Mem3DG: Modeling Membrane Mechanochemical Dynamics in 3D using Discrete Differential Geometry

C. Zhu[1], C. T. Lee[1*], and, P. Rangamani[1*]

[1]Department of Mechanical and Aerospace Engineering, University of California San Diego, La Jolla CA 92093;

[*]To whom correspondence must be addressed: ctlee@ucsd.edu, prangamani@ucsd.edu

Abstract

Biomembranes adopt varying morphologies that are vital to cellular functions. Many studies use computational modeling to understand how various mechanochemical factors contribute to membrane shape transformations. Compared to approximation-based methods (e.g., finite element method), the class of discrete mesh models offers greater flexibility to simulate complex physics and shapes in three dimensions; its formulation produces an efficient algorithm while maintaining coordinate-free geometric descriptions. However, ambiguities in geometric definitions in the discrete context have led to a lack of consensus on which discrete mesh model is theoretically and numerically optimal; a bijective relationship between the terms contributing to both the energy and forces from the discrete and smooth geometric theories remains to be established. We address this and present an extensible framework, `Mem3DG`, for modeling 3D mechanochemical dynamics of membranes based on Discrete Differential Geometry (DDG) on triangulated meshes. The formalism of DDG resolves the inconsistency and provides a unifying perspective on how to relate the smooth and discrete energy and forces. To demonstrate, `Mem3DG` is used to model a sequence of examples with increasing mechanochemical complexity: recovering classical shape transformations such as 1) biconcave disk, dumbbell, and unduloid and 2) spherical bud on spherical, flat-patch membrane; investigating how the coupling of membrane mechanics with protein mobility jointly affects phase and shape transformation. As high-resolution 3D imaging of membrane ultrastructure becomes more readily available, we envision `Mem3DG` to be applied as an end-to-end tool to simulate realistic cell geometry under user-specified mechanochemical conditions.

Keywords

Discrete differential geometry; Membrane mechanics; Triangulated mesh; Cell shape; Membrane trafficking; Helfrich Hamiltonian

Why it matters

Cellular membranes have shapes and shape changes which characterize cells/organelles, and support nutrient trafficking among other critical processes. Modeling membrane shape changes using mechanical principles can provide insight into how cells robustly bend membranes to support life. Mathematical and computational strategies to solve the equations describing membrane shape evolution can be complex and challenging without simplifying assumptions. Here, we present a new, general, numerical approach to model arbitrary 3D membrane shapes in response to interaction with curvature sensing and generating membrane proteins. The accompanying implementation, `Mem3DG`, is a software tool to make computational membrane mechanics accessible to the general researcher.

Highlights

- Introduces a discrete theory for membrane mechanics which connects with smooth theory
- Discrete energy/force are functions of basic geometric quantities
- Recipes for extending the discrete framework with additional physics are provided

- Describes a user-friendly software implementation called `Mem3DG`
- `Mem3DG` is designed to facilitate modeling in tandem with experimental studies

Significance and novelty

Helfrich Hamiltonian is widely used to model biomembranes. Many numerical methods have been developed to solve the geometric PDE. Compared to approximation-based methods, discrete-mesh-based models have many advantages when incorporating multiphysics in 3D due to their flexibility, efficiency, and straightforward implementation. However, there is no consensus on which mesh-based model is optimal, and a connection between the discrete and smooth geometric theory remains obscure. In this work, we provide a unifying perspective by identifying fundamental geometric invariants of the discrete force derived from a discrete energy. As a demonstration of generality of the framework to various physics, we follow a formulaic procedure to derive additional physics such as interfacial line tension, surface-bulk adsorption, protein lateral diffusion, and curvature-dependent protein aggregation.

1 Introduction

Computational modeling of lipid bilayer mechanics has long been accepted as a way to probe the biophysical aspects of membrane curvature generation. The ability of lipid bilayers and cellular membranes to bend in response to various applied forces has been studied extensively from the mathematical modeling perspective. However, the nonlinear system of equations that result from such modeling often leads to a computational bottleneck to generate predictions from simulations that can be tested against experimentally observed shapes. In this study, we develop a mesh-based model using discrete differential geometry to reduce this bottleneck. To justify why our method is necessary and is a computational advance, we first describe the importance of membrane curvature generation in biology, the current state-of-the-art in membrane mechanics modeling, and finally explicitly state the goals of our approach.

1.1 Membrane curvature generation in biology

As one of the most essential and conserved structures of cells, cellular membranes perform many functions. First, they form compartments to separate chemical environments. Beyond the passive role of partitioning space, lipids in the membranes interact with proteins and other cellular components influencing cell signaling (e.g., by localizing molecules and acting as an entropic barrier) (*1*, *2*). Membrane morphology and topology changes are critical for trafficking cargo in and out of cells and are very carefully regulated (*3*–*8*). Central to these roles is the ability of the membrane to curve and adopt varying morphological configurations from spheres to highly-curved and variegated structures.

Advances in experimental studies of membrane–protein interactions (*9*–*20*), ultrastructural imaging (*21*–*30*), and image analysis (*9*–*11*, *31*–*37*) have revealed much about the molecular interactions that regulate membrane curvature. To investigate the mechanics behind these interactions, many theoretical and computational models in terms of membrane energetics and thermodynamics have been developed (*7*, *38*–*52*). These models, owing to the ease of *in silico* experimentation, have become an important tool for generating and testing hypotheses (*53*, *54*). These mechanics models and associated simulations have been used to provide intuition on the mechanical requirements for forming and maintaining complex cellular membrane shapes (*55*–*63*).

While the utility of this approach has been established and many models have been developed (*38*), many models are limited by critical assumptions or other technical challenges. For example, the ability to use geometries from membrane ultrastructural imaging experiments as a starting condition would improve model realism (*64*). With respect to computational complexity, the solver should be able to model deformations and topological changes in three dimensions and be compatible with both energy minimization and time-integration for comparing with static and time-series experiments respectively. This is in contrast to the current

assumptions of the existence of an axis of symmetry that is quite commonly made for purposes of ease of simulation (*65*). An additional feature for these solvers should be that their implementation is modular such that the addition of new physics or increasing model complexity should be straightforward. This includes the potential for coupling the membrane model with agent-based and other simulations to propagate other cellular components such as the cytoskeleton (*66*). Thus, new computational tools which are general, easy to use, and without restrictive assumptions are needed to bring modeling closer to experimental observations of membrane shapes in cells.

1.2 State-of-the-art membrane modeling

To emphasize the motivations behind our choice of extending and developing a new mesh-based membrane model, we provide a summary of the legacy literature in modeling membrane mechanics. The most common theoretical model of membrane bending is the Helfrich-Canham-Evans Hamiltonian,[1] which describes the lipid bilayer as a two-dimensional fluid-like structure that exhibits solid-like elasticity in the out-of-plane direction (*39*, *40*, *68*–*70*). It is a continuum model which describes the bending energy of the membrane as a function of its mean and Gaussian curvatures. The assumptions for the continuum are satisfied as long as the deformations are much larger in length scale compared to the individual lipid components.

Given the necessary material properties and boundary conditions, by minimizing the Helfrich energy, we can obtain the equilibrium shape of the membrane (*39*, *70*–*72*). While straightforward in concept, energy minimization requires the determination of the forces on the membrane which is a challenging task (*65*). The forces on the membrane are given by the variation of the energy with respect to the embedded coordinate (i.e., shape) of the membrane[2]. Taking the shape derivatives of the Helfrich energy produces the "shape equation", so termed because solutions of this partial differential equation, with the prescribed boundary conditions, produce configurations at equilibrium (i.e., force-balance).

Solving the shape equation is non-trivial since it is a partial differential equation with fourth-order nonlinear terms. As a result, analytical solutions of the shape equation are known only for a few cases constrained to specific geometries and boundary conditions (*42*). For most systems, we must resort to the use of numerical methods. The simplest numerical schemes can be formulated by making restrictive assumptions such as considering only small deformations from a plane (e.g., Monge parametrization) or assuming that there exists an axis of symmetry such that the resulting boundary value system can be integrated (*38*). While these methods are suitable for idealized shapes, these assumptions are not consistent with the membrane shapes found in biology are and thus not general enough for advancing the field.

Solvers of membrane shape in 3D have also been developed and can be categorized into three groups: 1) phase field or level set methods (*73*–*75*), 2) Finite Element Method (FEM) (*76*–*83*), and 3) discrete surface mesh models (*60*, *84*–*94*). These methods and others, reviewed in detail by Guckenberger *et al.* (*95*), differ in the strategy used to discretize the membrane domain and compute the relevant derivatives. We compare the aforementioned general, 3D models to our established model criteria in Table 1 and elaborate below.

Phase-field and level set methods solve the shape equation by propagating a density field on an ambient volumetric mesh. The membrane shape is implicit in these models and can be found by drawing an isosurface or level set of the model. While this is ideal for modeling membrane topological changes, the implicit representation of the membrane adds complexity for interfacing with data generated using modern methods of visualizing membrane ultrastructure. The meshes output from ultrastructural studies must be converted into a density or phase field prior to input to the model. While this conversion is possible, representing the dynamic and variegated shapes of cellular membranes would require a dense volume mesh, which reduces computational tractability. The implicit surface representation also complicates the addition of new in-plane physics for end-users.

FEM and discrete mesh models use an explicit surface parametrization (i.e., a mesh). Thus the meshes output from ultrastructural imaging datasets can be used in these frameworks with minor modifications (*32*,

[1] The Helfrich energy is related to the Willmore energy in the mathematics literature (*67*)

[2] We call this variation the shape derivative which is distinct from the chemical derivative that will be introduced later in the context of mechanochemical coupling

Table 1: Comparison of common mathematical frameworks for modeling membrane mechanics with specifications to advance the mission of computational membrane mechanobiology. A general framework will permit the easy transfer of inputs and results between model and experiments. Models which can be coupled with other modeling schemes representing other cellular components can help address the complexity of cell biology. Discrete mesh models have many desirable traits, with respect to these specifications, at the cost of forgoing rigorous error analysis.

	Phase field/level set	**FEM**	**Discrete Mesh/`Mem3DG`**
General 3D	✓	✓	✓
Statics + dynamics	✓	✓	✓
Membrane heterogeneity	✓	✓	✓
Incorporation of agents/particles			✓
Incorporation of stochastic dynamics (e.g., DPD or MC)			✓
Explicit surface parametrization		✓	✓
Coordinate-free evaluation			✓
Ability to support topological changes	✓		requires mesh surgery
Error analysis	✓	✓	

96). Comparing FEM methods with our specifications we identify a few key challenges. First, the numerical evaluation of smooth geometric measurements on arbitrary manifolds in an FEM framework requires non-intuitive tensor algebra to translate the shape equation in coordinate where it is ready to be solved. After this formulation, solving the shape equation can require the use of high order function basis such as the C^1 conforming FEM based on subdivision scheme (76, 77) or isogeometric analysis (IGA) (79–81, 83), which adds code complexity and run-time cost. Extending an FEM framework to incorporate new physics, topological changes, or interfaces with other models requires advanced mathematical and coding skills. This can restrict the usage to the computational math community and prevent broad usage by the biophysics community.

Finally, evaluating discrete mesh-based methods, which define the system energy and/or forces using geometric primitives from a mesh, we find that they satisfy many of the requirements in Table 1. Due to the ease of use and implementation, discrete mesh models have gained in popularity and many different schemes can be found in the literature (60, 84–94, 97, 98). These schemes differ in their approach to defining and computing geometric measurements necessary for defining the energy and forces on a discrete object. Discrete geometries have discontinuities and limited information that leads to degenerate definitions for geometric values. For example, there is no canonical definition for the normal of a vertex of a mesh as opposed to the normal of a smooth geometry (87, 99, 100). One challenge for selecting the suitable formulation to use is the lack of approximation error metric for which the discrete definition best matches the smooth theory. Another confounding factor is the step at which the problem is discretized. Some implementations discretize the energy of the system by constructing standalone discrete energy, which captures the behavior of the Helfrich energy (65). From this discrete energy, they take the shape derivatives to obtain an expression for the discrete force. Without careful consideration, the discrete forces derived in this manner are unstructured and there is little resemblance to expressions of force from smooth theory. A second option is to discretize the smooth force expression directly(65, 95). While this preserves the geometric connection for the forces, there is no longer well-defined discrete energy. Several discrete mesh methods were benchmarked by Bian *et al.* (87) and Guckenberger *et al.* (95) who found differences in the accuracy, robustness, and ease of implementation (87,

95).

1.3 Goals of the current work

In this work, we outline a discrete mesh framework for modeling membrane mechanics with the following goals in mind: (a) we do not make *a priori* assumptions about axes of symmetry or restrict the coordinates in any way; (b) we resolve the ambiguity in the definition of geometric measurements on the mesh and permit a direct comparison for both the energy and force expressions in smooth and discrete contexts; and (c) this framework allows for use of meshes generated from ultrastructural imaging. We begin by defining discrete energy that is analogous to the Helfrich energy. Then using concepts from Discrete Differential Geometry (DDG), we derive discrete shape derivatives and group terms to produce a discrete shape equation. We will show that our discrete shape equation has a clear correspondence between the terms of the smooth shape equations (57, 67, 70, 71). Beyond establishing this important connection, we will show that the elegant analytical expressions for discrete variational terms from the DDG also yield improved geometric intuition and numerical accuracy (99, 100).

Benchmarking of our expressions was performed with our accompanying software implementation called Membrane Dynamics in 3D using Discrete Differential Geometry (`Mem3DG`). `Mem3DG` is written in C++, released under the Mozilla Public License version 2, and comes with accompanying documentation and tutorials which can be accessed on GitHub (`https://github.com/RangamaniLabUCSD/Mem3DG`). Beyond the computation of discrete energies and forces on a mesh of interest, we also include functionality for performing energy minimization and time integration. Using `Mem3DG`, we validate the exactness of the analytical expressions of force terms by numerically examining the convergence of the force terms as a function of system energy perturbation. To illustrate compliance with our tool specifications, we apply `Mem3DG` to a sequence of examples with increasing complexity. Finally, we outline the steps to incorporate additional physics such as membrane-protein interactions and surface diffusion into `Mem3DG`.

2 Methods

The lipid bilayer is modeled as a thin elastic shell using the Helfrich-Canham-Evans Hamiltonian or spontaneous curvature model (39, 69, 101). The bending energy, E_b, of a smooth surface or 2-manifold, $\mathcal{M}$, can be expressed in terms of the mean H, Gaussian K, and spontaneous curvature $\bar{H}$ with material parameters κ the bending and κ_G the saddle-splay moduli. Additional energy terms E_s and E_p account for the tension–area (λ–A) and pressure–volume (ΔP–V) relationships; The total energy of the bilayer is therefore

$$E = \underbrace{\int_{\mathcal{M}} \left[\kappa(H-\bar{H})^2 + \kappa_G K\right] dA}_{E_b} + \underbrace{\int_{\bar{A}}^{A} \lambda d\tilde{A}}_{E_s} - \underbrace{\int_{\bar{V}}^{V} \Delta P d\tilde{V}}_{E_p}. \tag{1}$$

The preferred surface area and volume, $\bar{A}$ and $\bar{V}$, combined with the spontaneous curvature, $\bar{H}$, characterize the zero-energy state for the system energy. In a nutshell, given the material properties, the system energy is fully determined by its geometric measurements such as volume, area, and curvatures.

Machinery to express these measurements have been a topic of extensive study in classical differential geometry (102, 103). However, finding the minima of the governing energy, solving the stationary solution to the geometric Partial Differential Equation (PDE), can be mathematically and numerically difficult. While differential geometry provides succinct expressions to describe the measurements in a coordinate-free fashion, computational methods often require the introduction of a coordinate basis and subsequent manipulation of expressions using tensor algebra, which can obscure the underlying geometric intuition.

As an alternative, forgoing the need for a smooth geometry, one can treat a discrete geometry (such as a geometric mesh) as the input. This perspective where the discrete geometry is the actual geometry is that of DDG(104). By eliminating the burdens of treating the input mesh as an approximation of a smooth object, DDG capitalizes upon the piecewise nature of meshes to produce efficient and parallelizable finite difference-like

formulae which are amenable to algorithmic implementation while maintaining clear geometric meaning. In the following sections, we use concepts from DDG to formulate a discrete analog to the smooth membrane shape problem. Following the derivation of the discrete theory, we describe the development of an accompanying software implementation called `Mem3DG`.

Table 2: Glossary of commonly used symbols and conventions

A. Geometric primitives:

$\mathcal{M}$	smooth or discrete 2-manifold
$\vec{r} \in \mathbb{R}^3$	embedded coordinate of $\mathcal{M}$
l	edge length
$\angle$	corner angle
φ	dihedral angle
A	area of mesh cell, e.g., face A_{ijk}, edge A_{ij} and vertex A_i
$\vec{n}$	surface normal

B. Surface Integral:

$\int a$	integrated quantity over mesh cell e.g., $A_i a_i$ or $A_{ijk} a_{ijk}$
$\sum_{v_i}$	sum over all vertices v_i of the mesh
$\sum_{e_{ij}}$	sum over all edges e_{ij} of the mesh
$\sum_{f_{ijk}}$	sum over all faces f_{ijk} of the mesh
$\sum_{v_j \in N(a)}$	sum over the vertex v_j in the neighborhood of a
$\sum_{e_{ij} \in N(a)}$	sum over the edges e_{ij} in the neighborhood of a
$\sum_{f_{ijk} \in N(a)}$	sum over the face f_{ijk} in the neighborhood of a

C. Tensors:

$x \in \mathbb{R}$	scalar quantity
$x^{\text{type}}_{\text{index}}$	sub- and super-script convention e.g., $\int \vec{f}_i^b$ is the bending force for vertex i
$\vec{x} \in \mathbb{R}^3$	vector quantity
$\boldsymbol{x} = \{x_i\}$	$(n \times 1)$ indexed scalar quantity
$\vec{\boldsymbol{x}} = \{\boldsymbol{x}_i\}$	$(n \times 3)$ indexed vector quantity
$\tilde{\boldsymbol{X}}$	matrix or tensor quantity

D. Derivatives:

$\nabla_{\vec{r}}$	shape derivative
∇_{ϕ}	chemical derivative
$\nabla_{\vec{\theta}}$	surface gradient
$\dot{a}$	time derivative
Δ_s	Laplace-Beltrami operator

E. Physical Variables:

E	energy
f	force density
μ	chemical potential
H	mean curvature
K	Gaussian curvature
A	surface area
V	enclosed volume
$\bar{\cdot}$	preferred state e.g., $\bar{H}$ is the spontaneous curvature
$\phi \in [0, 1]$	protein density parameter
λ	membrane tension
ΔP	osmotic pressure across the membrane
κ	bending rigidity
κ_G	Gaussian modulus
K_A	stretching modulus
K_V	osmotic strength constant
$\bar{c}$	molar ambient concentration
n	molar quantity of enclosed solute
η	Dirichlet energy constant
ε	adsorption energy constant
ξ	membrane drag constant
B	protein mobility constant

2.1 Notation and preliminaries

We assume the following notation conventions and provide a table of important symbols (Table 2). To aid the reader on how the elements of the mesh are used in the derivation, several fundamental geometric primitives (i.e., values on a mesh which are easily measurable; listed in Table 2A) are illustrated in Fig. 1A-C.

We note that in discrete contexts the notation, $\int a$, should be considered the discrete (integrated) counterpart of a pointwise measurement a in a smooth setting. The rationale and significance behind the usage of an integrated quantity in discrete contexts are elaborated in Appendix B and the DDG literature (*99*, *100*). Using this notation, discrete surface integrals are expressed as sums of integrated values over the discrete mesh components listed in Table 2B (e.g., $\sum_{v_i} \int a_i$ is the discrete analog to $\int_{\mathcal{M}} a$). It is possible to interchange

between integrated, $\int a_i$, and pointwise, a_i, quantities by using the dual area (A_i),

$$a_i = \int a_i / A_i. \tag{2}$$

For simplicity, we will not use separate notations for operators applying in smooth and discrete settings. The context can be inferred from the objects to which the operators are applied. Where it serves to improve our geometric or other intuition, smooth objects will be presented alongside discrete objects for comparison.

2.2 Obtaining a discrete energy defined by mesh primitives

Following the perspective of DDG, we restrict our input to the family of triangulated manifold meshes, $\mathcal{M}$ (i.e., discrete 2-manifolds embedded in $\mathbb{R}^3$)[3].

Paralleling the smooth Helfrich Hamiltonian (Eq. (1)), a functional of geometric measurements of a surface, the discrete Helfrich Hamiltonian is composed of discrete analog of those measurements,

$$E(\vec{\boldsymbol{r}}) = \underbrace{\sum_{v_i} \left[\kappa_i \int (H_i - \bar{H}_i)^2 + \kappa_G \int K_i \right]}_{E_b} + \underbrace{\int_{\bar{A}}^{A} \lambda(\tilde{A}; \vec{\boldsymbol{r}})\, d\tilde{A}}_{E_s} - \underbrace{\int_{\bar{V}}^{V} \Delta P(\tilde{V}; \vec{\boldsymbol{r}})\, d\tilde{V}}_{E_p}. \tag{3}$$

In comparison with Eq. (1), H_i and K_i are pointwise mean and Gaussian curvature measurements on vertices, $\int (H_i - \bar{H}_i)^2$ is the integrated Willmore measure, and the smooth surface integral is replaced by its discrete analog (i.e. finite summation), $\sum_{v_i}$ (Table 2B).

