractile strains and t_{fin} the total simulation time.

For the viscoelastic response, we employ the Standard Linear Solid model with the energy rate functional from Equation 5.61:

$$\mathcal{P}_{\text{SLS,g}} = \int_\Omega \left[\dot{\Psi}_s(\mathbf{F}_g^{-T}\mathbf{C}\mathbf{F}_g^{-1}) + \dot{\Psi}_{m,e}(\mathbf{F}_g^{-T}\mathbf{C}\mathbf{F}_g^{-1}\mathbf{C}_{m,v}^{-1})\right] dV + \int_\Omega \phi(\mathbf{C}_{m,v}, \dot{\mathbf{C}}_{m,v})\, dV \tag{5.64}$$

where considering the form of the growth tensor from Equation 5.62, we can see that:

$$\mathbf{F}_g^{-T}\mathbf{C}\mathbf{F}_g^{-1} = \frac{1}{(1 - g(t)\beta(\boldsymbol{X}))^2}\mathbf{C} \tag{5.65}$$

and using this, the energy rate functional to be minimized at each given value of $g(t)$ is implemented as:

$$\mathcal{P}_{\text{SLS},g}(\boldsymbol{u}, \mathbf{C}_{m,v}) = \int_\Omega \left[\dot{\Psi}_s(\mathbf{C}, g, \beta) + \dot{\Psi}_{m,e}(\mathbf{C}\mathbf{C}_{m,v}^{-1}, g, \beta)\right] dV + \int_\Omega \phi(\mathbf{C}_{m,v}, \dot{\mathbf{C}}_{m,v})\, dV \tag{5.66}$$

5.5.3 Simulation Parameters and implementation details

The computational domain Ω is generated via *Gmsh*, as described in Section 2.1, and discretized using $\sim$20,000 elements first-order tetrahedral elements chosen by serial refinement until simulation results remained consistent. Geometrical parameters were chosen to match confocal imaging data. Simulation parameters and material constants for the SLS model are chosen based on experimental measurements of epithelial tissues and adjusted with time rescaled to observe the shape change, Young's modulus for both elastic components of SLS material: $E_m = E_s = E = 5$ kPa, Poisson's ratio: $\nu_m = \nu_s = \nu = 0.3$, viscosity: $\eta_m = 1000$ Pa·s, total simulation time: $t_{\text{fin}} = 1$ s. When simulation time is evolved, the minimization problem at each timestep is solved using the Newton's method with adaptive stepping, described in Section 2.4.2, with the initial timestep $\Delta t = 0.01 t_{\text{fin}}$. The simulation results are visualized using *ParaView* [49].

5.5.4 Simulation Results

Ventral Furrow Formation

The ventral furrow simulation begins with an ellipsoidal geometry representing *Drosophila* embryo. The contractility pattern $\beta(\mathbf{X})$ is prescribed based on Tom20 staining intensity, with maximum values along the ventral midline(Figure 5.6). Setting $g_{\text{fin}} = 0.5$ in the viscoelastic simulation produces invagination consistent with experimental observations (Figure 5.7).

Neural Tube Closure

Neural tube closure is modeled using a half-cylinder geometry representing the neural plate. The contractility pattern follows the bilateral distribution of mitochondrial enrichment observed in chicken embryos, with peak values along the neural folds (Figure 5.8). With $g_{\text{fin}} = 0.4$, the simulation reproduces the characteristic folding and fusion sequence (Figure 5.9). The viscoelastic framework captures the progressive nature of neural tube closure, where anterior regions close before posterior regions.

These simulations successfully captured morphogenetic outcomes by implementing time-dependent mechanics alongside growth, using the dynamic framework for viscoelastic