The geometric properties of a given membrane configuration can be connected to the system's energy through constitutive relations. In this work, we assume that the surface tension follows a linear stress-strain model(*105*),

$$\lambda(A; \vec{\boldsymbol{r}}) = K_A \frac{A(\vec{\boldsymbol{r}}) - \bar{A}}{\bar{A}}, \tag{4}$$

where $\bar{A}$ is the preferred surface area of the membrane, and K_A is the stretching modulus of the membrane. The osmotic pressure can be defined based on the van't Hoff formula as

$$\Delta P(V; \vec{\boldsymbol{r}}) = P_{\text{in}} - P_{\text{out}} = iRT \left(\frac{n}{V} - \bar{c} \right), \tag{5}$$

where i, R, T, $\bar{c}$ and n are the van't Hoff index, ideal gas constant, temperature, ambient molar concentration, and molar amount of the enclosed solute. Substituting these constitutive relations (Eqs. (4) and (5)) into the energy (Eq. (3)), we get explicit expressions for E_s and E_p,

$$E(\vec{\boldsymbol{r}}) = E_b(\vec{\boldsymbol{r}}) + \underbrace{\frac{1}{2} K_A \frac{[A(\vec{\boldsymbol{r}}) - \bar{A}]^2}{\bar{A}}}_{E_s} + \underbrace{iRTn\,[r_c - \ln r_c - 1]}_{E_p}, \tag{6}$$

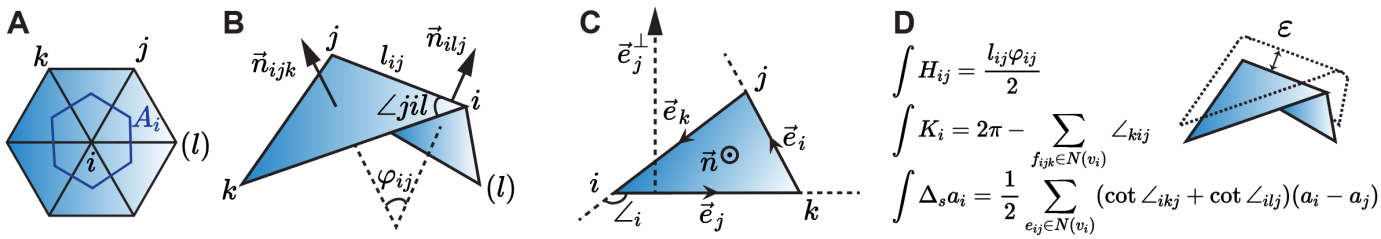
where $r_c = \bar{c}/(n/V)$ is the ratio of the concentrations of the ambient and enclosed solutions. Note that the preferred volume, $\bar{V}$, which is needed to evaluate the integral in Eq. (3), is related to to the parameters in Eq. (5) by $\bar{V} = n/\bar{c}$. If the system is around the isosmotic condition (e.g., $V \to \bar{V}$), the leading order of the energy is given as,

$$E_p \approx \frac{1}{2} K_V \frac{(V - \bar{V})^2}{\bar{V}^2}, \tag{7}$$

where $K_V \equiv iRTn$ groups the phenomenological parameters.

To compute the energy of a system, we must obtain values for several geometric measurements which appear in the discrete energy function (i.e., H, K, A, V, etc.). For measurements such as the volume and area, there are clear approaches for their evaluation on a triangulated mesh: summing the areas of the triangular

[3] We will use $\mathcal{M}$ for both the smooth and discrete surfaces.



E	Smooth	Discrete
Osmotic pressure	$\nabla_{\vec{r}}$ $\mathrm{vol}(\mathcal{M}) = \frac{1}{3}\int_{\mathcal{M}} \vec{n}\cdot\vec{r}\, dA$ $\vec{n}\, dA$	$\sum_{f_{ijk}} V_{0ijk}$ $\int \vec{n}_i = \frac{1}{3}\sum_{f_{ijk}\in N(v_i)} \int \vec{n}_{(ijk)} = \frac{1}{3}\sum_{f_{ijk}\in N(v_i)} A_{ijk}\vec{n}_{(ijk)}$
Surface tension	$\nabla_{\vec{r}}$ $\mathrm{Area}(\mathcal{M}) = \int_{\mathcal{M}} dA$ $2H\vec{n}\, dA$	$\sum_{f_{ijk}} A_{ijk}$ $\int 2\vec{H}_i = \sum_{e_{ij}\in N(v_i)} \int 2\vec{H}_{ij} = \frac{1}{2}\sum_{e_{ij}\in N(v_i)} (\cot\angle_{ikj} + \cot\angle_{ilj})\vec{e}_{ji} = \int \Delta_s \vec{r}_i$
Bending	$\nabla_{\vec{r}}$ $\mathrm{Mean}^2(\mathcal{M}) = \int_{\mathcal{M}} H^2\, dA$ $\left[2H\left(K - H^2\right) + \Delta_s H\right]\vec{n}dA$	$\sum_{v_i} \int H_i^2 = \sum_{v_i} \frac{1}{A_i}\left(\int H_i\right)^2$ $\sum_{e_{ij}\in N(v_i)} (H_i + H_j)\int \vec{K}_{ij} - \frac{H_i^2 + 2H_j^2}{3}\int 2\vec{H}_{ij} + \left(H_i \int \vec{S}_{ij,1} + H_j \int \vec{S}_{ij,2}\right)$
	$\nabla_{\vec{r}}$ $\mathrm{Gauss}(\mathcal{M}) = \int_{\mathcal{M}} K\, dA$ Gauss-Bonnet Assume closed 0	$\sum_{v_i} \int K_i$ Gauss-Bonnet Assume closed 0

Figure 1: Overview of the DDG framework. A, B, C) Illustrations of geometric primitives in the neighborhood of A) Fan around a vertex, B) Diamond around an edge, and C) Triangle on a face. D) Discrete definition of scalar edge mean curvature, $\int H_{ij}$, scalar vertex Gaussian curvature, $\int K_i$, and Laplace-Beltrami operator, $\int \Delta_s(\cdot)$. E) Comparative derivation of Helfrich shape equation in both smooth and discrete formulation.

faces and summing over the signed volume of tetrahedra (Fig. 1E – osmotic pressure and surface tension). For other measurements such as the discrete mean and Gaussian curvatures, additional care must be taken. While in smooth contexts these curvatures have unique definitions, in discrete contexts there are multiple approaches for their calculation. For example, the mean curvature can be computed via the application of the cotangent Laplacian, the kernel of the heat equation, or fitting polynomials to a local patch (*65*). As mentioned earlier, there are challenges with these approaches which can limit their numerical accuracy and obscure the connection to smooth theory. Here using discrete exterior calculus and identification of geometric invariants, we produce theoretically and numerically consistent discrete expressions.

Similar to the polygonal curve introduced in Appendix B, a triangulated mesh has zero curvature on facets and ill-defined curvature on edges and vertices. Using the Steiner view, the sharp corners formed by vertices and edges are made smooth with portions of spherical and cylindrical shells, which have well-defined mean curvature (Fig. 1D). Taking the limit as the radii of the cylinders and spheres decrease, the leading order contribution of total mean curvature is given by the Steiner formula on an edge,

$$\int H_{ij} = \frac{l_{ij}\varphi_{ij}}{2}, \tag{8}$$

referred to as the edge mean curvature, where l_{ij} is the length of edge e_{ij}, and φ_{ij} is the dihedral angle on e_{ij} (i.e., the angle formed by the face normals of the neighboring triangles incident to e_{ij}) (illustrated in Fig. 1B)(*99*, *100*). While not necessary, a triangulated mesh is often realized in $\mathbb{R}^3$ via vertex positions; thus it is conventional to prescribe data on vertices instead of edges. Summation of edgewise quantities over the "fan" neighborhood (Fig. 1A) provides the recipe of converting an edgewise to a vertexwise quantity,

$$(\cdot)_i = \frac{1}{2} \sum_{e_{ij} \in N(v_i)} (\cdot)_{ij}, \tag{9}$$

where the prefactor, $1/2$, accounts for fact that each edge is shared by two vertices.

While we have an integrated mean curvature, the discrete Helfrich Hamiltonian contains a pointwise mean curvature squared term. To define a pointwise mean curvature, the size of the domain *occupied* by the integrated mean curvature needs to be specified (c.f., Appendix B for rationale). The area, A_i, referred to as the dual area of the vertex v_i, can be defined as one-third of the areal sum of the incident triangles ("fan" illustrated in Fig. 1A)(*99*, *100*). Applying Eqs. (2) and (9) to Eq. (8), the pointwise mean curvature is thus,

$$H_i = \frac{\int H_i}{A_i} = \sum_{e_{ij} \in N(v_i)} \frac{l_{ij}\varphi_{ij}}{4A_i}. \tag{10}$$

Substituting Eq. (10) into the integrated Willmore measure term of Eq. (3), the integrated Willmore measure can be expressed as a function of the integrated mean and spontaneous curvature,

$$\int (H_i - \bar{H}_i)^2 = \frac{1}{A_i} \left(\int H_i - \int \bar{H}_i \right)^2. \tag{11}$$

Discrete Gaussian curvature is given by the angle defect formula,

$$\int K_i = 2\pi - \sum_{f_{ijk} \in N(v_i)} \angle_{kij}, \tag{12}$$

which is a well-known quantity that preserves many properties parallel to the smooth theory (e.g., Gauss-Bonnet, turning number, among other invariants). One way to derive the angle defect formula is to compute the area of a spherical n-gon contained by a local Gauss map of the neighboring n faces around a vertex (*99*, *100*).

Eq. (12) provides the general geometric definition to obtain the energetic contributions from the Gaussian curvature terms. In this study, we consider only systems with uniform saddle-splay modulus which do not

undergo topological changes. For these systems, the energy can be simplified based on the discrete Gauss-Bonnet theorem, which states that

$$\sum_{v_i} \int K_i = 2\pi\chi(\mathcal{M}) - \sum_{v_j \in \partial\mathcal{M}} \int \kappa_j^g, \tag{13}$$

where $\chi(\mathcal{M}) = |V| - |E| + |F|$ is the Euler characteristic of $\mathcal{M}$, a topological invariant where $|V|$, $|E|$ and $|F|$ represent the number of vertices, edges and faces of the mesh respectively, and $\int_{\partial\mathcal{M}} \kappa_i^g = \pi - \sum_{e_{ij} \in N(v_i)} \angle kij$ is the discrete *geodesic* curvature which measures the deviation of the boundary curve from a straight line when the surface is locally flattened. In summary, for this work, the Gaussian curvature term is non-constant only when $\mathcal{M}$ is not closed, and the energy solely involves the boundary elements.

A numerical comparison of the discrete scalar measurements with their smooth counterparts is shown in Fig. E.1. We note that for all geometric measurements (i.e., volume, area, and curvatures), unlike in smooth differential geometry where their numerical evaluation requires the introduction of coordinates, DDG measurements are functions of mesh primitives. By substituting these discrete geometric measurements from DDG into Eq. (6) and Eq. (3), we get a numerical recipe for computing the total system energy.

2.3 Force from discrete shape derivative of energy

We can obtain the force by taking the negative shape derivative of the energy. In continuous settings, the differentiation is an infinite-dimensional problem that requires the use of the calculus of variations and differential geometry to find analytical expressions (*39*, *43*, *70*, *71*) (Fig. 1E – smooth). Deriving the forces from the discrete energy (Eq. (3)) is a much simpler task.

Discrete forces can be obtained by taking partial derivatives of mesh primitives with respect to the vertex embedded coordinates, $\vec{r}$ (Fig. 1E – discrete). Regarding notation, despite the computational differences, the differential operations in both the discrete and smooth contexts are called (discrete) shape derivatives and denoted as $\nabla_{\vec{r}}(\cdot)$ due to the common geometric meaning. We note that the computation of discrete shape derivatives for membrane modeling has been described previously in the literature (*85*, *87*). Also that there are many overlapping definitions for discrete curvature, energy, and variations thereof in the graphics literature (*106*–*108*). Our work extends upon the prior art which evaluates derivatives algebraically, by introducing simplifications based upon the grouping of terms and identification of geometric objects. These simplifications have important implications for improving the geometric understanding as well as the run-time and numerical performance of an implementation.

At the high level, our goal is to express the forces on each vertex, given a set of physics, using geometric primitives and properties defined on specific mesh elements. By grouping terms, we find that the vertex-wise forces arising from the different physics can be expressed as weights which are functions of the input parameters and system configuration, multiplied by basic geometric vectors. We will show that these terms have an exact correspondence to terms in the smooth shape equation (Fig. 1E). We remark that, in some sense, the force expressions are reminiscent of finite difference equations which approximate differentials as a linear combination of values at discrete points. This may have implications for the suitability of modeling smooth biological surfaces with discrete meshes.

2.3.1 Force from osmotic pressure

For the smooth geometry, the shape derivative of the enclosed volume yields the outward pointing surface normal with its size equal to the local area element (*109*). For a discrete mesh, the shape derivative of the volume is given by the face normal on *triangular faces* with its local area element equaling to the *face area*, which is referred to as the *integrated* face normal, $\int \vec{n}_{(ijk)}$ (Fig. 1E – osmotic pressure) (*87*, *99*, *100*), where (ijk) denotes the symmetry under index permutation (e.g., $a_{i(jk)}$ means $a_{ijk} = a_{ikj}$). Similar to edge values, the force normal can be converted to vertex normal,

$$\int \vec{n}_i = \nabla_{\vec{r}_i} V = \frac{1}{3} \sum_{f_{ijk} \in N(v_i)} \int \vec{n}_{(ijk)} = \frac{1}{3} \sum_{f_{ijk} \in N(v_i)} A_{ijk} \vec{n}_{(ijk)}, \tag{14}$$

where analogous to Eq. (9), the prefactor $1/3$ accounts for fact that each face is shared by three vertices. The discrete vertex forces from the derivative of the pressure-volume work, $\int \vec{f}_i^p$, is then given by scaling it with the uniform osmotic pressure,

$$\int \vec{f}_i^p = \Delta P \int \vec{n}_i. \tag{15}$$

2.3.2 Forces from surface tension

Next considering the shape derivative of the surface energy, E_s, in smooth contexts, the derivative of the total surface area also points at the surface normal, with its magnitude measuring the size (dA) and the curvature ($2H$) of the local patch (Fig. 1E – surface tension) (*109*). In discrete case, we can directly compute the derivative of total area on each vertex by summing the area gradient of incident triangles with respect to the vertex position; The sum is therefore referred as (twice of) the integrated mean curvature vector on vertices,

$$\int 2\vec{H}_i = \nabla_{\vec{r}_i} A = \sum_{f_{ijk} \in N(v_i)} \nabla_{\vec{r}_i} A_{ijk} = \sum_{f_{ijk} \in N(v_i)} \int 2\vec{H}_{i(jk)}, \tag{16}$$

where we define $\int 2\vec{H}_{i(jk)} \equiv \nabla_{\vec{r}_i} A_{ijk}$, and $\int \vec{H}_{i(jk)}$ is the mean curvature vector on a triangle face corner. The capillary forces from surface tension, $\int \vec{f}_i^s$, are given by scaling the integrated mean curvature vector by the surface tension,

$$\int \vec{f}_i^s = -\lambda \int 2\vec{H}_i. \tag{17}$$

Evaluating the algebraic sum of area gradients reveals the "cotangent formula" applied to the vertex positions (Fig. 1E – surface tension). From independent derivations with unrelated frameworks (e.g., discrete exterior calculus and FEM), discretizing the smooth Laplace-Beltrami operator on a triangulated mesh produces the cotangent formula which is called the discrete Laplace-Beltrami operator, $\int \Delta_s$(*99*, *100*, *106*). Inspecting the expressions in Fig. 1E – surface tension, we see that our discrete expression parallels smooth theory, where the mean curvature is related to the coordinates through the application of the smooth Laplace-Beltrami operator,

$$\sum_{e_{ij} \in N(v_i)} \int 2\vec{H}_{ij} = \int \Delta_s \vec{r}_i \quad \leftrightarrow \quad \Delta_s \vec{r} = 2H\vec{n}. \tag{18}$$

2.3.3 Forces from bending

To evaluate the shape derivative of the discrete bending energy we consider the terms from the Gaussian and mean curvature separately. Since we do not consider non-uniform saddle-splay modulus and topological changes in this work, the total Gaussian curvature only varies if the surface has boundaries, $\partial\mathcal{M}$ (c.f., discrete Gauss-Bonnet theorem in Section 2.2). The shape derivative of the bending energy requires more algebra and the introduction of halfedges, $\underline{e}_{ij}$ (c.f., Appendix C.1). Here we will focus on the key results and refer the reader to the full derivations for each term in Appendix C.2.

There are four fundamental geometric vectors on halfedges that comprise the bending force at non-boundary vertices: the mean curvature vector (see Fig. 1B for indices),

$$\int 2\vec{H}_{ij} = \frac{1}{2}\left(\int 2\vec{H}_{i(jk)} + \int 2\vec{H}_{i(jl)}\right); \tag{19}$$

the Gaussian curvature vector,

$$\int \vec{K}_{ij} = \frac{1}{2}\varphi_{ij} \nabla_{\vec{r}_i} l_{ij}; \tag{20}$$

and the two Schlafli vectors,

$$\int \vec{S}_{ij,1} = \frac{1}{2} l_{ij} \nabla_{\vec{r}_i} \varphi_{ij}, \qquad \int \vec{S}_{ij,2} = \frac{1}{2} \left(l_{jk} \nabla_{\vec{r}_i} \varphi_{jk} + l_{jl} \nabla_{\vec{r}_i} \varphi_{jl} + l_{ji} \nabla_{\vec{r}_i} \varphi_{ji} \right), \tag{21}$$

which act to smooth the profile of local dihedral angles. Note that the shape derivatives are taken with respect to different vertices (i.e., $\nabla_{\vec{r}_i}$ or $\nabla_{\vec{r}_j}$), such that the mean curvature $\int \vec{H}_{ij}$, Gaussian curvature $\int \vec{K}_{ij}$, and Schlafli vectors $\int \vec{S}_{ij}$ are asymmetric under index permutation. To account for the orientation we refer to them as *halfedge* vector quantities on $\underline{e}_{ij}$ (Appendix C.1). A numerical comparison of the discrete geometric vector with their smooth counterparts is shown in Fig. E.1 and Fig. E.2.

The bending force $\int \vec{f}_i^b$ (Fig. 1E – bending) can be expressed as weights which are functions of input parameters multiplied by basic geometric measurements in scalar and vector form,

$$\begin{aligned} \int \vec{f}_i^b = \sum_{\underline{e}_{ij} \in N(v_i)} & - \left[\kappa_i (H_i - \bar{H}_i) + \kappa_j (H_j - \bar{H}_j) \right] \int \vec{K}_{ij} \\ & + \left[\frac{1}{3} \kappa_i (H_i - \bar{H}_i)(H_i + \bar{H}_i) + \frac{2}{3} \kappa_j (H_j - \bar{H}_j)(H_j + \bar{H}_j) \right] \int 2\vec{H}_{ij} \\ & - \left[\kappa_i (H_i - \bar{H}_i) \int \vec{S}_{ij,1} + \kappa_j (H_j - \bar{H}_j) \int \vec{S}_{ij,2} \right], \end{aligned} \tag{22}$$

which parallels the shape derivative of the smooth bending energy,

$$\begin{aligned} \nabla_{\vec{r}}^{\perp} E_b &= \nabla_{\vec{r}}^{\perp} \left[\int_{\mathcal{M}} \kappa (H - \bar{H})^2 dA \right] \\ &= \kappa \left[2(H - \bar{H}) \left(H^2 - K + \bar{H} H \right) + \Delta_s (H - \bar{H}) \right] dA, \end{aligned} \tag{23}$$

where $\nabla_{\vec{r}}^{\perp} = \nabla_{\vec{r}} \cdot \vec{n}$ is the shape derivative in the surface normal direction.

Comparing the smooth-discrete expressions, we make a few observations:

- The Schlafli vector terms, $\vec{S}$, is the discrete analog of the smooth biharmonic term, $\Delta_s(H - \bar{H})$, the high-order local smoothing force. The numerical comparison of these two terms, as well as results directly obtained using cotangent formula applied on pointwise scalar mean curvature, are covered in Fig. E.2 and Fig. E.1.
- Eq. (23) is the normal component of the shape derivative of the bending energy; an additional tangential component is required if surface heterogeneity exists (e.g. κ is not spatially uniform) (*40*, *65*). In contrast, the discrete shape derivative (Eq. (22)) is the total derivative in $\mathbb{R}^3$, which includes both the tangential and normal components[4]. Depending on the extent and symmetry of the heterogeneity, the discrete force can point in any direction in $\mathbb{R}^3$.
- The coefficients in Eq. (22) shows an intriguing pattern combining contribution from both v_i and v_j. From a finite-difference approximation standpoint, this results in an approximation scheme for which a rigorous error analysis has not yet been conducted.