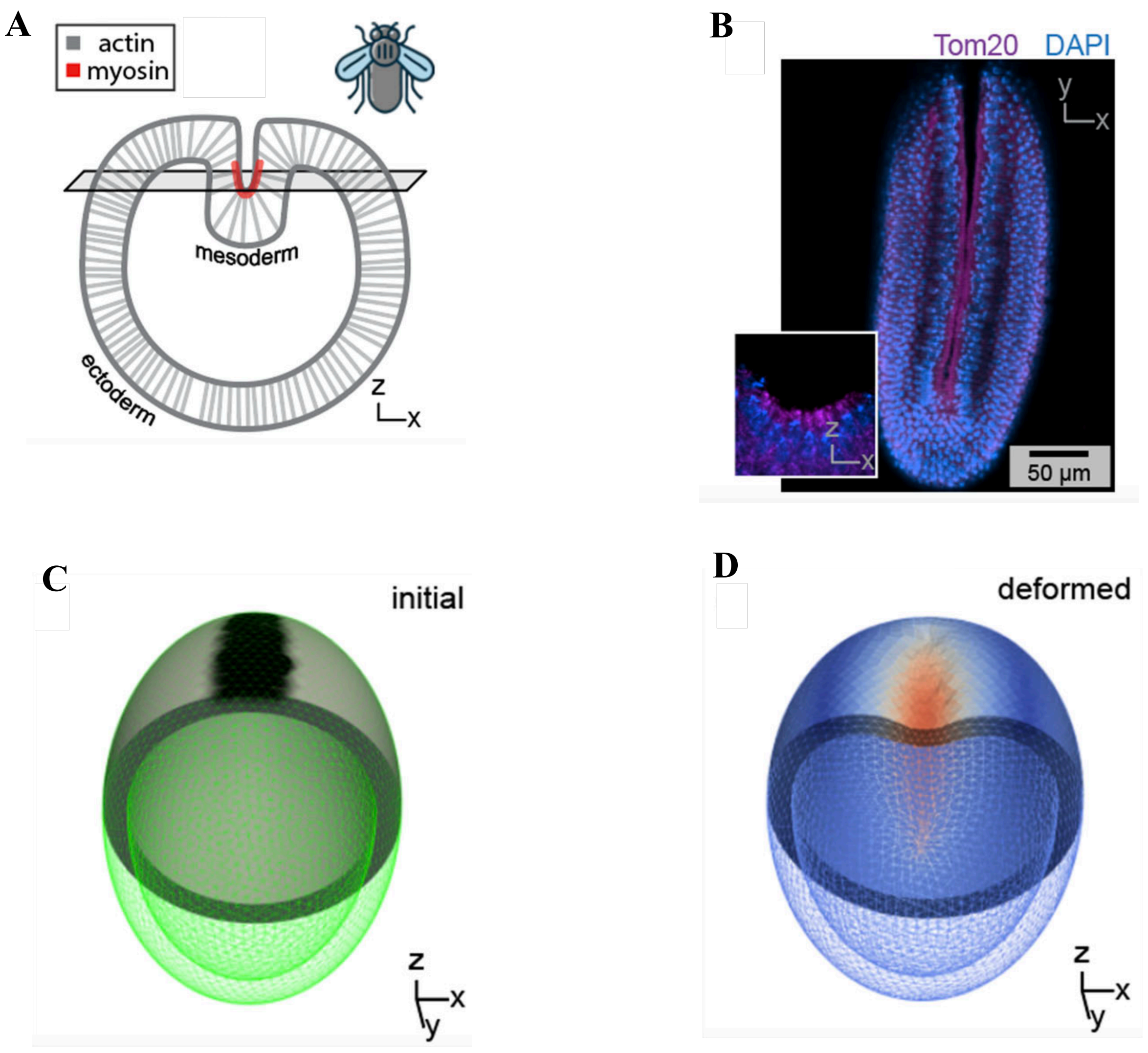


Figure 5.6: **(A)** Schematic of ventral furrow formation during gastrulation in *Drosophila*. **(B)** Confocal images showing staining for Tom20 (magenta) and nuclei (blue) during gastrulation. Inset shows z-x section. Scale bar 50 μm. **(C)** Cutaway view of initial geometry for simulation based on mitochondrial patterning of contractions during formation of the ventral furrow. Numerical mesh drawn in green, with applied active contraction field shown in grayscale. **(D)** Cutaway view of mesh deformed from (C) by active contractions, colored by the displacement field.

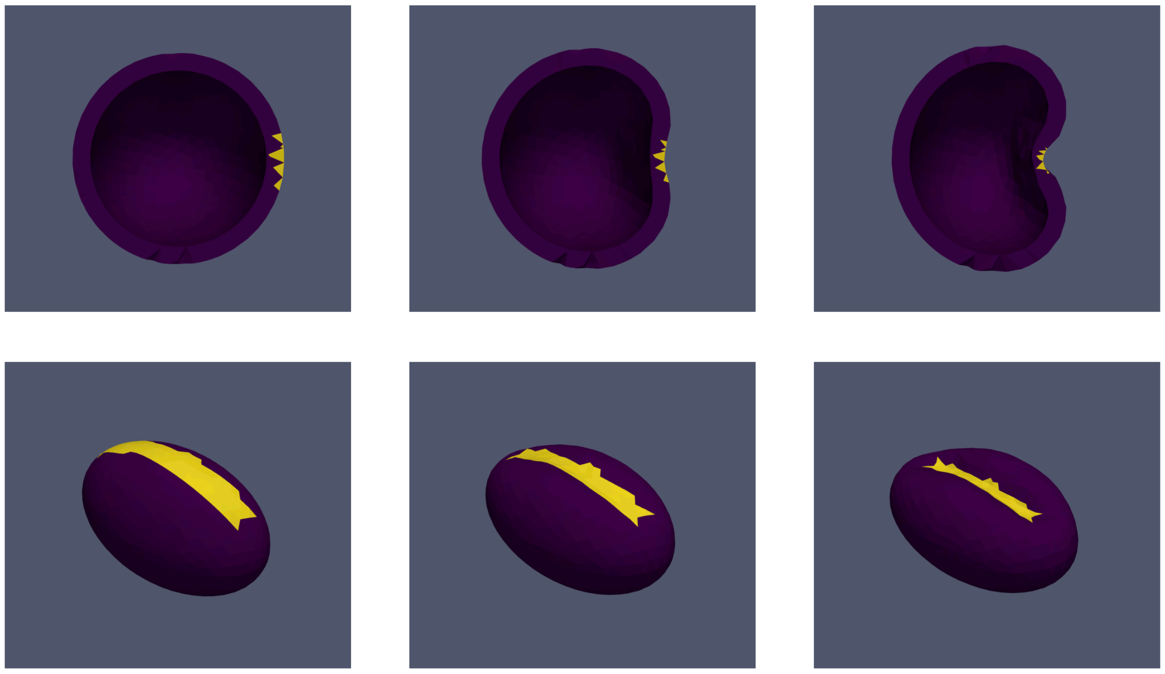


Figure 5.7: Frames of simulation showing deformation with increasing active contractions during ventral furrow formation. Regions where active contraction is concentrated is marked in yellow. The maximum value of active contraction strain $g(t)$ increases from left to right; starting from zero at the initial state(left), 25% in the middle, and 50% on the right. Top row: Slice view of the deforming mesh at the x-z midplane. Bottom row: Perspective view.