2.3.4 Net force and the benefit of DDG

Summing the force terms from each physics, we obtain a net force. The expression for the total force is a function of geometric primitives and was derived using concepts from DDG without the need to introduce coordinates and use tensor algebra. The numerical performance of these expressions are benchmarked for several scalar and vector measurements on smooth and discrete surfaces shown in Fig. E.1, Fig. E.2, and later discussed in Section 2.5.4. Also important, our resulting expressions allow the direct comparison of terms between smooth and discrete contexts. By choosing definitions that preserve the chain of shape derivatives for the geometric vectors of interest (c.f., Fig. A.2), we can preserve geometric intuition between

[4] In the smooth sense since there is no well-defined vertex normal direction in discrete geometry

smooth and discrete differential geometry theory (*100*). With respect to the numerical performance, since the terms of the discrete shape equation are defined using values from neighborhoods around individual vertices the algorithms are efficient and straightforward to parallelize. Owing to the local nature of the force evaluation, heterogeneities in material and other properties across the membrane are also straightforward to incorporate.

2.4 Defining metrics for simulation and error quantification

For monitoring simulation progress, exactness of force calculations with respect to the discrete energy, and convergence studies of computed quantities upon mesh refinement we introduce the following norms.

2.4.1 L2 norm

From a PDE perspective, the vertex forces are also called the residual of the shape equation, whose solution represents the equilibrium solution. The simulation is terminated when the residual is smaller than a user-specified threshold. The rationale for using the L_2 norm is justified by perturbing the system configuration and conducting an expansion on the system energy,

$$\begin{aligned} E(\vec{\boldsymbol{r}} + \epsilon \nabla E(\vec{\boldsymbol{r}})) &= E(\vec{\boldsymbol{r}}) + \epsilon \langle \nabla E(\vec{\boldsymbol{r}}), \nabla E(\vec{\boldsymbol{r}}) \rangle + \mathcal{O}\left(\epsilon^2\right) \\ &= E(\vec{\boldsymbol{r}}) + \epsilon \left\| \int \vec{\boldsymbol{f}} \right\|_{L_2}^2 + \mathcal{O}\left(\epsilon^2\right), \end{aligned} \tag{24}$$

where we refer the inner product of the force matrix as the L_2 norm of the forces. Computationally, this is equivalent to the standard Frobenius matrix L_2 norm,

$$\left\| \int \vec{\boldsymbol{f}} \right\|_{L_2} = \sqrt{\text{trace}\left(\int \vec{\boldsymbol{f}}^\top \int \vec{\boldsymbol{f}} \right)}. \tag{25}$$

Using the L_2 norm and Eq. (24), we can perform a numerical validation of the exactness of the discrete force calculation with respect to the discrete energy. We expect the force to approximate the energy up to 2nd order with respect to the size of a perturbation. This validation will be further elaborated in Section 3.4.

2.4.2 L1 norm

A scale-invariant L_1 norm is well-suited to quantify the magnitude of the error on varying domain size and mesh resolution. Given a vertexwise local scalar measurement, $\int \boldsymbol{a}$, or a vector measurement, $\int \vec{\boldsymbol{a}}$, and their reference values, $\int \bar{\boldsymbol{a}}$, and $\int \bar{\vec{\boldsymbol{a}}}$,

$$\begin{aligned} \left\| \int \boldsymbol{a} \right\|_{L_1} &= \frac{\sum_{v_i} \left| \int a_i - \int \bar{a}_i \right|}{A} \\ \left\| \int \vec{\boldsymbol{a}} \right\|_{L_1} &= \frac{\sum_{v_i} \left\| \int \vec{a}_i - \int \bar{\vec{a}}_i \right\|_{L_2}}{A}, \end{aligned} \tag{26}$$

where the normalizing factor, the total surface area A, is used to obtain a pointwise estimate of the error. The L_1 norm is applied in the local comparison of discrete and smooth measurements, where we further elaborate in Section 2.5.4.

2.5 Software implementation – `Mem3DG`

Along with the theoretical developments, we have developed an accompanying software implementation written in C++ called `Mem3DG`. Our goal in developing this software is to enable the easy use and application of the corresponding theory developed above to biological problems of interest.

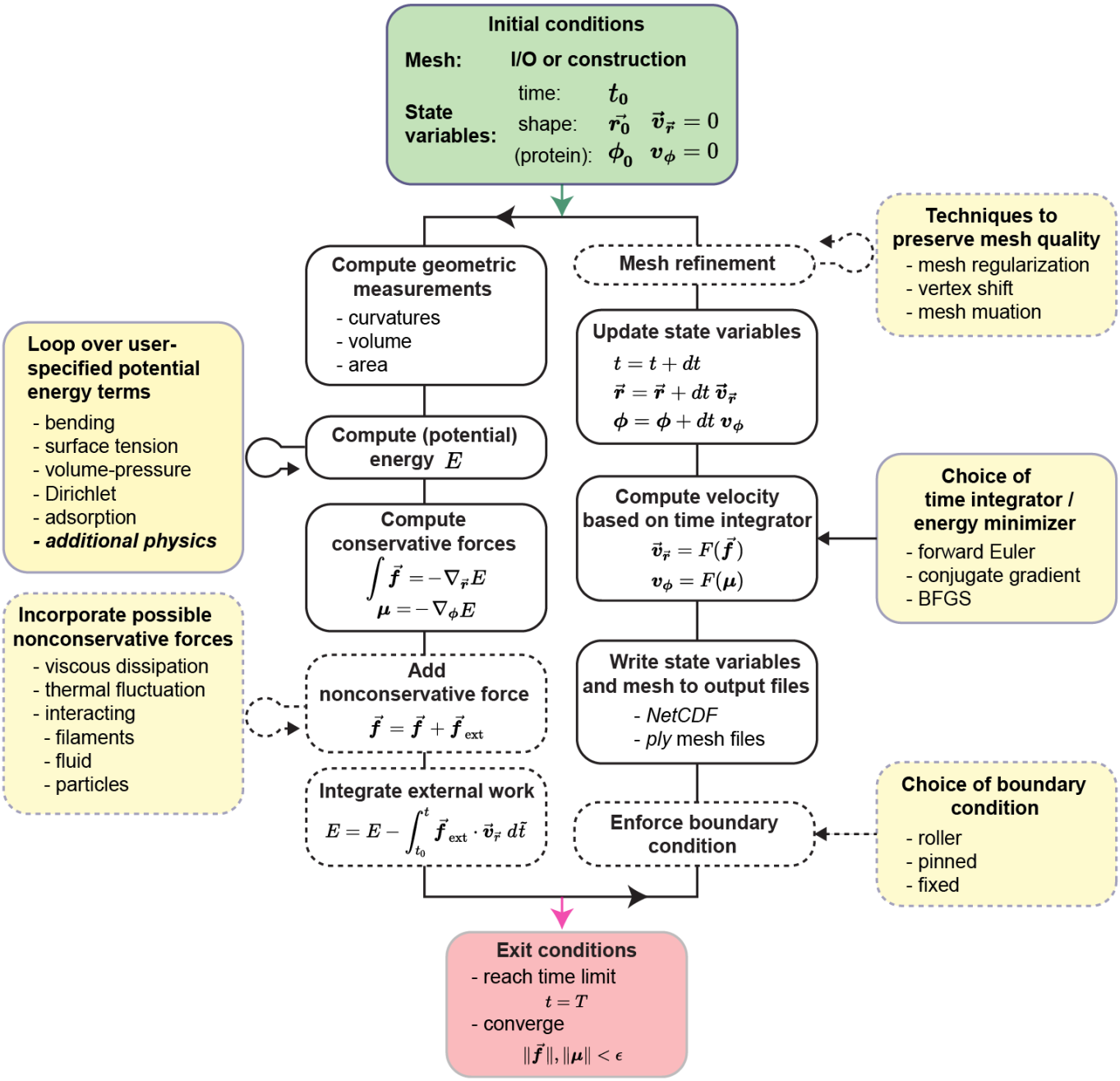


Figure 2: Overview of data flow within `Mem3DG`. The user provides `Mem3DG` with an initial condition in the form of a triangulated mesh and vertexwise state and kinematic variables (green box). The main loop (black loop) of `Mem3DG` evaluates the discrete energy and forces and propagates the trajectory, among other supporting steps. Modules in dashed lines are optional depending on whether the system mesh has boundaries and if external forces are specified. User-specified options and possible extensions of `Mem3DG` to accommodate various physics are highlighted in yellow boxes. `Mem3DG` automatically exits the simulation when the system converges or the maximum time step is reached.

Mem3DG is a library that contains several components to support this goal. Fig. 2 provides a synopsis of Mem3DG. The input to Mem3DG includes a triangulated mesh with its coordinate $\vec{r}$ embedded in $\mathbb{R}^3$. Users can choose to use Mem3DG to construct idealized meshes (e.g., icosphere, cylinder, or flat hexagonal patch) as an input or to read in meshes from several common mesh formats. Meshes are stored and manipulated in Mem3DG using the half-edge data structure provided by Geometry Central (*110*). The supported input file formats are those which are readable by hapPLY and Geometry Central (*110*, *111*). Once a mesh and parameters are loaded, Mem3DG can evaluate the discrete energy and forces of the system. Mem3DG adopts a modular design that facilitates the use of different energy and force components and has utilities which help the user to specify the physics and governing parameters. Mem3DG also supports local system simulations where the input mesh has boundaries. Additional details about the supported boundary conditions are given in Section 2.5.2.

To perform energy minimization and time integration of the system, various schemes have been implemented. These schemes are described in Section 2.5.3. As discussed further in Section 2.5.4, when a user wishes to use Mem3DG to represent complex biological membrane geometries, additional care regarding the quality of the mesh is necessary. Mem3DG includes algorithms for basic mesh regularization and remeshing which can be toggled by the user to support their applications. The simulation terminates when it reaches the time limit or the system reaches equilibrium, whose criteria is determined using the energy L_2 norm introduced in Section 2.4. A user can choose between several formats to output a trajectory over time or the configuration of the local minima from Mem3DG. In addition to the mesh outputs supported by Geometry Central, we have also developed a scheme for outputting mesh trajectories in NetCDF format (*112*). Mem3DG can read and visualize the output trajectories and mesh configurations using Geometry Central and Polyscope (*110*, *113*).

For rapid prototyping and enumeration of simulation conditions, we have also developed a Python API called PyMem3DG. The functionality in C++ is exposed in Python using bindings from pybind11 (*114*). Illustrative examples of using both Mem3DG and PyMem3DG are provided in the online tutorials. For the experiments discussed in this work, all of the simulations were performed using PyMem3DG and the accompanying code and initial configurations are on GitHub: https://github.com/RangamaniLabUCSD/Mem3DG.

2.5.1 Defining properties of a membrane reservoir for systems with open boundaries

To facilitate correspondence with wet experiments and to support the reduction of computational cost, it is possible to construct systems using meshes with open boundaries in Mem3DG. For example, when modeling the formation of a small endocytic bud from a large cell, the deformation is small compared to the broader system. If we assume that the bulk of the cell is invariant with respect to bud formation, the computational burden can be reduced by modeling only the local deformation; we can assume that the modeled patch is attached to an implicit membrane reservoir. To define this coupled system, the *constant* area (A_r) and volume (V_r) of the reservoir must also be provided. The total area and volume of the broader system is given by $A = A_{\text{patch}} + A_r$, and $V = V_{\text{patch}} + V_r$, where A_{patch} and V_{patch} are area and "enclosed volume" of the mesh patch respectively. In our models, we enforce that all elements of a boundary loop are on the same plane; this way V_{patch} can be unambiguously defined as the enclosed volume when each boundary loop is closed by a planar sheet. The capability to model systems attached to a reservoir reduces the modeled degrees of freedom while enabling intuitive physics to simplify the process of mimicking experimental conditions using Mem3DG.

2.5.2 Prescribing boundary conditions with force masking

Mem3DG supports modeling membranes with and without boundaries: a sphere (with no boundaries), a disk (with 1 boundary), and an open cylinder (with 2 boundaries). For systems without boundaries, since the force calculation of the scheme is exact, the forces will not introduce artificial rigid body motions, as was also noted by Bian *et al.* (*87*). To study systems with boundaries, Mem3DG currently supports three types of boundary conditions:

- **Roller**, where the movement of boundary vertices is restricted along a given direction or plane.
- **Pinned**, where the position of boundary vertices are pinned while the curvature is allowed to vary.
- **Fixed**, where both the position and the boundary curvature are fixed for vertices on the boundary.

The different boundary conditions are achieved by masking the elements of the force matrix corresponding to the boundary vertices and their neighborhood. For example, to apply roller boundary conditions, we mask the Z–component of the force on the boundary vertices, therefore constraining their movement to the X–Y plane; pinned boundary conditions mask all force components for the boundary vertices to fix their position; fixed boundary conditions mask all force components for the outermost three layers to fix both their position and curvature.

2.5.3 Time integration and energy minimization

In this work, we use the forward Euler algorithm to integrate the system dynamics and the nonlinear conjugate gradient method to solve for equilibrium conditions. Both solvers are complemented by a backtracking line search algorithm, which satisfies Wolfe conditions to support adaptive time-stepping and robust minimization (*115*).

The forward Euler scheme was chosen as the simplest dynamical propagator; physically it represents over-damped conditions where the environment of the membrane is too viscous for the system to carry any inertia. Mathematically the physics is described by,

$$\dot{\vec{r}} = \frac{1}{\xi}\int \vec{f} = \frac{1}{\xi}\int (\vec{f}^{b} + \vec{f}^{s} + \vec{f}^{p}), \tag{27}$$

where ξ is the drag coefficient. From an optimization perspective, forward Euler is equivalent to the gradient descent method for minimizing an objective function, which is the discrete energy in our case.

A second propagator is the nonlinear conjugate gradient method for locally minimizing the discrete energy to yield the equilibrium shape of the membrane. Since the system is nonlinear we periodically perform forward Euler (gradient descent) steps after several conjugate gradient steps. This approach of iterating between conjugate gradient and gradient descent steps is commonplace in the literature for solving nonlinear systems (*115*).

We note that other time integrators and energy minimizers are also compatible with `Mem3DG`. Included in the software are reference implementations of Velocity Verlet integration (for symplectic time integration), and Limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm (L-BFGS, a quasi-Newton method to the equilibrium shape for large scale problems where fast computation is needed). We do not discuss these additional solvers in this work.

2.5.4 Practical considerations for applying `Mem3DG` to biological problems

As we have noted above, in the DDG perspective, the mesh *is* the geometry and thus the formulation of the discrete forces and energies is exact. There are therefore very few restrictions on the resolution and quality of the input mesh. However, in biophysics, we often consider biological membranes as smooth systems. We expect that many users of `Mem3DG` may wish to approximate a smooth system using our discrete model. In doing so, they make an implicit assumption that such an approximation is reasonable. Although the relationships between geometric objects and their shape are preserved between the smooth and discrete contexts, our ability to approximate a smooth problem with a discrete mesh is not guaranteed. Similar to finite differences and FEM, additional constraints on mesh quality and resolution must be imposed. To verify and understand the limitations of the assumption that the discrete mesh is the geometry and includes all of the geometric information, we numerically test the convergence of the discrete quantities under variation of resolution on an oblate spheroid mesh. The additional details regarding these numerical experiments are presented in Appendix D.

Setting the characteristic length scale of the finest mesh to be $h = 1$, as the mesh coarsens (i.e., mesh size increases) h increases. Fig. 3 shows the scaling relationship of the deviation in magnitude between the smooth and discrete quantities. Fig. 3A shows the convergence property of global measurements that determines the energy (Eq. (1) and Eq. (3)), including the total area, A, enclosed volume, V, and total Gaussian curvature and mean curvature (squared), $\int_{\mathcal{M}} K dA$, $\int_{\mathcal{M}} H dA$ and $\int_{\mathcal{M}} H^2 dA$, respectively. Except for the total

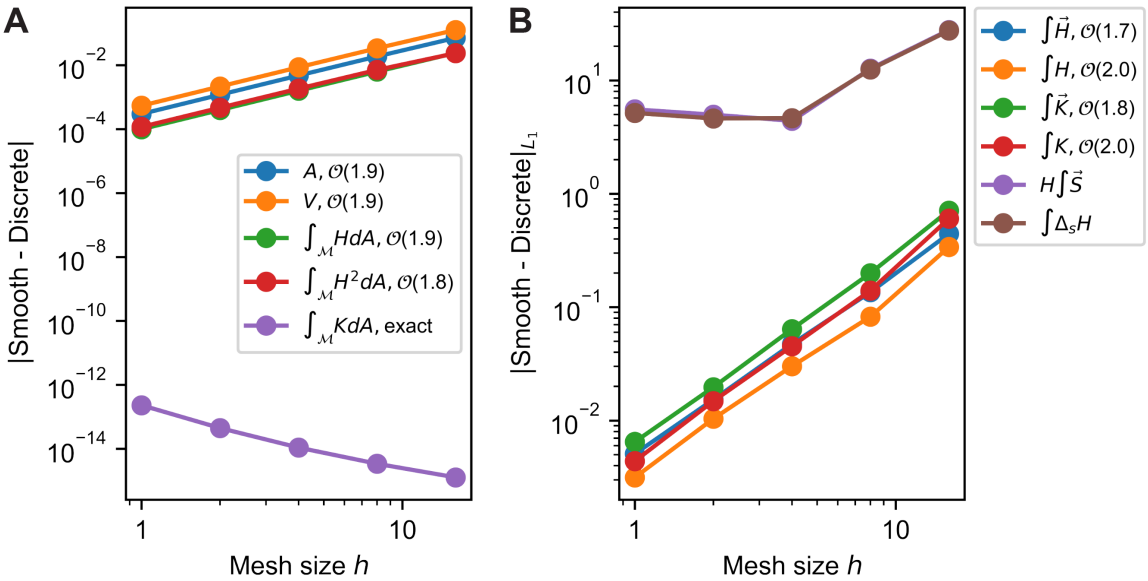


Figure 3: Comparison of discrete quantities with their smooth counterparts on spheroid shape. A) Convergence plot of global quantities, including total area, volume, mean curvature (squared), and Gaussian curvature, and B) Convergence plot of L_1 norm of scalar and vector local quantities, including the mean curvature, Gaussian curvature, and the biharmonic term.

Gaussian curvature being an exact topological invariant, all integrated quantities exhibit approximately 2nd order convergence rate.

We acknowledge that convergence of global measurements does not imply that local measurements will also converge. To validate the convergence of local measurements, which determines the convergence of local forces on the membrane (e.g., Eq. (15), Eq. (17) and Eq. (22)), we utilize the L_1 norm (Eq. (26)) to evaluate the deviation of local quantities from their smooth counterparts. Fig. 3B shows the local convergence plot. Similar to their global counterparts, local scalar mean and Gaussian curvature, $\int H$, and $\int K$, converge at $\mathcal{O}(h^2)$. Fig. 3B also shows the convergence of vector quantities, which not only contribute to the magnitude of the force but also set the direction of the force. The test shows that most vector quantities converge slightly slower than their scalar counterparts. Two terms exhibit poor convergence, the Schlafli vector term in Eq. (22), $H \int S$, and a scalar counterpart, $\int \Delta_s H$. The latter term corresponds to the direct application of the cotangent Laplacian (Eq. (17)) to the pointwise scalar mean curvature field; this approach is not used in our framework but is common in the literature(*65*). Both non-convergent expressions are discrete representations of the biharmonic term, $\Delta_s H$, which have been noted to be sensitive to noises of vertex coordinates in the prior literature (*95*). Recall that the biharmonic term is the fourth-order derivative of the embedded coordinates. Although the traditional approximation theories suggest that higher-order derivatives often exhibit slower rates of convergence (*116*), to the best of our knowledge, there is not yet a rigorous study that connects DDG with an approximation theory. Nevertheless, we anticipate that similar principles hold. Two spatial plots comparing local measurements between smooth and discrete contexts are provided in the appendix (Fig. E.1 and Fig. E.2); each test is conducted using the finest mesh size ($h = 1$). Based on this numerical validation, we conclude that the computation of energy converges with a 2nd order rate (Fig. 3A). While most components of the forces converge, the biharmonic term remains a limiting factor.

One other practical consideration for our models is that the Helfrich Hamiltonian, matching the in-plane fluidity of biological membranes, has no resistance to shearing. Without additional constraints, the mesh is susceptible to shearing motions which can deteriorate mesh quality in some conditions (*81*). This can degrade the implicit assumption that the discrete mesh represents a smooth geometry. To ensure that such an approximation can remain valid throughout a trajectory, we have incorporated algorithmic solutions to adaptively maintain an isotropically well-resolved discrete geometry. This is achieved by two operations: 1) mesh regularization using local force constraints which are common in finite element methods (*76*, *80*, *81*, *83*) Appendix E.2. and 2) mesh mutations such as decimating, flipping, and collapsing edges. Mesh mutations are also a common practice to cope with deterioration in other mesh simulations which use a Monte

Movie 1: Animated time series simulation of Fig. 4C–reference. The color map indicates the local pointwise mean curvature, H, and t represents the numerical time. Available on GitHub: https://github.com/RangamaniLabUCSD/2020-Mem3DG-Applications/blob/master/examples/videos/compressed/pearl_control_1.mp4

Movie 2: Animated time series simulation of the unduloid shape shown in Fig. 4C–medium osmolarity. The color map indicates the local pointwise mean curvature, H, and t represents the numerical time. Available on GitHub: https://github.com/RangamaniLabUCSD/2020-Mem3DG-Applications/blob/master/examples/videos/compressed/pearl_pearl_1.mp4

Carlo integration (*60*, *87*–*93*). The algorithms for mesh regularization and mutation are further described in Appendix E.

3 Results and Discussion

To further validate the method and to provide a sense of how `Mem3DG` can be used and extended to solve more complex physics, we apply `Mem3DG` to a sequence of examples with increasing complexity. First, we model well-studied systems with homogeneous membrane conditions. We show that `Mem3DG` is capable of reproducing the classical solutions without imposing the axisymmetric constraint commonly adopted by other solvers. The later examples set a blueprint for extending and modifying `Mem3DG` for particular systems of interest. We introduce new energy and corresponding force terms to expand the formulation for complex systems of interest. We emphasize that the goal of these examples is to illustrate the generality of the theory and software and to outline specific steps for future extensions; we do not perform rigorous experimental comparisons nor extract scientific insights. Additional care must be taken to mimic specific biological experiments for model validity, which is left for future work.

Each of the following sections considers a different class of membrane biophysics problem of increasing complexity in the coupling of the in-plane protein density parameter, $\phi \in [0, 1]$. To mimic the various influences protein-lipid interactions can have on the bilayer, the protein density can be set to influence membrane properties such as the spontaneous curvature, $\bar{H}(\phi)$, and bending rigidity, $\kappa(\phi)$. More complex phenomena such as the production of in-plane interfacial forces from membrane-protein phase separation(*55*, *59*, *117*) can also be modeled. In our final proof of concept, we extend `Mem3DG` to support full mechanochemical dynamics, where proteins can mobilize in- and out-of-plane through adsorption and lateral diffusion, based on its coupling with membrane material properties and shape transformation. These scenarios highlight the relative ease of extending `Mem3DG` with additional physics and the potential utility to simulate realistic experimental scenarios. Note that for all of the examples, unless otherwise specified, the bending rigidity of membrane, κ, is assumed to be the rigidity of a bare membrane, $\kappa_b = 8.22 \times 10^{-5}$ µm · nN. Despite the superior performance of the nonlinear conjugate gradient method in finding an energy minimizing configuration, to maintain both static and dynamic interpretability, we perform all simulations using a forward Euler integrator unless otherwise noted. All simulations presented in this work were conducted on a standard workstation with Intel Xeon processors. Although the numerical algorithms are amenable to parallelization, `Mem3DG` is currently a single-threaded program. Using a single core, the simulations here complete in minutes and up to two hours.

3.1 Modeling spherical and cylindrical membranes with homogeneous physical properties

We begin our examples by using `Mem3DG` to find the equilibrium shapes of membranes with homogeneous protein density, ϕ. We ask, given an initial membrane configuration with uniform bending modulus and spontaneous curvature, what are the minimizers of the system energy? The answers are the classical equilib-

Movie 3: Animated time series simulation of the beads-on-a-string structure shown in Fig. 4C–high tension. Disruption of the intermediate metastable state, a symmetric structure with two beads shown in Appendix A, occurs. The color map indicates the local pointwise mean curvature, H, and t represents the numerical time. Available on GitHub: https://github.com/RangamaniLabUCSD/2020-Mem3DG-Applications/blob/master/examples/videos/compressed/pearl_bead_1.mp4

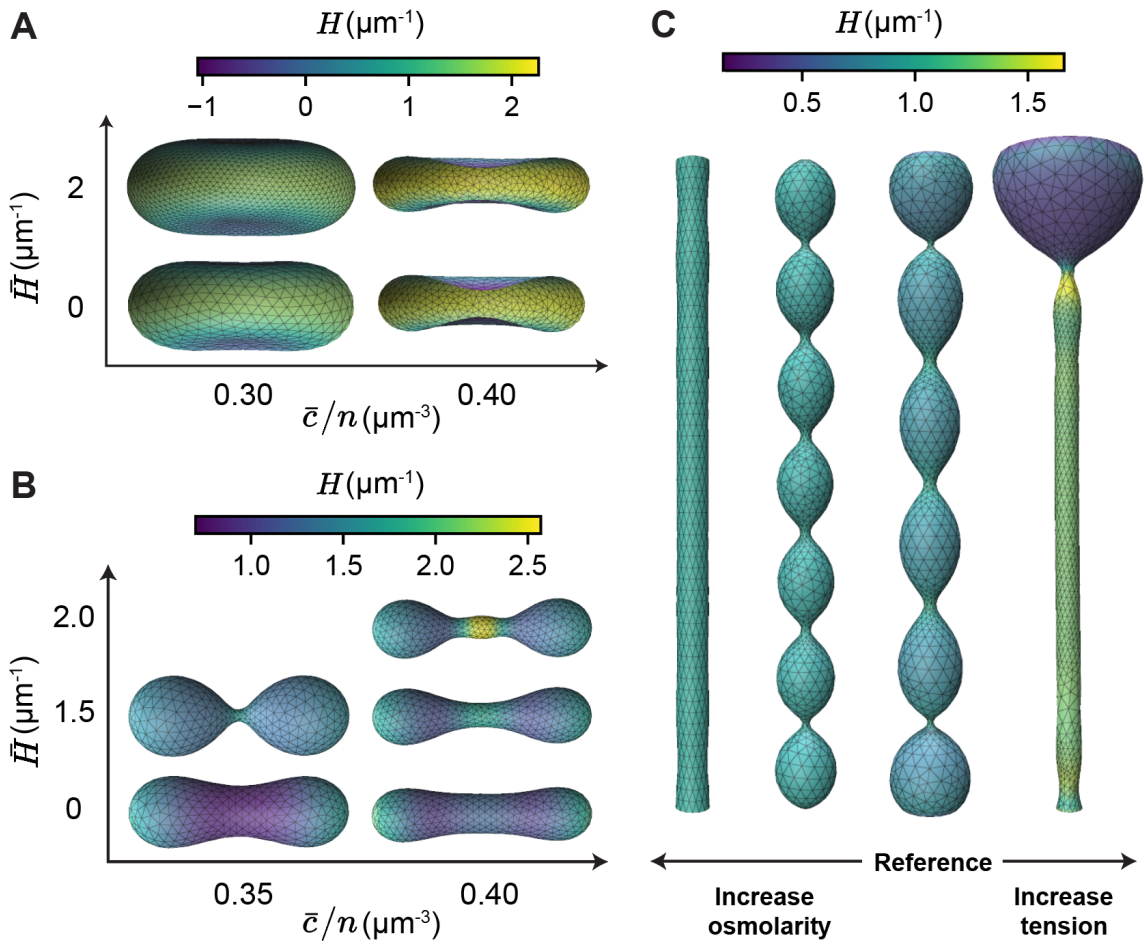


Figure 4: Recover typical equilibrium shapes of membranes with homogeneous material properties. A-B) Equilibrium solutions under different osmolarity ($\bar{c}$) and spontaneous curvature ($\bar{H}$) conditions, with initial condition of A) Oblate spheroid and B) Prolate spheroid. We vary the osmolarity by adjusting the concentration of the ambient solution, $\bar{c}$, while holding the enclosed amount of solute, n, constant. C) Equilibrium solutions of a tubular membrane structure under variations in osmolarity and surface tension.

rium solutions to the shape equation obtained analytically (*42*), and numerically using many methods with different assumptions (*39*, *118*). We will show solutions obtained using `Mem3DG` with topologies of sphere and tube (Fig. 4). These geometries were selected not only because of their potential for comparison with the legacy literature but also because they are reminiscent of biological membranous structures such as red blood cell (*119*, *120*), cell-cell tunneling and tethering (*121*–*123*), neuron beading (*124*, *125*), among other biological processes.

Starting with closed topological spheres, Fig. 4A and B shows the typical equilibrium shapes under osmotic stress while the surface area is conserved. The preferred area of the vesicle, $\bar{A} = 4\pi\,\mu\text{m}^2$, represents a sphere of radius 1 µm. This constraint is achieved by prescribing a large stretching modulus, K_A, such that the areal strain, $(A - \bar{A})/A$, is less than one percent. The strength constant of osmotic pressure, K_V is set to be 0.1 µm·nN. Initializing the simulations from an oblate spheroid, as the osmolarity increases (e.g., the normalized ambient solution, $\bar{c}/n$), we recover the well-known biconcave red blood cell shape (*101*, *118*) (Fig. 4A). The vesicle adopts a more convex configuration as we increase the spontaneous curvature, indicating an overall increase in its mean curvature with the concomitant decrease of areas with negative mean curvature (the dimple regions). In contrast, starting from a prolate spheroid, as the spontaneous curvature increases, the vesicle adopts a dumbbell configuration as the energetically preferred state (Fig. 4B). The size of the beads on the dumbbell is governed by the osmolarity, $\bar{c}/n$. These trends with respect to the variations of the spontaneous curvature and osmolarity are consistent with the analytical and numerical results in the broader literature (*42*, *87*). Qualitatively the predicted geometries of closed vesicles with homogeneous spontaneous curvature match the predictions of a detailed benchmark of mesh-based methods performed by Bian *et al.* (*87*).

We also modeled the shapes of membranes starting from an open cylinder configuration under different osmotic and surface tension conditions (Fig. 4C). This problem is related to a well-studied phenomenon called the Plateau-Rayleigh instability (*126*, *127*). The Plateau-Rayleigh instability describes how surface tension breaks up a falling stream of fluid into liquid droplets. Compared with a liquid stream, a lipid membrane provides additional resistance against instability due to its rigidity. Zhong-can and Helfrich (*128*) obtain stability regimes as a function of membrane bending rigidity and spontaneous curvature using the spectral stability analysis (*128*). Though osmotic pressure is often reported as an important cause of morphological instability (*125*, *129*, *130*), the effect of osmotic pressure is difficult to isolate in wet experiments because the change to osmolarity affects the surface tension, which is a key driver of the instability. In our simulations, the two effects are decoupled, making the investigation of individual contributions to the morphology possible. All shapes in Fig. 4C evolve from the initial tubular mesh with radius of 1 µm and axial length of 19.9 µm, under a constant spontaneous curvature of 1 μm^{-1}. These simulations are set up as local models (c.f., Section 2.5.1) where the explicit mesh is assumed to be coupled to a membrane reservoir. Additional geometric information defining the membrane reservoir and boundary conditions are required to initialize the local model. The tubular structure is considered to be a cylinder that connects two otherwise detached domains (e.g., membrane reservoirs), which combined have a total reservoir volume, $V_r = 4.19\,\mu\text{m}^3$. The strength of osmotic pressure, K_V, is set to be 0.01 µm·nN. To isolate the effect of osmotic pressure and surface tension on the morphology, we prescribe a specific surface tension that we assume to be invariant with respect to changes to the surface area. On the two boundary loops of the mesh, we apply roller boundary conditions, which restrict the movement of boundary vertices in the axial direction. The length of the tube is thus constrained to be 19.9 µm while the radius of the tube including the boundaries is free to shrink or expand.

As the osmolarity increases from the reference condition ($\bar{c}/n = 0.022\,\mu\text{m}^{-3}$) (Movie 1), we obtain solutions such as the unduloid (or pearled) structure at $\bar{c}/n = 0.030\,\mu\text{m}^{-3}$ (Movie 2), and tube at $\bar{c}/n = 0.051\,\mu\text{m}^{-3}$, which follow the trends from both analytical (*42*) and experimental observations (*19*, *125*, *129*). As we increase the surface tension from the reference condition ($\lambda = 1 \times 10^{-7}\,\text{nN}\cdot\mu\text{m}^{-1}$) to a tension-dominated regime ($\lambda = 1 \times 10^{-4}\,\text{nN}\cdot\mu\text{m}^{-1}$), we obtain the beads-on-a-string structure which minimizes the surface-to-volume ratio (Movie 3). The formation of beads-on-a-string is an interesting configuration that has been identified in biological membranes and other systems (*124*, *125*). Note that our simulations revealed a symmetric metastable state where two large beads forms at either end (Appendix A), connected by a thin tube, prior to adopting the asymmetric conformation shown in Fig. 4C. We believe that discretization artifacts such as mesh mutations act as a perturbation to break the symmetry of the metastable intermediate and transition

Movie 4: Animated time series simulation using the reference parameters, shown by Fig. 5A-C. The color map indicates the local point-wise spontaneous curvature and t represents the numerical time. Available on GitHub: https://github.com/RangamaniLabUCSD/2020-Mem3DG-Applications/blob/master/examples/videos/compressed/patch_control_1.mp4

Movie 5: Animated time series simulation of the bending-rigidity-driven budding, shown by Fig. 5D-F. The color map indicates the projection of the bending force onto the surface normal and t represents the numerical time. Available on GitHub: https://github.com/RangamaniLabUCSD/2020-Mem3DG-Applications/blob/master/examples/videos/compressed/patch_Kb_1.mp4

the membrane to a single bead configuration.

These examples with uniform spontaneous curvature profile prove the ability of Mem3DG to reproduce the expected classical solutions for spherical and tubular membrane geometries. Note that no axisymmetric constraint is imposed in these simulations. Mem3DG solves the system in full three dimensions and the symmetrical configurations are due to the problem physics. The ability to adapt to changing and complex curvatures of the membrane using discrete mesh is achieved using mesh mutation and other manipulations within solver steps. For example, the pinched neck regions of the tubes are automatically decimated with finer triangles than other regions of the mesh. For a global closed membrane simulation such as in Fig. 4A, B, we do not remove any rigid body motions from the system; Since the forces from DDG are exact and we used the forward Euler integrator, no artificial rigid body motions are introduced throughout the simulation. These examples show that that the derivation of the discrete energy and forces along with the software implementation are accurate and proceed to test Mem3DG with more complex examples.

3.2 Modeling endocytic budding mechanisms

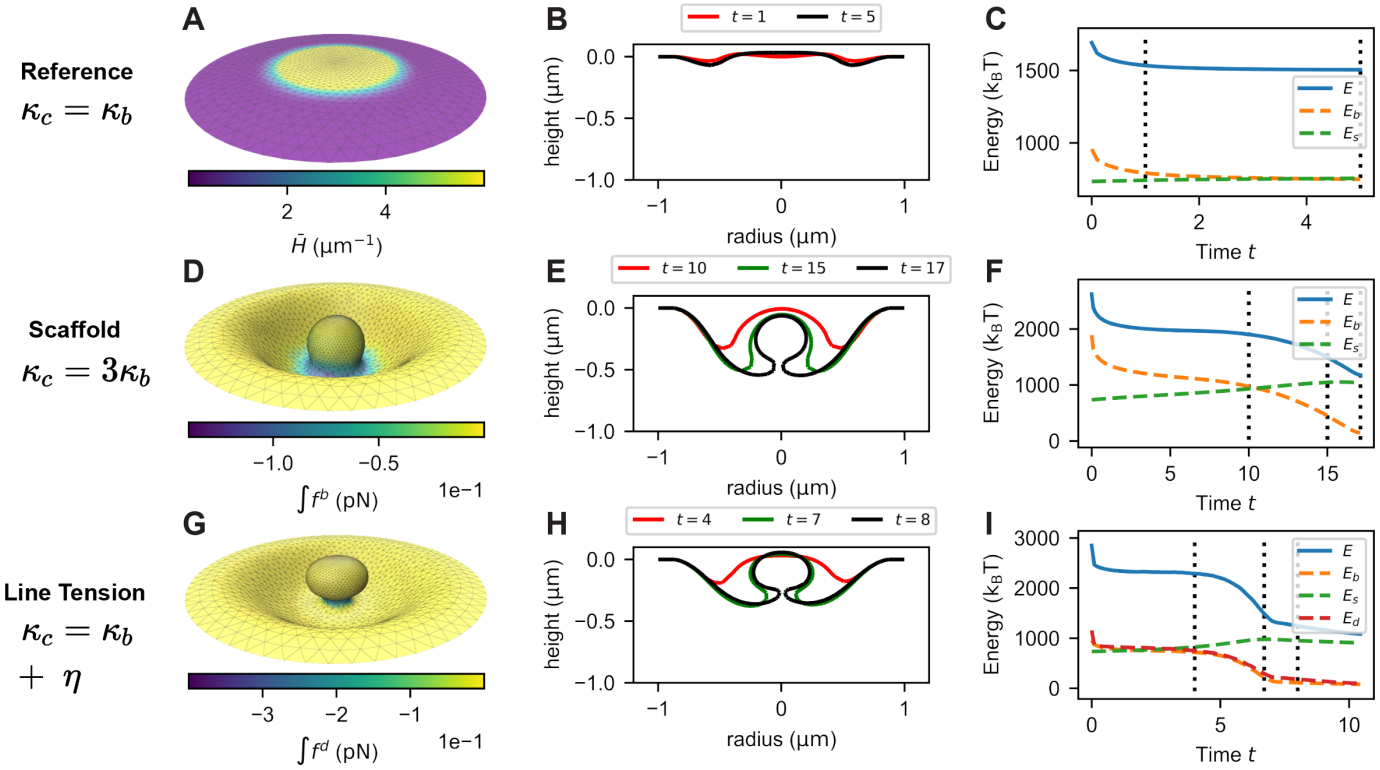


Figure 5: Budding dynamics by robust mechanisms of protein scaffolding and interfacial line tension constriction. A-C) Control group, D-F) Bending-driven scaffolding mechanism, and G-I) Interfacial line tension assisted budding. A) Spontaneous curvature distribution, $\bar{H}$, on initially flat patch. D) Normal projection of the bending force at $T = 15$. G) Normal projection of the line tension force at $T = 7$. B, E, H) Shape evolution through time-series snapshots of the Y-Z cross-sections of the membrane, corresponding to the vertical dash lines in C, F, I) Trajectory plots of system energy and its competing components.

Our goal is to highlight the potential of Mem3DG and its associated framework for investigating mechanical

Movie 6: Animated time series simulation of budding driven by interfacial line tension, shown by Fig. 5G-I. The color map indicates the signed projection of the line tension force onto the surface normal and t represents the numerical time. Available on GitHub: https://github.com/RangamaniLabUCSD/2020-Mem3DG-Applications/blob/master/examples/videos/compressed/patch_eta_1.mp4

phenomena relevant to cellular biology. Endocytosis is a cellular process in which cells uptake cargo from the extracellular environment; the transported material is engulfed by the cell membrane which then buds off to form a vesicle (*13*). Endocytosis occurs through various mechanisms including the clathrin-mediated endocytosis (*13*, *131*). It has been shown that clathrin aggregates on the plasma membrane helping to deform the membrane and form a spherical bud (*9*, *13*, *59*). However, the specific mechanisms of how membrane-clathrin interactions facilitate membrane curvature generation remain unresolved. While there is significant literature investigating the many proposed mechanisms, here we develop models to demonstrate the bud formation via spatially localized spontaneous curvature, combined with a line tension term arising from phase separations on the membrane(*132*).

We model endocytic budding on a circular patch with radius 1 µm (a disc with one boundary loop). We assume that the patch is a *local* system which is coupled to a large vesicle (Section 2.5.1). A heterogeneous protein density, $\phi \in [0, 1]$, is applied to mimic the distribution of clathrin and other scaffolding proteins. Shown in Fig. 5A, the protein density is high ($\phi = 1$) towards the center of the a geodesic disk with radius 0.5 µm) and decreases towards the boundaries ($\phi = 0$). During simulation, the geodesic distance to the center of the patch is periodically computed using the heat method (*133*). Vertexwise ϕ is assigned based on the stair-step profile smoothed by the hyperbolic tangent function applied to the geodesic distance. Each experiment is initialized as a flat patch and the displacement of boundary vertices is restricted using a fixed boundary condition. Since the patch is viewed as a small piece within a larger closed vesicle reservoir, we assume that the surface tension is constant.

A common model to account for the preferential bending owing to protein-membrane interactions is through the spontaneous curvature; we assume $\bar{H}(\phi) = \bar{H}_c\,\phi$, where $\bar{H}_c = 6\,\mu\text{m}^{-1}$ is the spontaneous curvature imposed by the membrane protein coat. Proteins such as clathrin are known to form stiff scaffolds on the membrane. Similar to the spontaneous curvature, we can assume a linear relationship between bending rigidity and protein density, $\kappa(\phi) = \kappa_b + \kappa_c\,\phi$, where constant κ_b is the rigidity of the bare membrane, and κ_c is additional rigidity of the protein scaffold.

Shown in Fig. 5A-C and Movie 4, is the control simulation where we set the contribution to the rigidity from protein to be the same as that of the raw membrane, $\kappa_c = \kappa_b$. Fig. 5A shows the initial flat configuration of the control experiment; the color bar shows the heterogeneous spontaneous curvature resulting from the prescribed protein density profile. In the control experiment, the bending force is resisted by the surface tension (Fig. 5C) until, at the final frame in Fig. 5B ($t = 5$), the membrane reaches the equilibrium configuration where the surface tension cancels with the bending force. In a second model, we assume that the scaffolding proteins are much more rigid than the bare membrane, $\kappa_c = 3\kappa_b$. Fig. 5D-F and Movie 5 show the bud formation due to this increased protein scaffolding effect. The greater rigidity results in an increase of initial bending energy, which outcompetes the resistance from the surface tension (Fig. 5F). Fig. 5E shows the shape evolution from a flat patch to a successful bud with a pinched neck. Fig. 5D shows the signed projection of the bending force onto the vertex normal, $\int f_i^b = \int \vec{f}_i^b \cdot \vec{n}_i$, at $T = 15$.[5] We can see an "effective line tension" driven by the heterogeneous spontaneous curvature which constricts the neck. This phenomenon is theoretically explored in detail by Alimohamadi *et al.* (*58*).