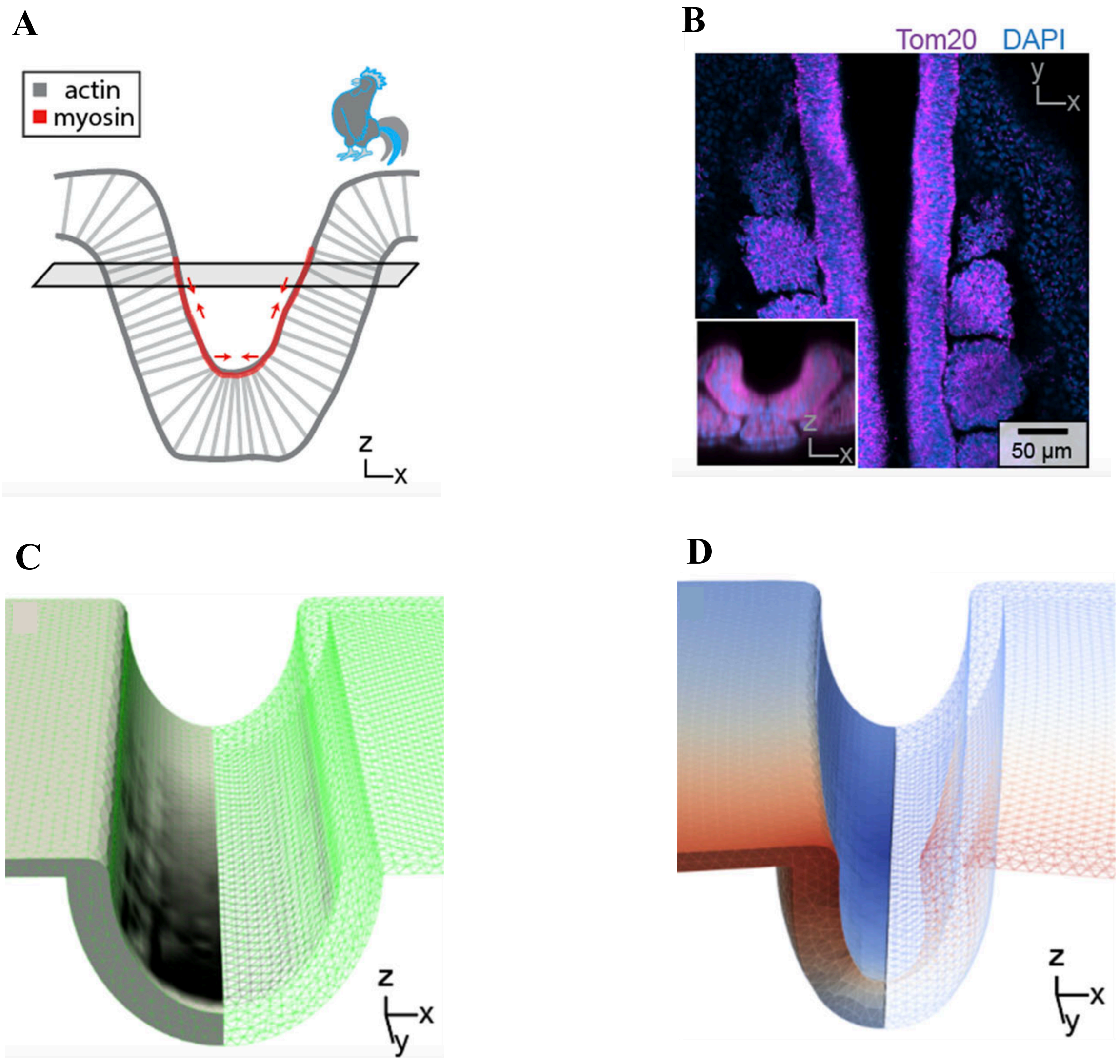


Figure 5.8: **(A)** Schematic of closure of the neural tube during neurulation in chicken. **(B)** Confocal images showing staining for Tom20 (magenta) and nuclei (blue) during neurulation. Inset shows z-x section. Scale bar 50 μm. **(C)** Cutaway view of initial geometry for simulation based on mitochondrial patterning of contractions during formation of the neural tube. Numerical mesh drawn in green, with applied active contraction field shown in grayscale. **(D)** Cutaway view of mesh deformed from (C) by active contractions, colored by the displacement field.

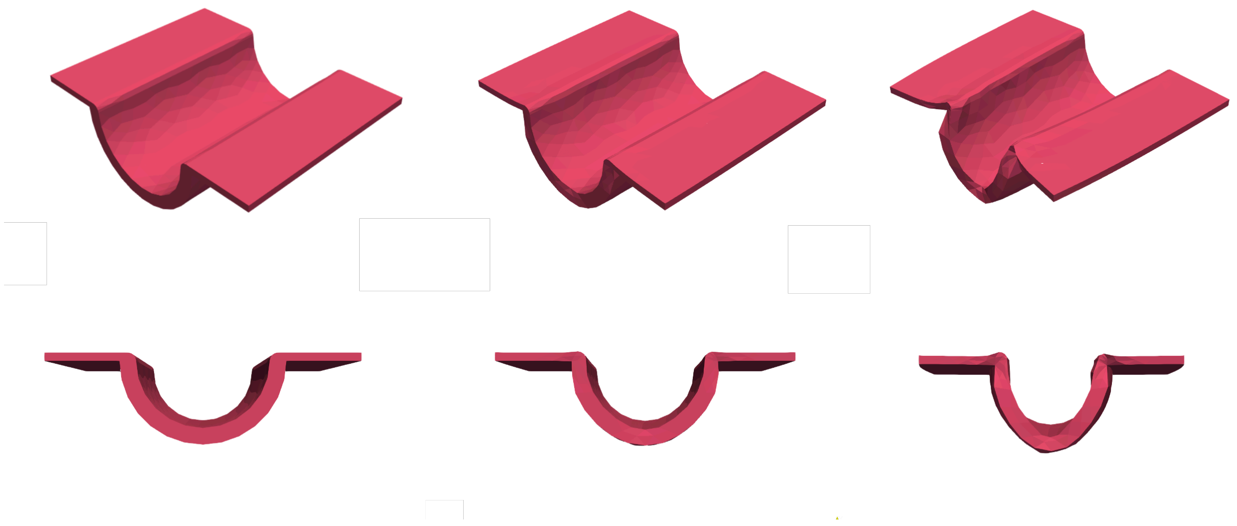


Figure 5.9: Frames of simulation showing deformation with increasing active contractions during closure of neural tube in chicken. Active contraction acts in the regions marked in Figure 5.8(C). The maximum value of active contraction strain $g(t)$ increases from left to right; starting from zero at the initial state(left), 20% in the middle, and 40% on the right. Top row: Perspective view. Bottom row: Top view from x-z plane.

morphing developed in this chapter.

5.6 Conclusion

This chapter extended the morphoelastic framework to incorporate viscoelastic material behavior, addressing a fundamental limitation when modeling biological tissues with time-dependent mechanical properties.

We developed an energy-rate minimization formulation that captures both elastic energy storage and viscous dissipation, enabling finite deformation analysis of viscoelastic materials. The modular rheological framework systematically constructs complex material behaviors from elementary spring and dashpot elements. This approach was validated through creep test simulations for Kelvin-Voigt, Maxwell, and Standard Linear Solid models, showing excellent agreement with analytical solutions.

Application to epithelial morphogenesis demonstrated the framework's biological relevance. Using experimentally observed mitochondrial patterns to guide spatial contractility, we successfully simulated ventral furrow formation in *Drosophila* and neural tube closure in chicken embryos. These simulations revealed how the interplay between growth rate and viscoelastic relaxation time determines final tissue morphology.