For our third model, based on the prior observations that protein phase separation on surfaces can lead to a line tension (*132*), we incorporate a Ginzburg-Landau interfacial energy into the system,

$$E_d = \frac{1}{2}\sum_{f_{ijk}} \eta \int \|\nabla_{\vec{\theta}}\phi\|_{ijk}^2 \rightarrow \frac{1}{2}\int_{\mathcal{M}} \eta \|\nabla_{\vec{\theta}}\phi\|^2 \, dA. \tag{28}$$

where η, referred to as the Dirichlet energy constant, governs the strength of the energy penalty, and $\nabla_{\vec{\theta}}\phi$ is the discrete surface gradient of the protein density profile. The term is similar to previous modelling efforts

[5] Outward-pointing angle-weighted normal; the same applies to the interfacial line tension.

Movie 7: Animated time series simulation of protein aggregation on a dendritic spine (Fig. 6B). The color map shows the order parameter for protein density, $\phi \in (0, 1)$, and t represents the numerical time. Available on GitHub: https://github.com/RangamaniLabUCSD/2020-Mem3DG-Applications/blob/master/examples/videos/compressed/spine_1.mp4

by Elliott and Stinner (78) and Ma and Klug (77) using FEM; because we use the protein phase separation as a prior, we exclude the double-well term which models the thermodynamics of phase separation, and incorporate only the Dirichlet energy component that penalizes the heterogeneity of membrane composition.

Defined as the slope of the linearly interpolation of ϕ on faces of the mesh, f_{ijk}, the discrete surface gradient of the protein density is,

$$\nabla_{\vec{\theta}}\, \phi_i = \frac{1}{2A_{ijk}} \sum_{\underline{e}_i \in N(f_{ijk})} \phi_i \vec{e}_i^{\,\perp}, \tag{29}$$

where following illustration in Fig. 1C, $\vec{\underline{e}}_i$ is the vector aligned with the halfedge $\underline{e}_i$, with its length of l_i, and $(\cdot)^{\perp}$ represents a 90° counterclockwise rotation in the plane of f_{ijk}. The resulting line tension force $\int \vec{f}^{d}$ is then the shape derivative of the Dirichlet energy, $\nabla_{\vec{r}} E_d$, which acts to minimize the region with sharp heterogeneity. The detailed derivation of the shape derivative is elaborated in Appendix C.3, where we follow the formulaic approach by taking geometric derivatives of basic mesh primitives shown in Eq. (C.13). Fig. 5G-I and Movie 6 show the trajectory where we used control bending rigidity, $\kappa_c = \kappa_b$, and the additional interfacial line tension, $\eta = 5 \times 10^{-4}$ µm · nN. We find that the interfacial line tension, jointly with the bending force, lowers the system energy and helps the formation of a spherical bud (Fig. 5I, H). Fig. 5G shows the snapshot ($t = 7$) with the color map representing the signed normal projection of the interfacial line tension that acts to constrict the neck. These examples of endocytic bud formation help to illustrate the utility of `Mem3DG` and the accompanying theoretical framework. Since physical parameters are assigned on a per-vertex basis, it is straightforward to incorporate heterogeneity such as the nonuniform bending rigidity and spontaneous curvature. In smooth theory and its derived discrete mesh models, when the membrane is heterogeneous, it is required to decompose the force separately in normal and tangential direction (40, 65); In contrast, the general derivation of the discrete bending force following the formalism of DDG permits modeling membrane with heterogeneous material properties without any modification to its formulation (Section 2.3.3). The introduction of Dirichlet energy and line tension force serves to highlight the relative ease to extend the modeled physics.

3.3 Protein aggregation on the realistic mesh of a dendritic spine

Table 3: Parameters used in Section 3.3

Parameters	Values
ϕ_0	0.1
κ_c	0 nN·µm
$\bar{H}_c$	10 µm^{-1}
B	3 nN^{-1}·µm^{-1}·s^{-1}
η	0.01 µm·nN

While the prior examples have focused on the mechanical response of the membrane given a bound protein distribution, we can also model the inverse problem. Given the membrane shape, how do curvature-sensing proteins diffuse in the plane of the membrane and distribute over the domain? And how does the resultant protein distribution influence the stresses of the system? To model the protein dynamics, we use three terms corresponding to protein binding, curvature sensitivity, and lateral diffusion.

To model the binding of proteins to the membrane, we assume that the energy of adsorption, ε, is constant

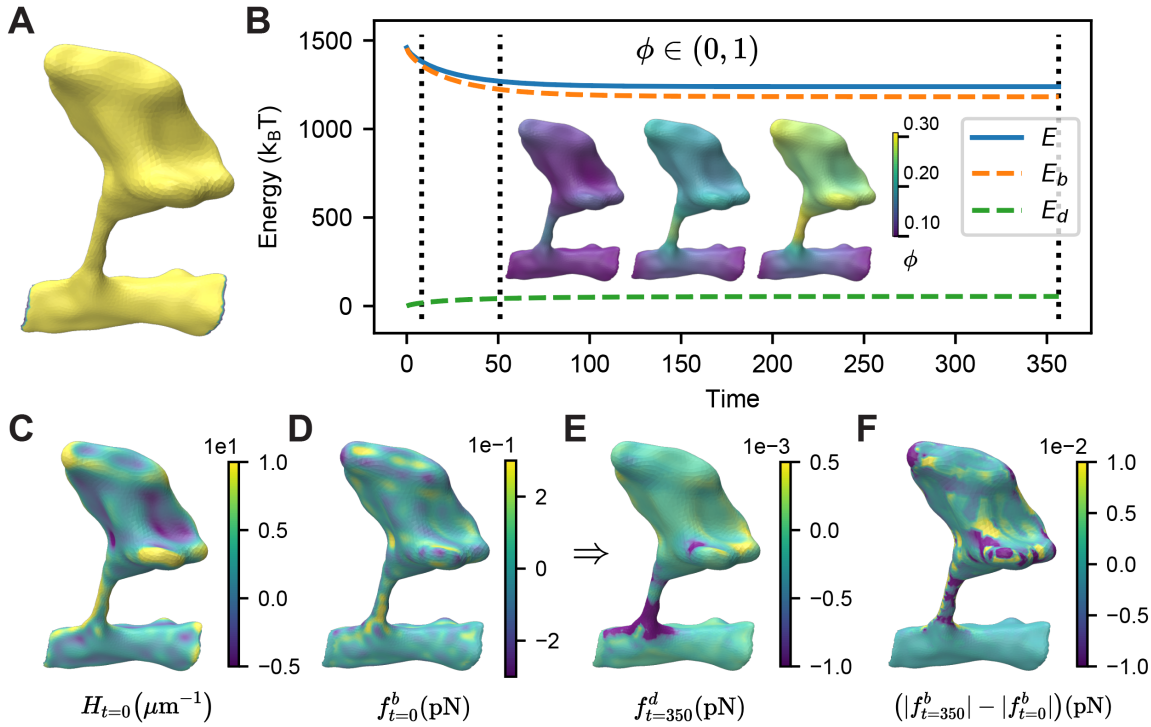


Figure 6: Protein aggregation on a realistic dendritic spine geometry. A) Mesh of the dendritic spine and its boundary elements. B) Trajectory of protein evolution and components of system energy. C) Mean curvature of the geometry. The normal component of D) the bending force at $t = 0$, E) the line tension force produced by the equilibrium protein distribution, and F) the difference in the bending force profile produced by final protein distribution as opposed to the initial distribution.

and uniform across the surface such that the discrete adsorption energy is,

$$E_a = \varepsilon \sum_i \int \phi_i, \tag{30}$$

where ϕ_i is an order parameter representing the area density of protein at each vertex. Taking the derivative with respect to ϕ, referred to as the chemical derivative,

$$\mu_i^a = -\nabla_\phi E_a = -\int \varepsilon, \tag{31}$$

we obtain the adsorption component of the chemical potential. To account for protein curvature-sensitivity, we find the chemical potential of the bending energy,

$$\mu_i^b = -\nabla_\phi E_b = \int [2\kappa_i (H_i - \bar{H}_i)\nabla_\phi \bar{H}_i - (H_i - \bar{H}_i)^2 \nabla_\phi \kappa_i], \tag{32}$$

where we assume that $\nabla_\phi \kappa_i = \kappa_c$, and $\nabla_\phi \bar{H}_i = \bar{H}_c$ where κ_c and $\bar{H}_c$ are constant parameters defined in Section 3.2. The first term of Eq. (32) is the shape mismatch penalty; considering the binding of a rigid protein which induces a significant spontaneous curvature change, if the curvature of membrane is far from this new spontaneous curvature, then the shape mismatch between the membrane and proteins will prevent binding. Alternatively, if the protein is more flexible, a shape mismatch results in a small energetic penalty. The second term of Eq. (32) endows the protein with curvature sensitive binding.

The in-plane diffusion of the protein is accounted for by the chemical derivative of the smoothing Dirichlet energy, E_d,

$$\mu_i^d = -\nabla_\phi E_d = -\int \eta \Delta_s \phi_i, \tag{33}$$

where η is the same Dirichlet energy constant introduced in Eq. (28) that governs the strength of interfacial line tension, $\int \bar{f}^d$. The total chemical potential captures the bending, adsorption and diffusion components. A

Movie 8: Animated time series simulation of the hypertonic case shown in Fig. 7B-C. The color map shows the order parameter for protein density, $\phi \in (0, 1)$, and t represents the numerical time. Available on GitHub: https://github.com/RangamaniLabUCSD/2020-Mem3DG-Applications/blob/master/examples/videos/compressed/bud_hyper_kappa_c.mp4

mobility rate constant, B, determines the time scale of the chemical response,

$$\dot{\boldsymbol{\phi}} = B\boldsymbol{\mu} = B(\boldsymbol{\mu}_b + \boldsymbol{\mu}_a + \boldsymbol{\mu}_d). \tag{34}$$

We investigate the influence of curvature-dependent binding to a realistic dendritic spine geometry which was reconstructed from electron micrographs and curated using `GAMer 2` (Fig. 6A) (*32*). The mean curvature of the spine geometry is shown Fig. 6C. We isolate the effect of curvature-dependent binding by assuming that the shape of the spine is fixed and impose Dirichlet boundary conditions at the base on the spine to fix the protein concentration, $\phi = 0.1$ (Fig. 6A).

Starting from a homogeneous protein distribution, $\phi_0 = 0.1$, Fig. 6B and Movie 7 show the evolution of the protein distribution and a trajectory of the system energy. Note that for simplicity, we have turned off the adsorption energy term since it only shifts the basal protein-membrane interactions which will also be set by the Dirichlet boundary condition. `Mem3DG` constrains the range of $\phi \in (0, 1)$ using the interior point method (*115*). Due to the curvature sensitivity of the protein, illustrated by the snapshots (Fig. 6B, $T = 350$) representing the final protein distribution, the protein aggregates towards regions of high curvature (e.g., on the spine head).

Although the proteins can reduce the bending energy by modulating the local bending modulus and spontaneous curvature, the protein distribution at equilibrium does not cancel out the bending energy. We expect that the Dirichlet energy term, which limits ϕ to be smooth across the geometry, restricts the protein from further aggregating to the extent required to cancel out the bending energy. The components of forces on the initial and final configurations of the spine are compared in Fig. 6D-F. The initial homogeneous protein distribution has no line tension forces and a bending force shown in Fig. 6D. After the protein distribution reaches the steady state, line tension appears in response to membrane heterogeneity Fig. 6E. We hypothesize that, similar to Section 3.2, the line tension constricts the neck of the spine and helps to support the cup-like structures in the spine head. While in most regions the distribution of proteins reduces the force, several regions experience increased stress Fig. 6F. Note that the magnitude of the forces generated by proteins in this model is orders of magnitude smaller than the initial bending force. Thus, this example demonstrates that `Mem3DG` can be used on meshes imported from realistic geometries of cellular substructures.

3.4 Membrane dynamics with full mechanochemical feedback

Table 4: Parameters used in Section 3.4 for models with full mechanochemical feedback

Parameters	Values
ϕ_0	0.1
κ_c	8.22×10^{-5} μm·nN
$\bar{H}_c$	10 μm^{-1}
K_V	0.5 nN·μm
K_A	1 nN·μm^{-1}
B	3 $nN^{-1}\cdot\mu m^{-1}\cdot s^{-1}$
ξ	1 nN·s·μm^{-1}
ε	-1×10^{-3} nN·μm
η	0.1 μm·nN
$\bar{V}$	2.91, 3.95 and 4.99 μm^3

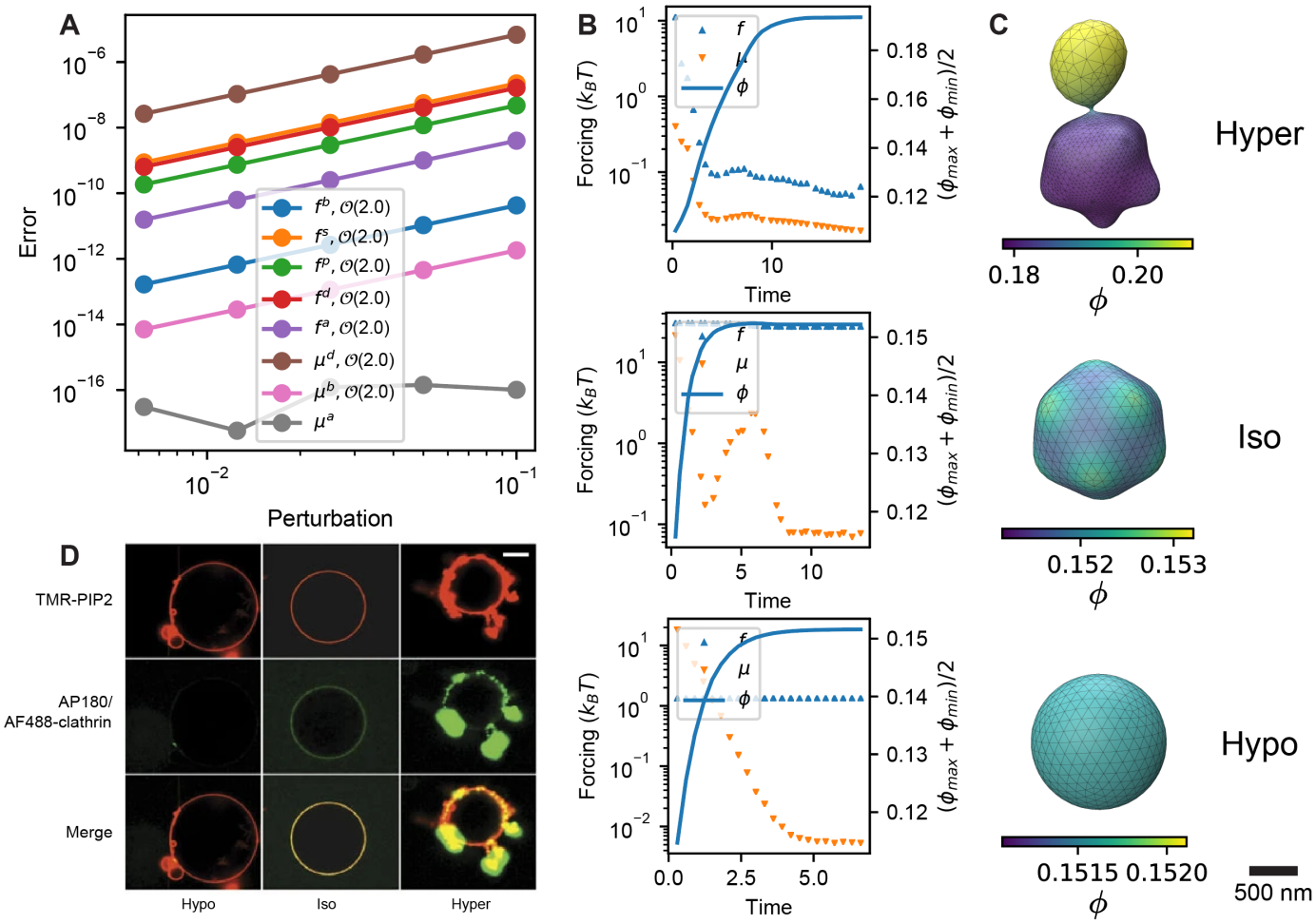


Figure 7: Modeling budding from a vesicle driven by the full mechanochemical feedback of membrane–protein interactions. A) Validation of the exactness of the discrete forcing with respect to the discrete energy. The terms correspond to forces from bending f_b, tension-area f_s, pressure-volume f_p, Dirichlet f_d, and protein binding f_a. μ_d, μ_b, and μ_a are the chemical potential of diffusion, bending, and binding respectively. B) The time trajectory of budding dynamics in hypertonic, isotonic, and hypotonic osmotic conditions. C) The final snapshot of the system configuration under hypertonic, isotonic, hypotonic conditions. D) Similar geometries to those shown in C) have been observed in experiments by Saleem *et al.* (59).

Movie 9: Animated time series simulation of the isotonic case shown in Fig. 7B-C. The color map shows the order parameter for protein density, $\phi \in (0, 1)$, and t represents the numerical time. Available on GitHub: https://github.com/RangamaniLabUCSD/2020-Mem3DG-Applications/blob/master/examples/videos/compressed/bud_iso_kappa_c.mp4

In this section, we will demonstrate the use of Mem3DG to model the complete mechanochemical feedback of a protein-membrane system. For the following simulations, not only can proteins bind in a curvature-dependent manner, but the membrane can also deform leading to a feedback loop. We have introduced all of the force terms in previous sections except the shape derivative of the adsorption energy,

$$\int \vec{f}_i^a = -\nabla_{\vec{r}} E_a = -\sum_{\underline{e}_{ij} \in N(v_i)} \varepsilon \left(\frac{\phi_i}{3} + \frac{2\phi_j}{3} \right) \int 2\vec{H}_i, \tag{35}$$

which accounts for the change in the area of protein coverage (i.e., expanded coverage if $\varepsilon < 0$).

Revisiting the claim that all discrete forcing is exact with respect to the discrete energy, we validate by examining the convergence of the forcing terms with respect to the size of perturbation to the system configuration, ϵ (Fig. 7A). This is based on the leading order expansion done in Eq. (24), which concludes that the forcing terms are exact if their rate of convergence is at least 2nd order. Shown in Fig. 7A, this is true for all forcing terms; note that since the adsorption energy, E_a, is a linear function with respect to ϕ, μ^a can be determined to the machine precision for all perturbation sizes. A meaningful discrete-smooth comparison of all terms in the energy and forcing similar to Section 2.5.4 requires the analytical solutions of the bending force and interfacial line tension arising from the spatially heterogeneous membrane properties, which to the best of our knowledge, are not available. In Section 3.2, we introduced a heterogeneous membrane composition as a static property. By prescribing the protein density profile, we can get hints about how to form membrane buds from a patch. Here we lift this assumption and simulate the dynamics of osmotic pressure-driven budding from a vesicle. The dynamics couples the protein-membrane mechanochemical feedback and includes protein binding and diffusion introduced in Section 3.3. The expressions of discrete free energy and forcings do not change but we allow the membrane configuration and protein density to interact and evolve simultaneously.

We start each simulation from a sphere with a uniform protein concentration, $\varphi = \varphi_0 = 0.1$. We consider the evolution of the system in different osmotic conditions: hyper-, iso-, and hypotonic, $\bar{V} = 2.91, 3.95$ and 4.99 µm^3, respectively. Additional parameters for these simulations are given in Table 4. Fig. 7B shows plots of the mechanical, $\|\vec{\boldsymbol{f}}\|_{L_2}^2$, and chemical response, $\|\boldsymbol{\mu}\|_{L_2}^2$, along with the protein density, $(\phi_{\text{max}} + \phi_{\text{min}})/2$, over the trajectory for each osmotic condition. Note that under hypo- and isotonic conditions, the system reaches the (quasi) steady-state where both the shape and protein distribution stabilize, while in a hypertonic solution, the system continues to experience strong mechanical force and protein mobility, which we expect to drive further morphological changes of the membrane beyond our simulation stopping point. Fig. 7C shows the final snapshot of each simulation across the osmotic conditions with the protein density represented by the color map. In hypertonic conditions, the osmotic pressure provides sufficient perturbations to membrane morphology, which initializes the positive feedback loop between membrane curvature generation and protein aggregation; This mechanochemical feedback jointly promotes the outward bending of the membrane and results in bud formation (Fig. 7C–hyper and Movie 8). Under isotonic and hypotonic conditions, the osmolarity does not permit the large change in the volume required to form spherical buds with a thin neck. In hypotonic condition, the pressure-tension balance provides substantial stability to the initial spherical configuration. In comparison, in the isotonic condition, the comparable competition between the chemical and mechanical response leads to a patterned protein distribution and an undulating morphology (Fig. 7C–hypo and Movie 9). This example illustrates the possibility to utilize Mem3DG to model a full mechanochemical feedback between membrane and protein. Although we do not intend to replicate the exact experimental conditions and assumptions, the geometries obtained from these simulations resemble the shapes obtained by Saleem *et al.* (*59*) who investigated budding from spherical vesicles under differing osmotic conditions (Fig. 7D) (*59*).

4 Summary

In this work, we introduce a new perspective for constructing a 3D membrane mechanics model on discrete meshes. The goal of our approach is to close the gap between existing discrete mesh based models (*60*, *84*–*94*, *97*, *98*) and the smooth theory. Specifically, we seek to advance the discussion behind the choice of

algorithmic approaches for computing geometric values required for the discrete energy and force (*65*, *87*, *95*). We start by writing a discrete energy, Eq. (3), mirroring the spontaneous curvature model. Then using the perspective of DDG, we show that there is a formulaic approach for deriving the corresponding discrete force terms based on fundamental geometric vectors. By identifying geometric invariants and grouping terms, the resulting discrete forces have exact correspondence to the traditional smooth theory. This helps us to facilitate the comparison between smooth and discrete contexts enabling new geometric perspectives and understanding of numerical accuracy. Moreover, the discrete force terms are functions of readily accessible geometric primitives, and the formulation is amenable for efficient implementation and extension.

We have produced a reference software implementation called `Mem3DG`. Using `Mem3DG`, we validate our theory by reproducing the solutions to the classical shape transformations of a spherical and tubular vesicle. We further demonstrate the derivation and incorporation of additional physics terms to model protein-membrane interactions. Following our formulaic approach using DDG, we derived the discrete analog of various physics such as the interfacial line tension, surface-bulk adsorption, protein lateral diffusion, and curvature-dependent protein aggregation. To exemplify all the introduced physics, the full mechanochemical coupling between the membrane shape and protein density evolution results in protein localization, pattern formation, and budding. These examples serve to highlight the extensibility of the framework, which does not require the introduction of coordinates and advanced tensor calculus commonly needed to solve problems on arbitrary manifolds. The software implementation `Mem3DG` was designed to facilitate coordination between theoretical modeling and wet experiments; we hope to support the modeling of scenes with well-resolved protein-membrane interactions such as in the electron tomograms (*134*). We expect that as the advances in biophysical modeling and membrane ultrastructure imaging progresses, `Mem3DG` will emerge as a useful tool to test new hypotheses and understand cellular mechanobiology.

5 Author contribution

P.R. conceived the research. P.R., C.Z, and C.T.L. designed the research. C.Z developed the mathematical models and analyzed the results. C.Z and C.T.L. developed the software. C.Z, C.T.L., and P.R. wrote and edited the article. All authors read and approved the final article.

6 Declaration of Interest

The authors declare no competing interests.

7 Acknowledgments

The authors would like to acknowledge Dr. Ali Behzadan, Prof. Ravi Ramamoorthi, and Prof. Albert Chern for helpful discussions and critical feedback. This work was supported in part by the National Institutes of Health R01GM132106, National Science Foundation DMS 1934411, and an Air Force Office of Scientific Research DURIP FA9550-19-1-0181 to PR. CTL also acknowledges support from a Hartwell Foundation Postdoctoral Fellowship.

Appendices

A Supplemental figures

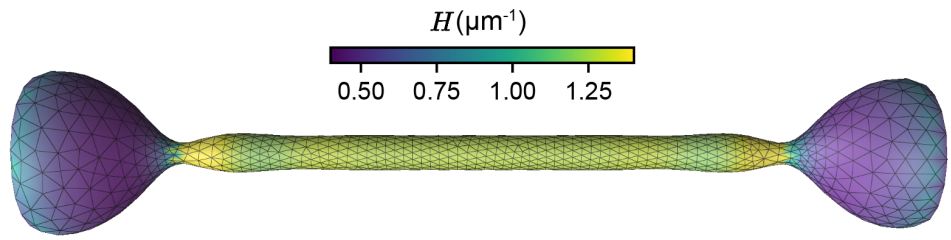


Figure A.1: A symmetric metastable state with two beads instead of a single larger bead is observed, prior to collapsing to the solution shown in Fig. 4C – high tension

Smooth ⟷ **Discrete**

$$\mathrm{vol}(\mathcal{M}) = \frac{1}{3}\int_{\mathcal{M}} \vec{n}\cdot\vec{r}\, dA \xrightarrow{\nabla_{\vec{r}}} \vec{n}\, dA \qquad \sum_{f_{ijk}} V_{0ijk} \xrightarrow{\nabla_{\vec{r}}} \int \vec{n}_i = \frac{1}{3}\sum_{f_{ijk}\in N(v_i)} \int \vec{n}_{(ijk)} = \frac{1}{3}\sum_{f_{ijk}\in N(v_i)} A_{ijk}\vec{n}_{(ijk)}$$

$$\mathrm{Area}(\mathcal{M}) = \int_{\mathcal{M}} dA \xrightarrow{\nabla_{\vec{r}}} 2H\vec{n}\, dA \qquad \sum_{f_{ijk}} A_{ijk} \xrightarrow{\nabla_{\vec{r}}} \int 2\vec{H}_i = \sum_{\underline{e}_{ij}\in N(v_i)} \int 2\vec{H}_{ij} = \frac{1}{2}\sum_{\underline{e}_{ij}\in N(v_i)} (\cot\angle_{ikj} + \cot\angle_{ilj})\vec{\underline{e}}_{ji}$$

$$\mathrm{Mean}(\mathcal{M}) = \int_{\mathcal{M}} H\, dA \xrightarrow[\text{Assume closed}]{\nabla_{\vec{r}}} K\vec{n}\, dA \qquad \sum_{e_{ij}} \int H_{ij} = \frac{1}{2}\sum_{e_{ij}} l_{ij}\varphi_{ij} \xrightarrow[\text{Assume closed}]{\nabla_{\vec{r}}} \int \vec{K}_i = \sum_{\underline{e}_{ij}\in N(v_i)} \int \vec{K}_{ij} = \frac{1}{2}\sum_{\underline{e}_{ij}\in N(v_i)} \frac{\varphi_{ij}}{l_{ij}}\vec{\underline{e}}_{ji}$$

$$\mathrm{Gauss}(\mathcal{M}) = \int_{\mathcal{M}} K\, dA \xrightarrow[\text{Gauss-Bonnet}]{\nabla_{\vec{r}}} 0 \qquad \sum_{v_i} \int K_i = \sum_{v_i}\left(2\pi - \sum_{f_{ijk}\in v_i} \angle_{jik}\right) \xrightarrow[\text{Gauss-Bonnet}]{\nabla_{\vec{r}}} 0$$

Figure A.2: Steiner's formula in continuous and discrete geometry: chain of smooth and discrete shape derivatives of integrated geometric measurements (*135*).

B Rationale for integrated measurements in discrete contexts

The rationale for why an integrated measurement in discrete contexts is the natural counterpart to pointwise measurements in smooth contexts can be demonstrated by considering the curvature of a discrete polygonal curve. If we attempt to define the curvature, C, of the discrete polygonal curve in a naïve pointwise manner, following the procedure in smooth settings, we will find zero curvature along edges and infinite curvature

(owing to the discontinuity) on vertices. Thus the traditional view of curvature from smooth manifolds reveals no useful information about the geometry of the discrete curve. We must find another geometric relationship that can translate between smooth and discrete contexts to maintain the geometric connection.

One relationship from smooth differential geometry is the equivalence of the integrated curvature and the turning angle ψ (i.e., the total angle by which the tangent vector of the curve turns over some domain l). Returning to the discrete context, we can seek to preserve this relationship between the integrated curvature and turning angle by finding a compatible definition. Since the discrete turning angle, ψ_i, between two connected edges of the discrete polygonal curve is well-defined, we can set the "discrete" curvature, $\int C$, of a vertex, v_i, to be

$$\left(\int C\right)_i \equiv \psi_i. \tag{B.1}$$

We note that the notation for the discrete curvature, $\left(\int C\right)_i$ is used only in this illustrative example; in the remainder of the text, we will omit the parenthesis and use the simplified notation, $\int C_i$. To make sense of the integral over a discrete object, additional care must be taken to represent the curvature from a distributional sense (*99*). This is related to traditional approximation methods, such as the point allocation method, which bridges a smooth and discrete problem by convoluting the smooth problem with impulse functions (e.g., the Dirac delta function) at a finite number of locations (*116*).

As we have shown, integrated geometric measurements enable us to preserve geometric relationships (from smooth contexts) for discrete objects, and are thus preferred over pointwise definitions. Nevertheless, we often require a pointwise discrete measurement for use in algorithms and visualization. An integrated measurement can be converted to a meaningful pointwise discrete measurement by normalizing the value over a domain. For the discrete polygonal curve, the domain can be the dual vertex length, l_i (i.e., the discrete analog of l). l_i is given by half of the sum lengths of the two incident edges. A pointwise curvature on the vertex v_i is then given by,

$$C_i = \int C_i/l_i = \psi_i/l_i. \tag{B.2}$$

Another rationale for using an integrated value for a discrete geometric measurement is that we can arrive at the same definition from multiple perspectives. Returning to the definition of the curvature of a polygonal curve, without introducing the turning angle, we can arrive at the same result by adopting the Steiner view (*99*, *136*)[6]. In the Steiner view, we replace the sharp vertices with a smooth circular arc with radius ϵ such that the discrete geometry is made smooth such that the curvature is well-defined everywhere. As the only curved section, every circular arc has a discrete (integrated) curvature,

$$\int C = \int_{\text{arc}} C\, ds = C_{\text{arc}} l_{\text{arc}} = \frac{1}{\epsilon}(\epsilon\psi) = \psi, \tag{B.3}$$

where $C_{\text{arc}} = 1/\epsilon$ is the curvature of the circular arc, and $l_{\text{arc}} = \epsilon\psi$ is the arc length. We see that in the Steiner view, the integrated curvature is still equivalent to the turning angle. Following similar logic, other discrete definitions are described in Section 2.2 and the DDG literature (*99*, *100*).

C Discrete shape and chemical derivatives of discrete energy

C.1 Halfedge on a triangulated mesh

A scalar quantity on an edge is symmetric with respect to index permutation; For example, the scalar mean curvature (Eq. (8)),

$$\int H_{ij} = \int H_{ji} = \frac{l_{ij}\varphi_{ij}}{2}. \tag{C.1}$$

However, as we will show in detail in the following sections, this symmetry does not apply to vector quantities, which compose the discrete shape derivative of the energy, force. For example, the corresponding mean

[6] We will use the Steiner view to define the discrete curvature of a surface in Section 2.2

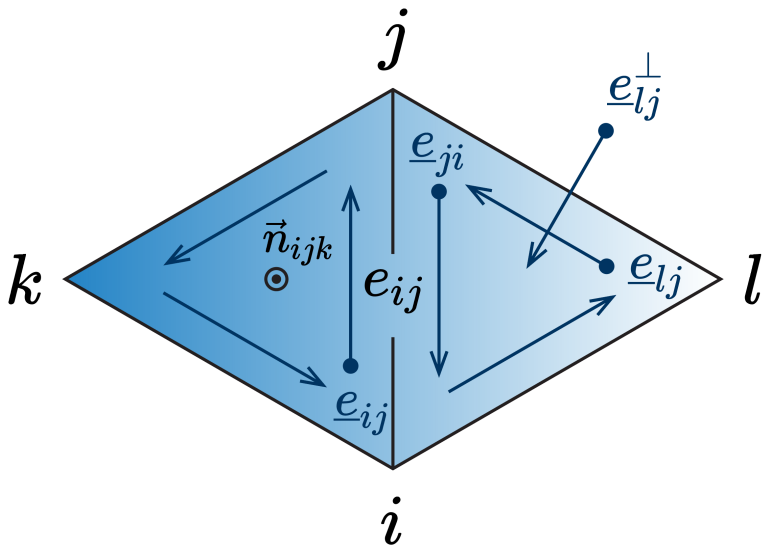


Figure C.1: Schematics for halfedges on a triangulated mesh.

curvature vector,

$$\int \vec{H}_{ij} \neq \int \vec{H}_{ji}. \tag{C.2}$$

To highlight the directionality of vector quantities and disambiguate the notation, here we review the concept of a halfedge on a triangulated mesh. Given any non-boundary edge, e_{ij}, on a manifold mesh, there exits two associated halfedges, $\underline{e}_{ij}$ and $\underline{e}_{ji}$ (Fig. C.1). This convention leads to an oriented (counterclockwise) halfedge loop on each triangle face and subsequently a well-defined 1) 90° counterclockwise rotation of the halfedge in the plane of the face (e.g., $\underline{e}_{lj} \rightarrow \underline{e}_{lj}^{\perp}$), and 2) face normal (outward) based on the right hand rule Fig. C.1. Beside being used to differentiate vector/scalar quantities, the concept of halfedge is widely adopted data structure for managing connected graphs, or meshes, for which we refer the reader to the broader literature (*110*, *137*).

C.2 Deriving the bending force as the shape derivative of bending energy

The geometric derivatives of mesh primitives, including edge length, l, dihedral angle, φ, vertex dual area, A, are given as

$$\nabla_{\vec{r}_i} l_{ij} = \frac{\vec{\underline{e}}_{ji}}{l_{ij}}, \tag{C.3a}$$

$$\nabla_{\vec{r}_i} \varphi_{ij} = \frac{1}{l_{ij}}(\cot \angle_{ijk} \vec{n}_{ijk} + \cot \angle_{ijl} \vec{n}_{ilj}), \tag{C.3b}$$

$$\nabla_{\vec{r}_i} \varphi_{jk} = -\frac{1}{l_{jk}}(\cot \angle_{ijk} + \cot \angle_{ikj}) \vec{n}_{ijk} = -\frac{l_{jk}}{2A_{ijk}} \vec{n}_{ijk}, \tag{C.3c}$$

$$\nabla_{\vec{r}_i} A_i = \frac{1}{3} \sum_{f_{ijk} \in N(v_i)} \nabla_{\vec{r}_i} A_{ijk} = \frac{1}{6} \sum_{\underline{e}_{ij} \in N(v_i)} (\cot \angle_{ikj} + \cot \angle_{ilj}) \vec{\underline{e}}_{ji}, \tag{C.3d}$$

$$\nabla_{\vec{r}_i} A_j = \frac{1}{3} \sum_{f_{ijk} \in N(e_{ij})} \nabla_{\vec{r}_i} A_{ijk} = \frac{1}{6}(\vec{\underline{e}}_{jk}^{\perp} + \vec{\underline{e}}_{lj}^{\perp}), \tag{C.3e}$$

where $\vec{n}_{ijk}$ is the unit normal vector of the face f_{ijk}, $\vec{\underline{e}}_{ji}$ is the vector aligned with the halfedge, $\underline{e}_{ji}$, with its length of l_{ij} (*138*). The indices and nomenclature in Eq. (C.3b), Eq. (C.3c) and Eq. (C.3e) are illustrated in the diamond neighborhood (Fig. C.1) and those of Eq. (C.3d) are illustrated in the fan neighborhood (Fig. 1A).

To simplify the expression and provide more structure for the subsequent discrete variation, it is convenient to define some fundamental curvature vectors,

$$\int 2\vec{H}_{ij} = \frac{1}{2}(\nabla_{\vec{r}_i} A_{ijk} + \nabla_{\vec{r}_i} A_{ijl}) = \frac{1}{4}(\vec{\underline{e}}_{jk}^{\perp} + \vec{\underline{e}}_{lj}^{\perp}) \tag{C.4a}$$

$$\int \vec{K}_{ij} = \frac{1}{2}\varphi_{ij}\nabla_{\vec{r}_i} l_{ij} \tag{C.4b}$$

$$\int \vec{S}_{ij,1} = \frac{1}{2} l_{ij}\nabla_{\vec{r}_i}\varphi_{ij} = \frac{1}{2}(\cot\angle_{ijk}\vec{n}_{ijk} + \cot\angle_{ijl}\vec{n}_{ilj}) \tag{C.4c}$$

$$\int \vec{S}_{ij,2} = \frac{1}{2}\left(l_{jk}\nabla_{\vec{r}_i}\varphi_{jk} + l_{jl}\nabla_{\vec{r}_i}\varphi_{jl} + l_{ji}\nabla_{\vec{r}_i}\varphi_{ji}\right) = -\frac{1}{2}(\cot\angle_{jki}\vec{n}_{ijk} + \cot\angle_{ilj}\vec{n}_{ilj}), \tag{C.4d}$$

where the mean curvature vector, $\int \vec{H}$, results from area gradient; Gaussian curvature vector, $\int \vec{K}$, and the Schlafli vector, $\int \vec{S}$, consists of the two components of the variation of total mean curvature, $\frac{1}{2}\sum_{e_{ij}} l_{ij}\varphi_{ij}$. The asymmetry of vector quantities in Eq. (C.4) under index permutation (Eq. (C.2)) arises from the vertex we take the shape derivative with respect to (i.e., v_i, or v_j); because of the asymmetry, we can associate each Schlafli vector with a unique halfedge. Similar to the translation from edge values to vertex value (Eq. (9)), we can also translate the halfedge value to vertex value by summing all halfedge values over the fan neighborhood,

$$\int (\cdot)_i = \sum_{\underline{e}_{ij}\in N(v_i)} \int (\cdot)_{ij}. \tag{C.5}$$

Note that unlike translating edge values, there is no prefactor $1/2$ for translating halfedge values because each halfedge is uniquely associated with one vertex. The translated curvature vectors on a vertex cane compared against vertexwise smooth analytical solutions as benchmarked in Section 2.5.4. Now we have all of the elements needed to derive the derivatives of the discrete Willmore bending energy. Because the discrete energy is locally supported by the vertex, v_i, and its 1-ring neighbors, $v_j \in N(v_i)$, we can separate them into the "diagonal" term, and "off-diagonal" term,

$$\begin{aligned}\int \vec{f}_i^b = -\nabla_{\vec{r}_i} E_b &= -\nabla_{\vec{r}_i}\left(\sum_i \kappa_i (H_i(\vec{r}) - \bar{H}_i)^2 A_i(\vec{r})\right) \\ &= -\underbrace{\nabla_{\vec{r}_i}\left[\kappa_i (H_i - \bar{H}_i)^2 A_i\right]}_{\text{diagonal}} - \underbrace{\sum_{v_j\in N(v_i)} \nabla_{\vec{r}_i}\left[\kappa_j (H_j - \bar{H}_j)^2 A_j\right]}_{\text{off-diagonal}}.\end{aligned} \tag{C.6}$$

Using the derivatives of geometric primitives in Eq. (C.3), we can assemble the derivatives of local pointwise mean curvature for both the "diagonal" term,

$$\begin{aligned}\nabla_{\vec{r}_i} H_i &= \frac{1}{4}\sum_{\underline{e}_{ij}\in N(v_i)} \nabla_{\vec{r}_i}\frac{l_{ij}\varphi_{ij}}{A_i} \\ &= \frac{1}{4A_i}\sum_{\underline{e}_{ij}\in N(v_i)}\left(\varphi_{ij}\nabla_{\vec{r}_i} l_{ij} + l_{ij}\nabla_{\vec{r}_i}\varphi_{ij}\right) - \frac{H_i}{A_i}\nabla_{\vec{r}_i} A_i \\ &= \frac{1}{A_i}\sum_{\underline{e}_{ij}\in N(v_i)} \frac{1}{2}\left(\int \vec{K}_{ij} + \int \vec{S}_{ij,1}\right) - \frac{2}{3}H_i \int \vec{H}_{ij},\end{aligned} \tag{C.7}$$

and for the "off-diagonal" term,

$$\begin{aligned}\nabla_{\vec{r}_i} H_j &= \frac{1}{4}\sum_{\underline{e}_{jk}\in N(v_j)} \nabla_{\vec{r}_i}\frac{l_{jk}\varphi_{jk}}{A_j} \\ &= \frac{1}{4A_j}\left(l_{jk}\nabla_{\vec{r}_i}\varphi_{jk} + l_{jl}\nabla_{\vec{r}_i}\varphi_{jl} + \varphi_{ji}\nabla_{\vec{r}_i} l_{ji} + l_{ji}\nabla_{\vec{r}_i}\varphi_{ji}\right) - \frac{H_j}{A_j}\nabla_{\vec{r}_i} A_j \\ &= \frac{1}{2A_j}\left(\int \vec{K}_{ij} + \int \vec{S}_{ij,2}\right) - \frac{4}{3}H_j \int \vec{H}_{ij}.\end{aligned} \tag{C.8}$$

When written in the halfedge form, factoring out the fundamental curvature vectors introduced in Eq. (C.4), we obtain the discrete bending force as

$$\begin{aligned}\int \vec{f}_i^b = \sum_{\underline{e}_{ij}\in N(v_i)} & -\left[\kappa_i(H_i-\bar{H}_i)+\kappa_j(H_j-\bar{H}_j)\right]\int \vec{K}_{ij} \\ & +\left[\frac{1}{3}\kappa_i(H_i-\bar{H}_i)(H_i+\bar{H}_i)+\frac{2}{3}\kappa_j(H_j-\bar{H}_j)(H_j+\bar{H}_j)\right]\int 2\vec{H}_{ij} \\ & -\left[\kappa_i(H_i-\bar{H}_i)\int \vec{S}_{ij,1}+\kappa_j(H_j-\bar{H}_j)\int \vec{S}_{ij,2}\right].\end{aligned} \tag{C.9}$$

When the surface is not closed, boundary vertices, $v_i \in \partial\mathcal{M}$, experience an additional force from the Gaussian curvature component,

$$\int \vec{f}_i^b = \text{RHS of Eq. (C.9)} + \sum_{f_{ijk}\in N(v_i)} (-1)^p \nabla_{\vec{r}_i}\angle_{f_{ijk}}, \quad v_i \in \partial\mathcal{M}, \tag{C.10}$$

where p is the number of boundary vertices in face f_{ijk}, and

$$\angle_{f_{ijk}} = \begin{cases} \angle_{ijk} & \text{if } v_j, v_k \notin \partial\mathcal{M}, \\ \angle_{ikj} & \text{if } v_j \in \partial\mathcal{M}, \\ \angle_{ijk} & \text{if } v_k \in \partial\mathcal{M}. \end{cases} \tag{C.11}$$

C.3 Deriving the line tension and diffusion as the shape and chemical derivatives of the Dirichlet energy

Since the discrete Dirichlet energy is constructed on the triangular face and therefore does not involve any neighborhood, we simplify the notation by adopting the convention illustrated in Fig. 1C. The gradient of protein density is given by the slope of the fitted plane over the vertexwise protein density, which is piecewise constant for each face,

$$\nabla_{\vec{\theta}}\,\phi_i = \frac{1}{2A_{ijk}} \sum_{\vec{\underline{e}}_k\in N(f_{ijk})} \phi_k \vec{e}_k^{\,\perp}, \tag{C.12}$$

where we adopt the counterclockwise convention (e.g. $\vec{e}_k = \vec{\underline{e}}_{ji}$) and $(\cdot)^\perp$ represents a 90° counterclockwise rotation in plane of the face, f_{ijk}.