The viscoelastic framework developed here provides essential capabilities for modeling morphogenesis when growth and mechanical relaxation occur on comparable timescales, establishing a foundation for investigating more complex coupled phenomena in developmental biology.

Chapter 6

Conclusions and Future Directions

6.1 Summary of Contributions

This dissertation developed computational methods to predict and design morphing in biological and engineered systems. We addressed challenges in simulating growth-induced shape change, designing self-folding structures under gravity, and modeling viscoelastic tissue deformation. The key contributions are:

6.1.1 Computational Framework for Growth Mechanics

We established robust numerical methods for simulating large deformations driven by heterogeneous growth fields. The framework incorporates mesh generation for complex geometries, energy minimization formulations for finite deformation, and adaptive solution algorithms including arc-length methods for tracking instabilities. This computational infrastructure enabled all subsequent investigations and provides a foundation for future morphogenesis research.

6.1.2 Mechanistic Understanding of Biological Morphogenesis

Through application to lizard lung development, we revealed how mechanical principles govern epithelial corrugation formation. Our simulations demonstrated that luminal pressure drives initial protrusions through gaps in smooth muscle meshworks, with a critical stiffness ratio of $E_{sm}/E_{ep} \geq 25$ required for successful morphogenesis. These findings establish quantitative design principles that link tissue architecture to mechanical properties.

6.1.3 Predictive Design Tools for Self-Folding Systems

We developed integrated computational and machine learning approaches to overcome scaling limitations in self-folding structures. This created both forward prediction capabilities and pathways for inverse design for large scale self-folding. This framework enables systematic exploration of design spaces and identification of parameter regimes that achieve reliable folding under terrestrial gravity.

6.1.4 Viscoelastic Extensions for Biological Tissues

We extended morphoelastic theory to incorporate time-dependent material behavior, addressing a critical limitation in elastic modeling of biological systems. The formulated energy-rate minimization captures both reversible elastic effects dissipative viscous effects, with applications to epithelial morphogenesis demonstrating that growth and mechanical relaxation coupling determines final tissue configurations.

6.2 Insights

The following unifying insights for morphing matter emerge from our work:

Material heterogeneity enables complex morphologies. Both biological tissues and engineered systems exploit spatial variations in material properties to achieve targeted shape transformations.

The computational methods developed reveal how heterogeneous growth fields and patterned active materials can be designed to produce specific morphologies.

Time scales determines modeling choices The relationship between growth timescales and material relaxation influences final configurations. When these timescales are comparable, viscoelastic effects cannot be neglected since the system need not remain in a near-equilibrium state as growth evolves, necessitating the time-dependent formulations developed in Chapter 5. For growth occurring much slower than mechanical responses, the morphoelastic framework (Chapter 2) which deforms through successive growth states without time-dependent mechanics remains valid.

6.3 Limitations and Future Directions

While this dissertation advances computational morphogenesis, several limitations suggest directions for future research:

6.3.1 Mechanical feedback to growth

The current framework treats growth tensors as prescriptive and independent of ongoing mechanical deformation. However, experimental evidence indicates that growth rates respond to mechanical stress through mechanotransduction pathways [97]. Future work of morphing systems can model internal growth as determined by a prescribed baseline and feedback from mechanical stress.

6.3.2 Computational Efficiency

The high computational cost of full three-dimensional simulations limits design space exploration to achieve predictive design of self-folding units. Advances in GPU-accelerated finite element methods and reduced-order modeling could accelerate simulations of complex morphing systems.

6.3.3 Inverse Design Automation

The machine learning approaches developed for self-folding systems suggest broader possibilities for automated design. Extending these methods to growth-driven morphogenesis and multi-material systems could enable discovery of solutions to diverse shape-programming challenges.

6.4 Broader Impact

The computational frameworks developed in this dissertation can have potential impact beyond the specific systems studied:

Biomedical Engineering: Understanding tissue morphogenesis enables design of scaffolds with reagents to guide tissue engineering. The viscoelastic framework provides tools for predicting how engineered tissues will evolve under mechanical stimuli.

Soft Robotics: Self-folding principles can create deployable robotic systems with minimal actuation requirements. Moreover, our models predicting folding deformation can be extended to also predict stresses and strains during operation. Our forward and inverse design framework provides guidelines for designing autonomous morphing.

Sustainable Manufacturing: Shape-programmable materials reduce assembly complexity and enable flat-pack shipping of three-dimensional structures. The inverse design tools facilitate development of self-assembling products.

Fundamental Biology: Computational models reveal mechanical principles underlying developmental processes, complementing genetic and molecular approaches to understanding morphogenesis.

6.5 Concluding Remarks

This dissertation demonstrates that computational mechanics provides a powerful lens for understanding and designing shape change across biological and engineered systems. By developing unified physics-based models that can span from cellular contractility to architectural-scale folding, we establish quantitative connections between material properties, mechanical forces, and morphological outcomes.

The methods presented here offer both fundamental insights into biological morphogenesis and practical tools for engineering adaptive structures. As manufacturing capabilities advance and our understanding of biological development deepens, these computational approaches will play an increasingly impactful role in designing the shape-shifting systems of the future.

Bibliography

[1] Michael A. Palmer et al. "Stress ball morphogenesis: How the lizard builds its lung". In: *Science Advances* 7.52 (Dec. 22, 2021). Publisher: American Association for the Advancement of Science, eabk0161. doi: `10.1126/sciadv.abk0161`. url: `https://www.science.org/doi/10.1126/sciadv.abk0161` (visited on 03/20/2025).

[2] Raymond E. Goldstein and Alain Goriely. "Dynamic buckling of morphoelastic filaments". In: *Physical Review E* 74.1 (July 11, 2006). Publisher: American Physical Society, p. 010901. doi: `10.1103/PhysRevE.74.010901`. url: `https://link.aps.org/doi/10.1103/PhysRevE.74.010901` (visited on 05/14/2025).

[3] Alain Goriely. *The Mathematics and Mechanics of Biological Growth*. Vol. 45. Interdisciplinary Applied Mathematics. New York, NY: Springer, 2017. isbn: 978-0-387-87709-9 978-0-387-87710-5. doi: `10.1007/978-0-387-87710-5`. url: `http://link.springer.com/10.1007/978-0-387-87710-5` (visited on 05/14/2025).

[4] Roman Vetter et al. "Simulating Thin Sheets: Buckling, Wrinkling, Folding and Growth". In: *Journal of Physics: Conference Series* 487.1 (Mar. 2014), p. 012012. issn: 1742-6596. doi: `10.1088/1742-6596/487/1/012012`. url: `https://dx.doi.org/10.1088/1742-6596/487/1/012012` (visited on 05/14/2025).

[5] Katharine Goodwin et al. "Smooth muscle differentiation shapes domain branches during mouse lung development". In: *Development (Cambridge, England)* 146.22 (Nov. 25, 2019), dev181172. issn: 1477-9129. doi: `10.1242/dev.181172`.

[6] Pascal Jean Frey and Paul Louis George. *Mesh Generation: Application to Finite Elements*. Wiley, 2008. isbn: 978-1-84821-029-5.

[7] Joe F. Thompson, Bharat K. Soni, and Nigel P. Weatherill. *Handbook of Grid Generation*. CRC Press, 1999. isbn: 978-0-8493-2687-5.

[8] Christophe Geuzaine and Jean-François Remacle. "Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities". In: *International Journal for Numerical Methods in Engineering* 79.11 (2009). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.2579, pp. 1309–1331. issn: 1097-0207. doi: `10.1002/nme.2579`. url: `https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.2579` (visited on 03/20/2025).

[9] Jochen Heinz Müller, Erwin Stein, and Lars Rose. "Quality mesh generation based on optimal Delaunay triangulations". In: *Computer Methods in Applied Mechanics and Engineering* 191.45 (2002), pp. 5119–5139. doi: `10.1016/S0045-7825(02)00438-0`.

[10] Gerhard A. Holzapfel. *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*. Wiley, 2000. isbn: 978-0-471-82319-3.

[11] Edward K. Rodriguez, Anne Hoger, and Andrew D. McCulloch. "Stress-dependent finite growth in soft elastic tissues". In: *Journal of Biomechanics* 27.4 (Apr. 1, 1994), pp. 455–467. issn: 0021-9290. doi: `10.1016/0021-9290(94)90021-3`. url: `https://www.sciencedirect.com/science/article/pii/0021929094900213` (visited on 03/20/2025).

[12] Klaus-Jürgen Bathe. *Finite Element Procedures*. Google-Books-ID: rWvefGICfO8C. Klaus-Jurgen Bathe, 2006. 1056 pp. isbn: 978-0-9790049-0-2.

[13] Stephen P. Timoshenko and James Norman Goodier. *Theory of Elasticity*. 3rd ed. McGraw-Hill, 1970. isbn: 978-0-07-064720-6.

[14] Jay D. Humphrey. *Cardiovascular Solid Mechanics: Cells, Tissues, and Organs*. Springer, 2002. isbn: 978-0-387-95168-3. doi: `10.1007/978-0-387-21576-1`.

[15] Javier Bonet and Richard D. Wood. *Nonlinear Continuum Mechanics for Finite Element Analysis*. 2nd ed. Cambridge University Press, 2008. isbn: 978-0-521-83870-2.

[16] Davide Ambrosi and Federica Guana. “Stress-modulated growth”. In: *Mathematics and Mechanics of Solids* 12.3 (2007), pp. 319–342. doi: `10.1177/1081286505059739`.

[17] Ellen Kuhl. “Growing matter: A review of growth in living systems”. In: *Journal of the Mechanical Behavior of Biomedical Materials* 29 (2014), pp. 529–543. doi: `10.1016/j.jmbbm.2013.10.009`.

[18] Martine Ben Amar and Alain Goriely. “Growth and instability in elastic tissues”. In: *Journal of the Mechanics and Physics of Solids* 53.10 (2005), pp. 2284–2319. doi: `10.1016/j.jmps.2005.04.008`.

[19] Thomas J. R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Publications, 2012. isbn: 978-0-486-41181-1.

[20] Hans Petter Langtangen and Kent-Andre Mardal. *Introduction to Numerical Methods for Variational Problems*. Vol. 21. Texts in Computational Science and Engineering. Cham: Springer International Publishing, 2019. isbn: 978-3-030-23787-5 978-3-030-23788-2. doi: `10.1007/978-3-030-23788-2`. url: `http://link.springer.com/10.1007/978-3-030-23788-2` (visited on 05/16/2025).

[21] Mats G. Larson and Fredrik Bengzon. *The Finite Element Method: Theory, Implementation, and Applications*. Vol. 10. Texts in Computational Science and Engineering. Berlin, Heidelberg: Springer, 2013. isbn: 978-3-642-33286-9 978-3-642-33287-6. doi: `10.1007/978-3-642-33287-6`. url: `https://link.springer.com/10.1007/978-3-642-33287-6` (visited on 05/16/2025).

[22] Anders Logg, Kent-Andre Mardal, and Garth Wells, eds. *Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book*. Vol. 84. Lecture Notes in Computational Science and Engineering. Berlin, Heidelberg: Springer, 2012. isbn: 978-3-642-23098-1 978-3-642-23099-8. doi: 10.1007/978-3-642-23099-8. url: https://link.springer.com/10.1007/978-3-642-23099-8 (visited on 05/16/2025).

[23] Ted Belytschko et al. *Nonlinear Finite Elements for Continua and Structures*. 2nd ed. Wiley, 2013. isbn: 978-1-118-63270-3.

[24] Eduard Riks. “An incremental approach to the solution of snapping and buckling problems”. In: *International Journal of Solids and Structures* 15.7 (1979), pp. 529–551. doi: 10.1016/0020-7683(79)90081-7.

[25] Michael A. Crisfield. *Non-linear Finite Element Analysis of Solids and Structures*. Vol. 1. Wiley, 1991. isbn: 978-0-471-92956-7.

[26] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, 2003. isbn: 978-0-89871-534-7. doi: 10.1137/1.9780898718003.

[27] Ewald R. Weibel. *Symmorphosis: On Form and Function in Shaping Life*. Google-Books-ID: c4Wf0qx9hSYC. Harvard University Press, 2000. 288 pp. isbn: 978-0-674-00068-1.