C.3.1 Line tension from the shape derivative of the Dirichlet energy

Substituting the definition of the discrete gradient into the Dirichlet energy (Eq. (28)), we expand the energy in terms of mesh primitives, whose geometric derivatives are given in Eq. (C.3). Additional formulae are needed to compute the geometric derivatives of the outer angles of the triangle (Fig. 1C)

$$\nabla_{\vec{r}_i}\angle_k = \frac{\vec{n}\times\vec{e}_j}{\|\vec{e}_j\|^2} \tag{C.13a}$$

$$\nabla_{\vec{r}_i}\angle_j = \frac{\vec{n}\times\vec{e}_k}{\|\vec{e}_k\|^2} \tag{C.13b}$$

$$\nabla_{\vec{r}_i}\angle_i = -(\nabla_{\vec{r}_i}\angle_k + \nabla_{\vec{r}_i}\angle_j), \tag{C.13c}$$

which arise from the calculation of the L_2 norm of the gradients as the result of vector inner product. When combined, the geometric derivatives for the quadratic gradient term is

$$\begin{aligned}
\nabla_{\vec{r}_i} \left\langle \sum \phi_k \vec{e}_k^{\perp}, \sum \phi_k \vec{e}_k^{\perp} \right\rangle = & \\
& + \phi_k \phi_k \vec{e}_k - 2\phi_j \phi_j \vec{e}_j \\
& + 2\phi_j \phi_i \|\vec{e}_i\| \left(-\hat{e}_j \cos \angle_k + \|\vec{e}_j\| \nabla_{\vec{r}_i} (\cos \angle_k) \right) \\
& + 2\phi_i \phi_k \|\vec{e}_i\| \left(\hat{e}_k \cos \angle_j + \|\vec{e}_k\| \nabla_{\vec{r}_i} (\cos \angle_j) \right) \\
& + 2\phi_j \phi_k \left(-\hat{e}_j \|\vec{e}_k\| \cos \angle_i + \|\vec{e}_j\| \hat{e}_k \cos \angle_i + \|\vec{e}_j\| \|\vec{e}_k\| \nabla_{\vec{r}_i} (\cos \angle_i) \right)
\end{aligned} \tag{C.14}$$

Then we can get the final shape derivative by combining the area gradient, or the mean curvature vector (Eq. (C.4)).

C.3.2 Surface diffusion from the chemical derivative of the Dirichlet energy

In the case where we are evolving the protein distribution, we need the chemical derivative of the Dirichlet energy. Before we look into the discrete case, we can first tackle the problem in the smooth setting, which is a classic textbook example. Using the Green's first identity, or integration by parts on a 2-manifold,

$$\int_{\mathcal{M}} (\psi \Delta_s \varphi + \nabla_{\vec{\theta}} \psi \cdot \nabla_{\vec{\theta}} \varphi) dA = \oint_{\partial \mathcal{M}} \psi \nabla_{\vec{\theta}} \varphi \cdot \vec{n} dS, \tag{C.15}$$

and ignoring the boundary term at the right hand side, we arrive at an alternative expression for the Dirichlet energy,

$$E_d = \frac{1}{2} \int_{\mathcal{M}} \eta \|\nabla_{\vec{\theta}} \phi\|^2 \, dA = -\frac{1}{2} \int_{\mathcal{M}} \eta \phi \Delta_s \phi \, dA. \tag{C.16}$$

The same procedure can be followed in the discrete case. The discrete Dirichlet energy (Eq. (28)) can be written in matrix form,

$$E_d = \frac{1}{2} \eta \boldsymbol{\phi}^\top \tilde{\boldsymbol{G}}^\top \tilde{\boldsymbol{T}} \tilde{\boldsymbol{G}} \boldsymbol{\phi} \tag{C.17}$$

where $\tilde{\boldsymbol{G}}$ is the gradient tensor which maps scalar value on vertices to vector values on faces, and $\tilde{\boldsymbol{T}} = \text{diag}(\boldsymbol{A}^{\text{face}})$ is the $|f| \times |f|$ diagonal matrix with entries corresponding to the area of each mesh triangle face. Through integration by parts on discrete geometry, the discrete Dirichlet energy can be equivalently expressed as

$$E_d = \frac{1}{2} \eta \boldsymbol{\phi}^\top \tilde{\boldsymbol{L}} \boldsymbol{\phi}, \tag{C.18}$$

which is a quadratic form with respect to the cotangent Laplacian matrix, $\tilde{\boldsymbol{L}}$ (*99*, *100*). The chemical derivative of the Dirichlet energy, or the diffusion potential, is

$$\boldsymbol{\mu}^d = -\nabla_\phi E_d = -\eta \int \Delta_s \boldsymbol{\phi} = -\eta \tilde{\boldsymbol{L}} \boldsymbol{\phi}. \tag{C.19}$$

In other words, the chemical gradient flow of the Dirichlet energy is the diffusion equation. Note that $\tilde{\boldsymbol{L}} = \tilde{\boldsymbol{G}}^\top \tilde{\boldsymbol{T}} \tilde{\boldsymbol{G}}$, $\tilde{\boldsymbol{G}}^\top$ is referred to as the discrete divergence operator that maps face vectors to scalars on vertices (*137*).

D Discrete-smooth comparison on spheroid

The smooth-discrete comparison is done on the spheroid with the parametrization,

$$(x, y, z) = (a \cos \beta \cos \theta, a \cos \beta \sin \theta, c \sin \beta), \tag{D.1}$$

where $a = 1$, $b = 0.5$, β is the parametric latitude and θ is the azimuth coordinate. All geometric measurements of the smooth geometry used for benchmarking were obtained using the symbolic algbra software *Sympy*. The corresponding discrete measurements are computed using `Mem3DG`, whose input spheroid mesh is mapped from a subdivided icosphere. The subsequent error norms for local measurements are computed based on definitions used in Section 2.4.

E Mesh regularization and mesh mutation

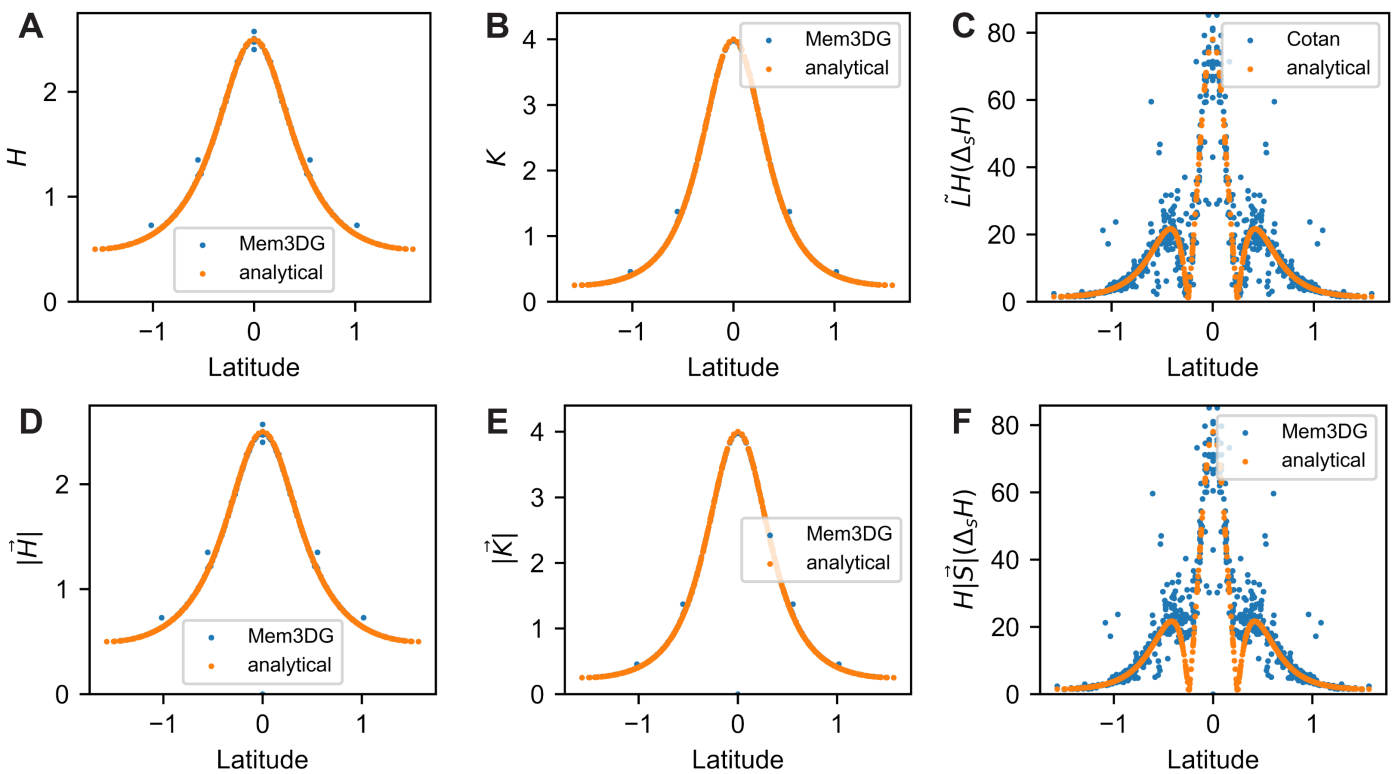


Figure E.1: Pointwise magnitude comparison of continuous and discrete measurements: A) scalar mean curvature, B) scalar Gaussian curvature, C) (scalar) bi-Laplacian term ∇H based on the cotan formula, D) vector mean curvature, E) vector Gaussian curvature, and F) (vector) bi-Laplacian term based on Schlafli vector. Note that the result of the cotangent Laplacian approach in C) produces a scalar result while our approach using the Schlafli vector in F) is a vector result, thus their direct comparison is not meaningful.

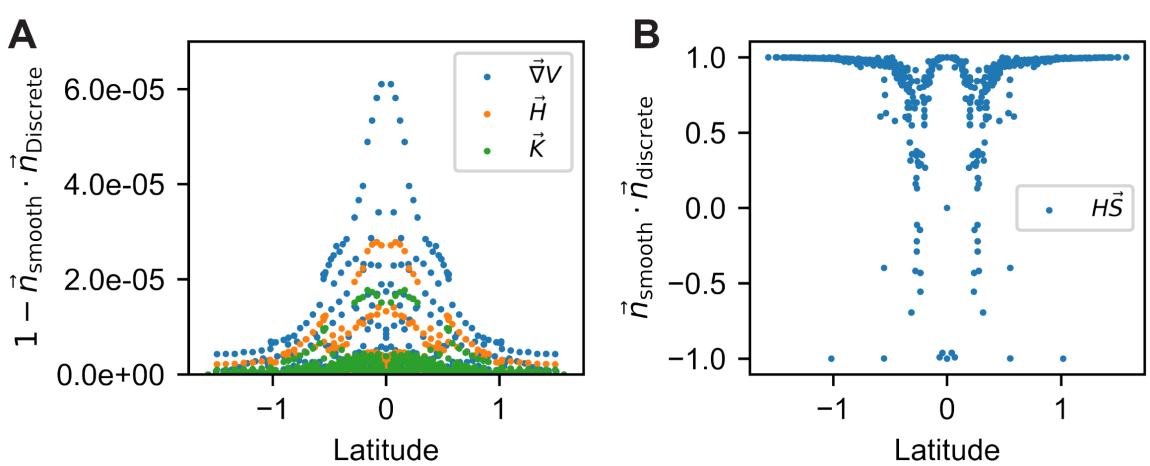


Figure E.2: Pointwise directional comparison of continuous and discrete measurements: discrete vertex normal based on A) volume gradient, $\nabla_{\vec{r}} V$, mean curvature vector, $\vec{H}$, and Gaussian curvature vector, $\vec{K}$, and B) Schlafli vector, $H\vec{S}$

E.1 Mesh mutation

Mesh mutation and refinement in combination with vertex shifting are the default methods to ensure that the mesh remains well-conditioned and well-resolved during simulation. Mesh mutations include edge flipping, collapsing, and splitting, which change the connectivity of the mesh. Vertex shifting moves the vertex to the barycenter of the fan neighborhood without changing the mesh topology (Fig. 1A). `Mem3DG` has a suite of possible criteria to initiate mesh mutation. Here we list the most important ones: 1) flip the edge of the non-Delaunay diamond neighborhood (Fig. 1B), 2) collapse the shortest edge in a skinny triangle face, 3) split the edge with high (geodesic) curvature. For additional details please refer to the software documentation.

For practical use, although mesh mutation introduces additional complexity in data write-out and computational costs associated with varying (usually growing) mesh size, it nevertheless provides a robust algorithm to ensure good mesh quality needed for valid discrete-smooth comparisons (Section 2.5.4) in static frames. For dynamical simulation, mesh mutations introduce an arbitrary interpolation of state variables, such as the position, velocity, and protein density. Rigorous study on how to interpolate these quantities to ensure the conservation of energy, momentum, and mass, remains to be done. Similarly, the interpolation used in this study introduce discontinuities of curvature and can create jumps in forces; This is particularly severe for terms with higher-order derivatives such as the biharmonic term in bending force (Eq. (22)).

E.2 Mesh regularization

Mesh regularization can be used when mesh mutations are not desired. The regularization force consists of three weakly enforced constraining forces, including the edge (length), $\vec{f}^e$, face (area), $\vec{f}^f$, and conformality (angle), $\vec{f}^c$, regularization forces,

$$\vec{f}_i^e = -K_e \sum_{e_{ij} \in N(v_i)} \frac{(l_{ij} - \bar{l}_{ij})}{\bar{l}_{ij}} \nabla_{\vec{r}_i} l_{ij}, \tag{E.1a}$$

$$\vec{f}_i^f = -K_f \sum_{f_{ijk} \in N(v_i)} \frac{(A_{ijk} - \bar{A}_{ijk})}{\bar{A}_{ijk}} \nabla_{\vec{r}_i} A_{ijk}, \tag{E.1b}$$

$$\vec{f}_i^c = -K_c \sum_{e_{ij} \in N(v_i)} \frac{(\lambda_{ij} - \bar{\lambda}_{ij})}{\bar{\lambda}_{ij}} \nabla_{\vec{r}_i} \lambda_{ij}, \tag{E.1c}$$

which are in the order of strongest to weakest. The length-cross-ratio, $\lambda_{ij} = l_{il} l_{jk} / l_{ki} l_{jl}$ is a metric of discrete conformality on triangulated mesh, where the indices is illustrated in Fig. 1A-B (*139*). Regularization forces require the input of a reference value for geometric measurements, $\bar{l}$, $\bar{A}$, and $\bar{\lambda}$, which can be derived from a well-conditioned reference mesh (usually the initial input mesh for the simulation). The intensity of each regularization force is controlled with parameters K_e, K_f, and K_c.

For practical use, regularization constraints should be minimally imposed because of their impact on system dynamics. In the worst case, regularization constraints can prevent the optimizer from reaching an energy minima. Thus a good practice is to start a simulation with no, conformality, face area, and finally edge length regularization, and subsequently raise the intensity/type of constraints based on the mesh quality desired. We do not recommend imposing constraints stronger than the face areal constraints, $\vec{f}^f$.

References

1. J. T. Groves, J. Kuriyan, *Nature structural & molecular biology* **17**, 659–665, ISSN: 1545-9993, DOI 10/d37mbf (June 2010).
2. X. Cheng, J. C. Smith, *Chemical Reviews* **119**, 5849–5880, ISSN: 0009-2665, DOI 10/gfvk9g (May 2019).
3. M. Anitei, B. Hoflack, *Nature Cell Biology* **14**, 11–19, ISSN: 1476-4679, DOI 10/fx37b7 (Jan. 2012).

4. J. S. Bonifacino, B. S. Glick, *Cell* **116**, 153–166, ISSN: 0092-8674, 1097-4172, DOI 10/b7x4x3 (Jan. 2004).
5. J. M. Herrmann, A. Spang, in *Membrane Trafficking: Second Edition*, ed. by B. L. Tang (Springer, New York, NY, 2015), pp. 1–12, ISBN: 978-1-4939-2309-0, DOI 10.1007/978-1-4939-2309-0_1.
6. H. T. McMahon, E. Boucrot, *Nature Reviews Molecular Cell Biology* **12**, 517–533, ISSN: 1471-0080, DOI 10/b49pxb (Aug. 2011).
7. J. Zimmerberg, M. M. Kozlov, *Nature Reviews Molecular Cell Biology* **7**, 9–19, ISSN: 1471-0080, DOI 10/dfh54v (Jan. 2006).
8. K. Farsad, P. D. Camilli, *Current Opinion in Cell Biology* **15**, 372–381, ISSN: 0955-0674, DOI 10/ctw22q (Aug. 2003).
9. O. Avinoam, M. Schorb, C. J. Beese, J. A. G. Briggs, M. Kaksonen, *Science* **348**, 1369–1372, ISSN: 0036-8075, 1095-9203, DOI 10.1126/science.aaa9555 (June 2015).
10. M. Terasaki *et al.*, *Cell* **154**, 285–296, ISSN: 00928674, DOI 10.1016/j.cell.2013.06.031 (July 2013).
11. Y. Shibata *et al.*, *Cell* **143**, 774–788, ISSN: 00928674, DOI 10.1016/j.cell.2010.11.007 (Nov. 2010).
12. H. T. McMahon, J. L. Gallop, *Nature* **438**, 590–596, ISSN: 1476-4687, DOI 10.1038/nature04396 (Dec. 2005).
13. H. T. McMahon, E. Boucrot, *Journal of Cell Science* **128**, 1065–1070, ISSN: 0021-9533, DOI 10/f64xp3 (Mar. 2015).
14. M. M. Kozlov *et al.*, *Current Opinion in Cell Biology*, Cell Organelles **29**, 53–60, ISSN: 0955-0674, DOI 10/f6f2f2 (Aug. 2014).
15. C. S. Scheve, P. A. Gonzales, N. Momin, J. C. Stachowiak, *Journal of the American Chemical Society* **135**, 1185–1188, ISSN: 0002-7863, DOI 10.1021/ja3099867 (Jan. 2013).
16. J. C. Stachowiak, C. C. Hayden, D. Y. Sasaki, *Proceedings of the National Academy of Sciences* **107**, 7781–7786, ISSN: 0027-8424, 1091-6490, DOI 10.1073/pnas.0913306107 (Apr. 2010).
17. J. C. Stachowiak *et al.*, *Nature Cell Biology* **14**, 944–949, ISSN: 1476-4679, DOI 10.1038/ncb2561 (Sept. 2012).
18. J. C. Stachowiak, F. M. Brodsky, E. A. Miller, *Nature Cell Biology* **15**, 1019–1027, ISSN: 1476-4679, DOI 10.1038/ncb2832 (Sept. 2013).
19. F. Yuan *et al.*, *Proceedings of the National Academy of Sciences* **118**, ISSN: 0027-8424, 1091-6490, DOI 10.1073/pnas.2017435118 (Mar. 2021).
20. M. Chabanon, J. C. Stachowiak, P. Rangamani, *Wiley Interdisciplinary Reviews: Systems Biology and Medicine* **9**, e1386, ISSN: 19395094, DOI 10.1002/wsbm.1386 (Sept. 2017).
21. G. Knott, H. Marchman, D. Wall, B. Lich, *Journal of Neuroscience* **28**, 2959–2964, ISSN: 0270-6474, 1529-2401, DOI 10.1523/JNEUROSCI.3189-07.2008 (Mar. 2008).
22. C. J. Peddie, L. M. Collinson, *Micron* **61**, 9–19, ISSN: 0968-4328, DOI 10.1016/J.MICRON.2014.01.009 (June 2014).
23. B. Titze, C. Genoud, *Biology of the Cell* **108**, 307–323, ISSN: 02484900, DOI 10.1111/boc.201600024 (Nov. 2016).
24. G. Knott, C. Genoud, *Journal of Cell Science* **126**, 4545–4552, ISSN: 0021-9533, 1477-9137, DOI 10.1242/jcs.124123 (Oct. 2013).
25. K. D. Micheva, S. J. Smith, *Neuron* **55**, 25–36, ISSN: 0896-6273, DOI 10.1016/j.neuron.2007.06.014 (July 2007).
26. E. Villa, M. Schaffer, J. M. Plitzko, W. Baumeister, *Current Opinion in Structural Biology* **23**, 771–777, ISSN: 0959440X, DOI 10.1016/j.sbi.2013.08.006 (Oct. 2013).
27. J. Nixon-Abell *et al.*, *Science (80-.).*, ISSN: 10959203, DOI 10.1126/science.aaf3928 (2016).
28. D. P. Hoffman *et al.*, *Science* **367**, ISSN: 0036-8075, 1095-9203, DOI 10.1126/science.aaz5357 (Jan. 2020).
29. C. S. Xu *et al.*, *Elife* **6**, ISSN: 2050-084X, DOI 10.7554/eLife.25916 (May 2017).