[28] S. F. Perry and H. -R. Duncker. “Lung architecture, volume and static mechanics in five species of lizards”. In: *Respiration Physiology* 34.1 (July 1, 1978), pp. 61–81. issn: 0034-5687. doi: 10.1016/0034-5687(78)90049-X. url: https://www.sciencedirect.com/science/article/pii/003456877890049X (visited on 03/24/2025).

[29] “Development”. In: *The Lung-Air Sac System of Birds: Development, Structure, and Function*. Ed. by John N. Maina. Berlin, Heidelberg: Springer, 2005, pp. 13–63. isbn: 978-3-540-29727-7. doi: 10.1007/3-540-29727-8_2. url: https://doi.org/10.1007/3-540-29727-8_2 (visited on 03/24/2025).

[30] John Burnard West. *Respiratory Physiology: The Essentials*. Google-Books-ID: rHaZVmFZRVcC. Lippincott Williams & Wilkins, 2008. 206 pp. isbn: 978-0-7817-7206-8.

[31] C. G. Farmer. “The Evolution of Unidirectional Pulmonary Airflow”. In: *Physiology* 30.4 (July 2015). Publisher: American Physiological Society, pp. 260–272. issn: 1548-9213. doi: `10.1152/physiol.00056.2014`. url: `https://journals.physiology.org/doi/full/10.1152/physiol.00056.2014` (visited on 03/24/2025).

[32] Robert L. Cieri et al. “New insight into the evolution of the vertebrate respiratory system and the discovery of unidirectional airflow in iguana lungs”. In: *Proceedings of the National Academy of Sciences of the United States of America* 111.48 (Dec. 2, 2014), pp. 17218–17223. issn: 1091-6490. doi: `10.1073/pnas.1405088111`.

[33] Robert L. Cieri and C. G. Farmer. “Unidirectional pulmonary airflow in vertebrates: a review of structure, function, and evolution”. In: *Journal of Comparative Physiology. B, Biochemical, Systemic, and Environmental Physiology* 186.5 (July 2016), pp. 541–552. issn: 1432-136X. doi: `10.1007/s00360-016-0983-3`.

[34] Tamara L Fetters and Joel W McGlothlin. “Life histories and invasions: accelerated laying rate and incubation time in an invasive lizard, Anolis sagrei”. In: *Biological Journal of the Linnean Society* 122.3 (Oct. 25, 2017), pp. 635–642. issn: 0024-4066. doi: `10.1093/biolinnean/blx102`. url: `https://doi.org/10.1093/biolinnean/blx102` (visited on 03/20/2025).

[35] Jessica Alföldi et al. “The genome of the green anole lizard and a comparative analysis with birds and mammals”. In: *Nature* 477.7366 (Sept. 2011). Publisher: Nature Publishing Group, pp. 587–591. issn: 1476-4687. doi: `10.1038/nature10390`. url: `https://www.nature.com/articles/nature10390` (visited on 03/20/2025).

[36] Steven F. Perry. “Morphometry of Reptilian Lungs”. In: *Reptilian Lungs: Functional Anatomy and Evolution*. Ed. by Steven F. Perry. Berlin, Heidelberg: Springer, 1983,

pp. 8–48. isbn: 978-3-642-68964-2. doi: `10.1007/978-3-642-68964-2_2`. url: `https://doi.org/10.1007/978-3-642-68964-2_2` (visited on 03/20/2025).

[37] Hye Young Kim et al. “Localized Smooth Muscle Differentiation Is Essential for Epithelial Bifurcation during Branching Morphogenesis of the Mammalian Lung”. In: *Developmental Cell* 34.6 (Sept. 28, 2015), pp. 719–726. issn: 1878-1551. doi: `10.1016/j.devcel.2015.08.012`.

[38] Michael A. Palmer and Celeste M. Nelson. “Fusion of airways during avian lung development constitutes a novel mechanism for the formation of continuous lumena in multicellular epithelia”. In: *Developmental Dynamics: An Official Publication of the American Association of Anatomists* 249.11 (Nov. 2020), pp. 1318–1333. issn: 1097-0177. doi: `10.1002/dvdy.215`.

[39] Tyler R. Huycke et al. “Genetic and Mechanical Regulation of Intestinal Smooth Muscle Development”. In: *Cell* 179.1 (Sept. 19, 2019), 90–105.e21. issn: 1097-4172. doi: `10.1016/j.cell.2019.08.041`.

[40] Amy E. Shyer et al. “Villification: How the Gut Gets Its Villi”. In: *Science* 342.6155 (Oct. 11, 2013). Publisher: American Association for the Advancement of Science, pp. 212–218. doi: `10.1126/science.1238842`. url: `https://www.science.org/doi/10.1126/science.1238842` (visited on 03/20/2025).

[41] Axel A. Thomson et al. “The role of smooth muscle in regulating prostatic induction”. In: *Development (Cambridge, England)* 129.8 (Apr. 2002), pp. 1905–1912. issn: 0950-1991. doi: `10.1242/dev.129.8.1905`.

[42] Robert L Cieri. “Pulmonary Smooth Muscle in Vertebrates: A Comparative Review of Structure and Function”. In: *Integrative and Comparative Biology* 59.1 (July 1, 2019), pp. 10–28. issn: 1540-7063. doi: `10.1093/icb/icz002`. url: `https://doi.org/10.1093/icb/icz002` (visited on 03/20/2025).

[43] Leonhard Euler. “Elementa doctrinae solidorum”. In: *Novi Commentarii academiae scientiarum Petropolitanae* (Jan. 1, 1758), pp. 109–140. url: `https://scholarlycommons.pacific.edu/euler-works/230`.

[44] Abdul Raqeeb et al. “Regulatable stiffness in relaxed airway smooth muscle: a target for asthma treatment?” In: *Journal of Applied Physiology (Bethesda, Md.: 1985)* 112.3 (Feb. 2012), pp. 337–346. issn: 1522-1601. doi: `10.1152/japplphysiol.01036.2011`.

[45] V. A. Lubarda and A. Hoger. “On the mechanics of solids with a growing mass”. In: *International Journal of Solids and Structures* 39.18 (Sept. 1, 2002), pp. 4627–4664. issn: 0020-7683. doi: `10.1016/S0020-7683(02)00352-9`. url: `https://www.sciencedirect.com/science/article/pii/S0020768302003529` (visited on 03/20/2025).

[46] R. W. Ogden. *Non-linear Elastic Deformations*. Google-Books-ID: 2u7wCaojfbEC. Courier Corporation, Jan. 1, 1997. 562 pp. isbn: 978-0-486-69648-5.

[47] Martin Alnæs et al. “The FEniCS Project Version 1.5”. In: *Archive of Numerical Software* 3.100 (Dec. 7, 2015). Number: 100. issn: 2197-8263. doi: `10.11588/ans.2015.100.20553`. url: `https://journals.ub.uni-heidelberg.de/index.php/ans/article/view/20553` (visited on 03/20/2025).

[48] Kendall E. Atkinson. *An Introduction to Numerical Analysis, 2nd Ed*. Google-Books-ID: lPV8Fv2XEosC. Wiley India Pvt. Limited, Sept. 2008. 716 pp. isbn: 978-81-265-1850-0.

[49] JAMES Ahrens, BERK Geveci, and CHARLES Law. “36 - ParaView: An End-User Tool for Large-Data Visualization”. In: *Visualization Handbook*. Ed. by Charles D. Hansen and Chris R. Johnson. Burlington: Butterworth-Heinemann, Jan. 1, 2005, pp. 717–731. isbn: 978-0-12-387582-2. doi: `10.1016/B978-012387582-2/50038-1`. url: `https://www.sciencedirect.com/science/article/pii/B9780123875822500381` (visited on 03/20/2025).

[50] Kevin S. Kolahi et al. “Effect of substrate stiffness on early mouse embryo development”. In: *PloS One* 7.7 (2012), e41717. issn: 1932-6203. doi: 10.1371/journal.pone.0041717.

[51] Victor D. Varner et al. “Mechanically patterning the embryonic airway epithelium”. In: *Proceedings of the National Academy of Sciences* 112.30 (July 28, 2015). Publisher: Proceedings of the National Academy of Sciences, pp. 9230–9235. doi: 10.1073/pnas.1504102112. url: https://www.pnas.org/doi/10.1073/pnas.1504102112 (visited on 03/20/2025).

[52] Daniela Rus and Michael T Tolley. “Design, fabrication and control of soft robots”. In: *Nature* 521.7553 (2015). Publisher: Nature Publishing Group, pp. 467–475.

[53] Elliot Hawkes et al. “Programmable matter by folding”. In: *Proceedings of the National Academy of Sciences* 107.28 (2010). Publisher: National Acad Sciences, pp. 12441–12445.

[54] Shannon A Zirbel et al. “Accommodating thickness in origami-based deployable arrays”. In: *Journal of Mechanical Design* 135.11 (2013). Publisher: American Society of Mechanical Engineers, p. 111005.

[55] Jesse L Silverberg et al. “Using origami design principles to fold reprogrammable mechanical metamaterials”. In: *Science* 345.6197 (2014). Publisher: American Association for the Advancement of Science, pp. 647–650.

[56] Samuel Felton et al. “Applied origami. A method for building self-folding machines”. In: *Science* 345.6197 (2014). Publisher: American Association for the Advancement of Science, pp. 644–646.

[57] Leonid Ionov. “Hydrogel-based actuators: possibilities and limitations”. In: *Materials Today* 17.10 (2014). Publisher: Elsevier, pp. 494–503.

[58] Jinlian Hu et al. “Recent advances in shape–memory polymers: structure, mechanism, functionality, modeling and applications”. In: *Progress in Polymer Science* 37.12 (2012). Publisher: Elsevier, pp. 1720–1763.

[59] Jessica Morgan, Spencer P Magleby, and Larry L Howell. “An approach to designing origami-adapted aerospace mechanisms”. In: *Journal of Mechanical Design* 138.5 (2016). Publisher: American Society of Mechanical Engineers, p. 052301.

[60] Collin L Randall, Evin Gultepe, and David H Gracias. “Self-folding devices and materials for biomedical applications”. In: *Trends in Biotechnology* 29.12 (2011). Publisher: Elsevier, pp. 597–606.

[61] Noy Bassik et al. “Enzymatically triggered actuation of miniaturized tools”. In: *Journal of the American Chemical Society* 132.46 (2010). Publisher: ACS Publications, pp. 16314–16317.

[62] Stephen P Timoshenko and James M Gere. *Theory of Elastic Stability*. 2nd. New York: McGraw-Hill, 1961.

[63] Jun-Hee Na et al. “Programming reversibly self-folding origami with micropatterned photo-crosslinkable polymer trilayers”. In: *Advanced Materials* 27.1 (2015). Publisher: Wiley Online Library, pp. 79–85.

[64] Michael T Tolley et al. “Self-folding origami: shape memory composites activated by uniform heating”. In: *Smart Materials and Structures* 23.9 (2014). Publisher: IOP Publishing, p. 094006.

[65] Andreas Lendlein and Robert Langer. “Biodegradable, elastic shape-memory polymers for potential biomedical applications”. In: *Science* 296.5573 (2002). Publisher: American Association for the Advancement of Science, pp. 1673–1676.

[66] H Jerry Qi et al. “Finite deformation thermo-mechanical behavior of thermally induced shape memory polymers”. In: *Journal of the Mechanics and Physics of Solids* 56.5 (2008). Publisher: Elsevier, pp. 1730–1751.

[67] Tomohiro Tachi. “Geometric considerations for the design of rigid origami structures”. In: *Proceedings of the International Association for Shell and Spatial Structures (IASS) Symposium*. Vol. 12. 2010, pp. 458–460.

[68] Koryo Miura. *Method of packaging and deployment of large membranes in space*. 618. Institute of Space and Astronautical Science, 1985, pp. 1–9.

[69] S. Timoshenko. “Analysis of Bi-Metal Thermostats”. In: *JOSA, Vol. 11, Issue 3, pp. 233-255* (Sept. 1, 1925). Publisher: Optica Publishing Group. doi: `10.1364/JOSA.11.000233`. url: `https://opg.optica.org/josa/abstract.cfm?uri=josa-11-3-233` (visited on 07/06/2025).

[70] Miles Cranmer. “Interpretable machine learning for science with PySR and SymbolicRegression.jl”. In: *arXiv preprint arXiv:2305.01582* (2023).

[71] Leo Breiman. “Random forests”. In: *Machine Learning* 45.1 (2001). Publisher: Springer, pp. 5–32.