30. T. G. Frey, C. A. Mannella, *Trends in Biochemical Sciences* **25**, 319–324, ISSN: 0968-0004, DOI 10/dzjjnc (July 2000).
31. Y. Wu *et al.*, *Proceedings of the National Academy of Sciences* **114**, E4859–E4867, ISSN: 0027-8424, 1091-6490, DOI 10.1073/pnas.1701078114 (June 2017).
32. C. T. Lee *et al.*, *PLoS computational biology* **16**, e1007756 (2020).
33. R. Mendelsohn *et al.*, *Journal of Comparative Neurology*, cne.25254, ISSN: 0021-9967, 1096-9861, DOI 10/gm72r9 (Oct. 2021).
34. M. Salfer, J. F. Collado, W. Baumeister, R. Fernández-Busnadiego, A. Martínez-Sánchez, *PLOS Computational Biology* **16**, e1007962, ISSN: 1553-7358, DOI 10/gmnx32 (Aug. 2020).
35. H. Tamada, J. Blanc, N. Korogod, C. C. Petersen, G. W. Knott, *eLife* **9**, e56384, ISSN: 2050-084X, DOI 10/gmqhr4 (Dec. 2020).
36. K. M. Davies *et al.*, *Proceedings of the National Academy of Sciences* **108**, 14121–14126, ISSN: 0027-8424, 1091-6490, DOI 10.1073/pnas.1103621108 (Aug. 2011).
37. K. M. Davies, C. Anselmi, I. Wittig, J. D. Faraldo-Gómez, W. Kühlbrandt, *Proceedings of the National Academy of Sciences* **109**, 13602–13607, ISSN: 0027-8424, 1091-6490, DOI 10.1073/pnas.1204593109 (Aug. 2012).
38. N. Ramakrishnan, P. B. Sunil Kumar, R. Radhakrishnan, *Physics Reports*, Mesoscale Computational Studies of Membrane Bilayer Remodeling by Curvature-Inducing Proteins **543**, 1–60, ISSN: 0370-1573, DOI 10.1016/j.physrep.2014.05.001 (Oct. 2014).
39. W. Helfrich, *Zeitschrift für Naturforschung c* **28**, 693–703 (1973).
40. D. Steigmann, *Archive for Rational Mechanics and Analysis* **150**, 127–152 (1999).
41. F. Campelo, C. Arnarez, S. J. Marrink, M. M. Kozlov, *Advances in Colloid and Interface Science*, Special Issue in Honour of Wolfgang Helfrich **208**, 25–33, ISSN: 0001-8686, DOI 10/f57v4s (June 2014).
42. H. Naito, M. Okuda, O.-Y. Zhong-can, *Physical Review Letters* **74**, 4345–4348, ISSN: 0031-9007, 1079-7114, DOI 10.1103/PhysRevLett.74.4345 (May 1995).
43. M. Deserno, *Chemistry and Physics of Lipids* **185**, 11–45, ISSN: 00093084, DOI 10.1016/j.chemphyslip.2014.05.001 (Jan. 2015).
44. D. Argudo, N. P. Bethel, F. V. Marcoline, M. Grabe, *Biochimica et Biophysica Acta (BBA) - Biomembranes*, New Approaches for Bridging Computation and Experiment on Membrane Proteins **1858**, 1619–1634, ISSN: 0005-2736, DOI 10.1016/j.bbamem.2016.02.003 (July 2016).
45. D. Argudo, N. P. Bethel, F. V. Marcoline, C. W. Wolgemuth, M. Grabe, *Biophysical Journal* **112**, 2159–2172, ISSN: 1542-0086, DOI 10.1016/j.bpj.2017.03.040 (May 2017).
46. M. Hamm, M. Kozlov, *The European Physical Journal E* **3**, 323–335, ISSN: 1292-8941, DOI 10/cp87nn (Dec. 2000).
47. A. Agrawal, D. J. Steigmann, *Continuum Mechanics and Thermodynamics* **21**, 57–82, ISSN: 0935-1175, 1432-0959, DOI 10.1007/s00161-009-0102-8 (June 2009).
48. P. Rangamani *et al.*, *Biomechanics and Modeling in Mechanobiology* **13**, 697–711, ISSN: 1617-7959, 1617-7940, DOI 10.1007/s10237-013-0528-6 (Aug. 2014).
49. S. Liese *et al.*, *Proceedings of the National Academy of Sciences* **117**, 28614–28624, ISSN: 0027-8424, 1091-6490, DOI 10/gm3677 (Nov. 2020).
50. S. Liese, A. Carlson, *Biophysical Journal* **120**, 2482–2489, ISSN: 00063495, DOI 10/gm369j (June 2021).
51. P. Bassereau *et al.*, *Journal of Physics D: Applied Physics* **51**, 343001, ISSN: 0022-3727, DOI 10.1088/1361-6463/aacb98 (July 2018).
52. T. Baumgart, B. R. Capraro, C. Zhu, S. L. Das, *Annual Review of Physical Chemistry* **62**, 483–506, ISSN: 0066-426X, 1545-1593, DOI 10/bzw68c (May 2011).
53. C. T. Lee, M. Akamatsu, P. Rangamani, *Current Opinion in Cell Biology* **71**, 38–45, ISSN: 09550674, DOI 10.1016/j.ceb.2021.01.011 (Aug. 2021).
54. A. E. Carlsson, *Current Opinion in Cell Biology*, Cell Architecture **50**, 1–7, ISSN: 0955-0674, DOI 10/gdnqms (Feb. 2018).

55. J. Liu, M. Kaksonen, D. G. Drubin, G. Oster, *Proceedings of the National Academy of Sciences* **103**, 10277–10282 (2006).
56. J. Liu, Y. Sun, D. G. Drubin, G. F. Oster, *PLoS biology* **7**, e1000204 (2009).
57. J. E. Hassinger, G. Oster, D. G. Drubin, P. Rangamani, *Proceedings of the national academy of sciences* **114**, E1118–E1127 (2017).
58. H. Alimohamadi, R. Vasan, J. E. Hassinger, J. C. Stachowiak, P. Rangamani, *Molecular biology of the cell* **29**, 2024–2035 (2018).
59. M. Saleem *et al.*, *Nature Communications* **6**, 6249, ISSN: 2041-1723, DOI 10.1038/ncomms7249 (May 2015).
60. M. Tachikawa, A. Mochizuki, *Proceedings of the National Academy of Sciences* **114**, 5177–5182, ISSN: 0027-8424, 1091-6490, DOI 10.1073/pnas.1619264114 (May 2017).
61. R. Natesan, R. Radhakrishnan, in *Advances in Planar Lipid Bilayers and Liposomes* (Elsevier, 2015), vol. 22, pp. 129–175, ISBN: 978-0-12-802878-0, DOI 10.1016/bs.adplan.2015.06.004.
62. N. J. Agrawal, R. Radhakrishnan, *Physical Review E* **80**, 011925, ISSN: 1539-3755, 1550-2376, DOI 10/bw4hpw (July 2009).
63. N. J. Agrawal, J. Nukpezah, R. Radhakrishnan, *PLoS Computational Biology* **6**, ed. by C. V. Rao, e1000926, ISSN: 1553-7358, DOI 10/bjppbr (Sept. 2010).
64. R. Ma, J. Berro, *Biophysical Journal* **120**, 1625–1640, ISSN: 0006-3495, DOI 10/gm39f2 (May 2021).
65. A. Guckenberger, S. Gekle, *Journal of Physics: Condensed Matter* **29**, 203001, ISSN: 0953-8984, 1361-648X, DOI 10.1088/1361-648X/aa6313 (May 2017).
66. M. Akamatsu *et al.*, *Elife* **9**, e49840 (2020).
67. T. Willmore, *Riemannian Geometry* (Oxford University Press, 1996), (https://books.google.com/books?id=L6qtDAEACAAJ).
68. E. A. Evans, *Mechanics and thermodynamics of biomembranes* (CRC press, 2018).
69. P. B. Canham, *Journal of theoretical biology* **26**, 61–81 (1970).
70. J. T. Jenkins, *SIAM Journal on Applied Mathematics* **32**, 755–764 (1977).
71. J. T. Jenkins, *Journal of mathematical biology* **4**, 149–169 (1977).
72. U. Seifert, *Advances in physics* **46**, 13–137 (1997).
73. Q. Du, C. Liu, X. Wang, *Journal of Computational Physics* **198**, 450–468, ISSN: 0021-9991, DOI 10.1016/j.jcp.2004.01.029 (Aug. 2004).
74. Q. Du, X. Wang, *International Journal of Numerical Analysis and Modeling* (2006).
75. D. Salac, M. Miksis, *Journal of Computational Physics* **230**, 8192–8215 (2011).
76. F. Feng, W. S. Klug, *Journal of Computational Physics* **220**, 394–408, ISSN: 00219991, DOI 10.1016/j.jcp.2006.05.023 (Dec. 2006).
77. L. Ma, W. S. Klug, *Journal of Computational Physics* **227**, 5816–5835 (2008).
78. C. M. Elliott, B. Stinner, *Journal of Computational Physics* **229**, 6585–6612, ISSN: 00219991, DOI 10.1016/j.jcp.2010.05.014 (Sept. 2010).
79. R. Rangarajan, H. Gao, *Journal of Computational Physics* **297**, 266–294, ISSN: 00219991, DOI 10.1016/j.jcp.2015.05.001 (Sept. 2015).
80. R. A. Sauer, T. X. Duong, K. K. Mandadapu, D. J. Steigmann, *Journal of Computational Physics* **330**, 436–466, ISSN: 00219991, DOI 10.1016/j.jcp.2016.11.004 (Feb. 2017).
81. R. Vasan, S. Rudraraju, M. Akamatsu, K. Garikipati, P. Rangamani, *Soft Matter* **16**, 784–797, ISSN: 1744-683X, 1744-6848, DOI 10.1039/C9SM01494B (Jan. 2020).
82. A. Torres-Sánchez, D. Millán, M. Arroyo, *Journal of Fluid Mechanics* **872**, 218–271, ISSN: 0022-1120, 1469-7645, DOI 10.1017/jfm.2019.341 (Aug. 2019).
83. D. Auddya *et al.*, "Biomembranes Undergo Complex, Non-Axisymmetric Deformations Governed by Kirchhoff-Love Kinematics and Revealed by a Three Dimensional Computational Framework", Preprint (Biophysics, Jan. 2021), DOI 10.1101/2021.01.28.428578.
84. G. Gompper, D. Kroll, *Journal de Physique I* **6**, 1305–1320 (1996).

85. F. Jülicher, *Journal de physique ii* **6**, 1797–1824 (1996).
86. Y. Kantor, D. R. Nelson, *Physical Review A* **36**, 4020 (1987).
87. X. Bian, S. Litvinov, P. Koumoutsakos, *Computer Methods in Applied Mechanics and Engineering* **359**, 112758, ISSN: 00457825, DOI 10.1016/j.cma.2019.112758 (Feb. 2020).
88. K. A. Brakke, *Experimental Mathematics* **1**, 141–165, ISSN: 1058-6458, 1944-950X, DOI 10.1080/10586458.1992.10504253 (Jan. 1992).
89. Y. Jie, L. Quanhui, L. Jixing, O.-Y. Zhong-Can, *Physical Review E* **58**, 4730–4736, ISSN: 1063-651X, 1095-3787, DOI 10.1103/PhysRevE.58.4730 (Oct. 1998).
90. D. M. Kroll, G. Gompper, *Science* **255**, 968–971, ISSN: 0036-8075, 1095-9203, DOI 10.1126/science.1546294 (Feb. 1992).
91. E. Atilgan, S. X. Sun, *The Journal of Chemical Physics* **126**, 095102, ISSN: 0021-9606, 1089-7690, DOI 10.1063/1.2483862 (Mar. 2007).
92. A. H. Bahrami, G. Hummer, *ACS Nano* **11**, 9558–9565, ISSN: 1936-0851, 1936-086X, DOI 10.1021/acsnano.7b05542 (Sept. 2017).
93. H. Noguchi, G. Gompper, *Proceedings of the National Academy of Sciences* **102**, 14159–14164, ISSN: 0027-8424, 1091-6490, DOI 10.1073/pnas.0504243102 (Oct. 2005).
94. K. Tsai *et al.*, *Physical Biology* **17**, 065011, ISSN: 1478-3975, DOI 10.1088/1478-3975/abb208 (Oct. 2020).
95. A. Guckenberger, M. P. Schraml, P. G. Chen, M. Leonetti, S. Gekle, *Computer Physics Communications* **207**, 1–23, ISSN: 00104655, DOI 10.1016/j.cpc.2016.04.018 (Oct. 2016).
96. J. R. Shewchuk, *Proc. 11th Int. Meshing Roundtable* **94720**, 115–126 (2002).
97. W. Pezeshkian, M. König, S. J. Marrink, J. H. Ipsen, *Frontiers in molecular biosciences* **6**, 59 (2019).
98. M. Sadeghi, T. R. Weikl, F. Noé, *The Journal of Chemical Physics* **148**, 044901, ISSN: 0021-9606, 1089-7690, DOI 10.1063/1.5009107 (Jan. 2018).
99. A. Chern, *Discrete Differential Geometry*, Nov. 19, 2020.
100. K. Crane, *DISCRETE DIFFERENTIAL GEOMETRY: AN APPLIED INTRODUCTION*, Apr. 13, 2020.
101. E. A. Evans, *Biophysical Journal* **14**, 923–931, ISSN: 0006-3495, DOI 10/fftb7r (Dec. 1974).
102. M. P. Do Carmo, *Differential geometry of curves and surfaces: revised and updated second edition* (Courier Dover Publications, 2016).
103. J. M. Lee, *Introduction to Smooth Manifolds* (Springer New York, New York, NY, 2012), vol. 218, ISBN: 978-1-4419-9981-8, DOI 10.1007/978-1-4419-9982-5.
104. A. I. Bobenko, Ed., *Discrete Differential Geometry* (Birkhäuser, Basel ; Boston, 2008), ISBN: 978-3-7643-8620-7.
105. R. Phillips, J. Kondev, J. Theriot, *Physical Biology of the Cell* (Garland Science, Taylor & Francis Group, New York, Nov. 2008), ISBN: 978-0815341635, (http://www.worldcat.org/search?qt=worldcat_org_all&q=0815341636).
106. M. Meyer, M. Desbrun, P. Schröder, A. H. Barr, in *Visualization and Mathematics III*, ed. by G. Farin *et al.* (Springer Berlin Heidelberg, Berlin, Heidelberg, 2003), pp. 35–57, ISBN: 978-3-642-05682-6, DOI 10.1007/978-3-662-05105-4_2.
107. M. Wardetzky, S. Mathur, F. Kälberer, E. Grinspun, presented at the ACM SIGGRAPH ASIA 2008 Courses on - SIGGRAPH Asia '08, pp. 1–5, DOI 10.1145/1508044.1508063.
108. M. Wardetzky, in *Discrete Differential Geometry*, ed. by A. I. Bobenko, J. M. Sullivan, P. Schröder, G. M. Ziegler (Birkhäuser Basel, Basel, 2008), vol. 38, pp. 275–286, ISBN: 978-3-7643-8620-7, DOI 10.1007/978-3-7643-8621-4_15.
109. M. Deserno.
110. N. Sharp, K. Crane, *et al.*, *geometry-central*, www.geometry-central.net, 2019.
111. N. Sharp, K. Crane, *et al.*, *hapPLY*, https://github.com/nmwsharp/happly, 2018.
112. R. Rew, G. Davis, *IEEE computer graphics and applications* **10**, 76–82 (1990).
113. N. Sharp *et al.*, *Polyscope*, www.polyscope.run, 2019.

114. W. Jakob, J. Rhinelander, D. Moldovan, *pybind11 – Seamless operability between C++11 and Python*, https://github.com/pybind/pybind11, 2017.

115. J. Nocedal, S. J. Wright, *Numerical Optimization* (Springer, New York, 2nd ed, 2006), ISBN: 978-0-387-30303-1.

116. T. J. Hughes, *The finite element method: linear static and dynamic finite element analysis* (Courier Corporation, 2012).

117. F. Jülicher, R. Lipowsky, *Physical Review E* **53**, 2670 (1996).

118. H. Deuling, W. Helfrich, *Biophysical journal* **16**, 861 (1976).

119. E. Evans, Y.-C. Fung, *Microvascular Research* **4**, 335–347, ISSN: 0026-2862, DOI 10.1016/0026-2862(72)90069-6 (1972).

120. H. Alimohamadi, A. S. Smith, R. B. Nowak, V. M. Fowler, P. Rangamani, *bioRxiv*, 668582, DOI 10.1101/668582 (Nov. 2019).

121. H.-H. Gerdes, R. N. Carvalho, *Current opinion in cell biology* **20**, 470–475 (2008).

122. K. M. Pearce *et al.*, *Molecular Biology of the Cell* **31**, 683–694, ISSN: 1059-1524, DOI 10.1091/mbc.E19-09-0536 (Mar. 2020).

123. H. Alimohamadi, B. Ovryn, P. Rangamani, *Scientific reports* **10**, 1–15 (2020).

124. A. Datar *et al.*, *Biophysical Journal* **117**, 880–891, ISSN: 00063495, DOI 10.1016/j.bpj.2019.07.046 (Sept. 2019).

125. P. A. Pullarkat, P. Dommersnes, P. Fernández, J.-F. Joanny, A. Ott, *Physical Review Letters* **96**, 048104, ISSN: 0031-9007, 1079-7114, DOI 10.1103/PhysRevLett.96.048104 (Feb. 2006).

126. J. Plateau, *Experimental and Theoretical Statics of Liquids Subject to Molecular Forces Only*, 1873.

127. L. Rayleigh, *On the Instability of Jets*, 1878.

128. O.-Y. Zhong-can, W. Helfrich, *Physcial Review A*, 9.

129. R. Bar-Ziv, E. Moses, *Physical review letters* **73**, 1392 (1994).

130. J. Sanborn, K. Oglęcka, R. S. Kraut, A. N. Parikh, *Faraday Discuss.* **161**, 167–176, ISSN: 1359-6640, 1364-5498, DOI 10.1039/C2FD20116J (2013).

131. S. Watanabe *et al.*, *Nature* **504**, 242–247, ISSN: 1476-4687, DOI 10.1038/nature12809 (Dec. 2013).

132. T. Baumgart, S. T. Hess, W. W. Webb, *Nature* **425**, 821–824, ISSN: 0028-0836, 1476-4687, DOI 10.1038/nature02013 (Oct. 2003).

133. K. Crane, C. Weischedel, M. Wardetzky, *Communications of the ACM* **60**, 90–99, ISSN: 00010782, DOI 10.1145/3131280 (Oct. 2017).

134. D. Serwas *et al.*, "Actin Force Generation in Vesicle Formation: Mechanistic Insights from Cryo-Electron Tomography", Preprint (Cell Biology, June 2021), DOI 10.1101/2021.06.28.450262.

135. K. Crane, *Lecture slide in discrete differential geometry*, Apr. 15, 2021.

136. J. Steiner, in *Jacob Steiner's Gesammelte Werke: Herausgegeben auf Veranlassung der königlich preussischen Akademie der Wissenschaften*, ed. by K. Weierstrass (Cambridge University Press, 2013), vol. 2, pp. 171–176, DOI 10.1017/CBO9781139567947.016.

137. A. Jacobson, D. Panozzo, *et al.*, *libigl: A simple C++ geometry processing library*, https://libigl.github.io/, 2018.

138. E. Grinspun, A. N. Hirani, M. Desbrun, P. Schröder, presented at the Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 62–67.

139. Y. Soliman *et al.*, *ACM Trans. Graph.* **40**, ISSN: 0730-0301, DOI 10.1145/3450626.3459759, (https://doi.org/10.1145/3450626.3459759) (July 2021).