[72] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *Nature* 521.7553 (2015). Publisher: Nature Publishing Group, pp. 436–444.

[73] Silviu-Marian Udrescu and Max Tegmark. “AI Feynman: A physics-inspired method for symbolic regression”. In: *Science Advances* 6.16 (2020). Publisher: American Association for the Advancement of Science, eaay2631.

[74] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd. New York: Springer, 2009.

[75] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. Cambridge, MA: MIT Press, 2016.

[76] J. Ross Quinlan. “Induction of decision trees”. In: *Machine Learning* 1.1 (1986). Publisher: Springer, pp. 81–106.

[77] *DecisionTreeClassifier*. scikit-learn. url: `https://scikit-learn/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html`.

[78] *Neural Networks — PyTorch Tutorials 2.7.0+cu126 documentation*. url: `https://pytorch.org/tutorials/beginner/blitz/neural_networks_tutorial.html,%20/beginner/blitz/neural_networks_tutorial.html` (visited on 07/28/2025).

[79] *Adam — PyTorch 2.7 documentation*. url: `https://docs.pytorch.org/docs/stable/generated/torch.optim.Adam.html` (visited on 07/28/2025).

[80] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. Jan. 30, 2017. doi: `10.48550/arXiv.1412.6980`. arXiv: `1412.6980[cs]`. url: `http://arxiv.org/abs/1412.6980` (visited on 07/28/2025).

[81] Yiqi Mao et al. “3D Printed Reversible Shape Changing Components with Stimuli Responsive Materials”. In: *Scientific Reports* 6.1 (Apr. 25, 2016). Publisher: Nature Publishing Group, p. 24761. issn: 2045-2322. doi: `10.1038/srep24761`. url: `https://www.nature.com/articles/srep24761` (visited on 07/19/2025).

[82] J.D. Humphrey. “Continuum biomechanics of soft biological tissues”. In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 459.2029 (2002), pp. 3–46.

[83] Y.C. Fung. *Biomechanics: Mechanical Properties of Living Tissues*. 2nd. New York: Springer Science & Business Media, 2013.

[84] T. Lecuit and P.F. Lenne. “Cell surface mechanics and the control of cell shape, tissue patterns and morphogenesis”. In: *Nature Reviews Molecular Cell Biology* 8.8 (2007), pp. 633–644.

[85] A.R. Harris et al. “Characterizing the mechanics of cultured cell monolayers”. In: *Proceedings of the National Academy of Sciences* 109.41 (2012), pp. 16449–16454.

[86] C. Guillot and T. Lecuit. “Mechanics of epithelial tissue homeostasis and morphogenesis”. In: *Science* 340.6137 (2013), pp. 1185–1189.

[87] L.A. Davidson. “Measuring mechanical properties of embryos and embryonic tissues”. In: *Methods in Cell Biology* 105 (2011), pp. 425–437.

[88] D. Kierzkowski et al. “Elastic domains regulate growth and organogenesis in the plant shoot apical meristem”. In: *Science* 335.6072 (2012), pp. 1096–1099.

[89] K. Doubrovinski et al. “Measurement of cortical elasticity in Drosophila melanogaster embryos using ferrofluids”. In: *Proceedings of the National Academy of Sciences* 114.5 (2017), pp. 1051–1056.

[90] N.I. Petridou et al. “Fluidization-mediated tissue spreading by mitotic cell rounding and non-canonical Wnt signalling”. In: *Nature Cell Biology* 21.2 (2019), pp. 169–178.

[91] P. Germain. “The Method of Virtual Power in Continuum Mechanics. Part 2: Microstructure”. In: *SIAM Journal on Applied Mathematics* 25.3 (1973). Publisher: Society for Industrial and Applied Mathematics, pp. 556–575. issn: 0036-1399. url: `https://www.jstor.org/stable/2100123` (visited on 07/28/2025).

[92] Eduardo Fancello, Jean-Philippe Ponthot, and Laurent Stainier. “A variational formulation of constitutive models and updates in non-linear finite viscoelasticity”. In: *International Journal for Numerical Methods in Engineering* 65.11 (2006). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.1525, pp. 1831–1864. issn: 1097-0207. doi: `10.1002/nme.1525`. url: `https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.1525` (visited on 07/28/2025).

[93] M. Ortiz and L. Stainier. “The variational formulation of viscoplastic constitutive updates”. In: *Computer Methods in Applied Mechanics and Engineering* 171.3 (Apr. 9, 1999), pp. 419–444. issn: 0045-7825. doi: `10.1016/S0045-7825(98)00219-9`. url: `https://www.sciencedirect.com/science/article/pii/S0045782598002199` (visited on 07/28/2025).

[94] R. Radovitzky and M. Ortiz. "Error estimation and adaptive meshing in strongly nonlinear dynamic problems". In: *Computer Methods in Applied Mechanics and Engineering* 172.1 (Apr. 16, 1999), pp. 203–240. issn: 0045-7825. doi: `10.1016/S0045-7825(98)00230-8`. url: `https://www.sciencedirect.com/science/article/pii/S0045782598002308` (visited on 07/28/2025).

[95] Adam C. Martin, Matthias Kaschube, and Eric F. Wieschaus. "Pulsed contractions of an actin-myosin network drive apical constriction". In: *Nature* 457.7228 (Jan. 22, 2009), pp. 495–499. issn: 1476-4687. doi: `10.1038/nature07522`.

[96] Jean-François Colas and Gary C. Schoenwolf. "Towards a cellular and molecular understanding of neurulation". In: *Developmental Dynamics* 221.2 (2001). _eprint: https://anatomypubs.onlinelibrary.wiley.com/doi/pdf/10.1002/dvdy.1144, pp. 117–145. issn: 1097-0177. doi: `10.1002/dvdy.1144`. url: `https://onlinelibrary.wiley.com/doi/abs/10.1002/dvdy.1144` (visited on 07/28/2025).

[97] Akiko Mammoto, Tadanori Mammoto, and Donald E. Ingber. "Mechanosensitive mechanisms in transcriptional regulation". In: *Journal of Cell Science* 125 (Pt 13 July 1, 2012), pp. 3061–3073. issn: 1477-9137. doi: `10.1242/jcs.093005`.