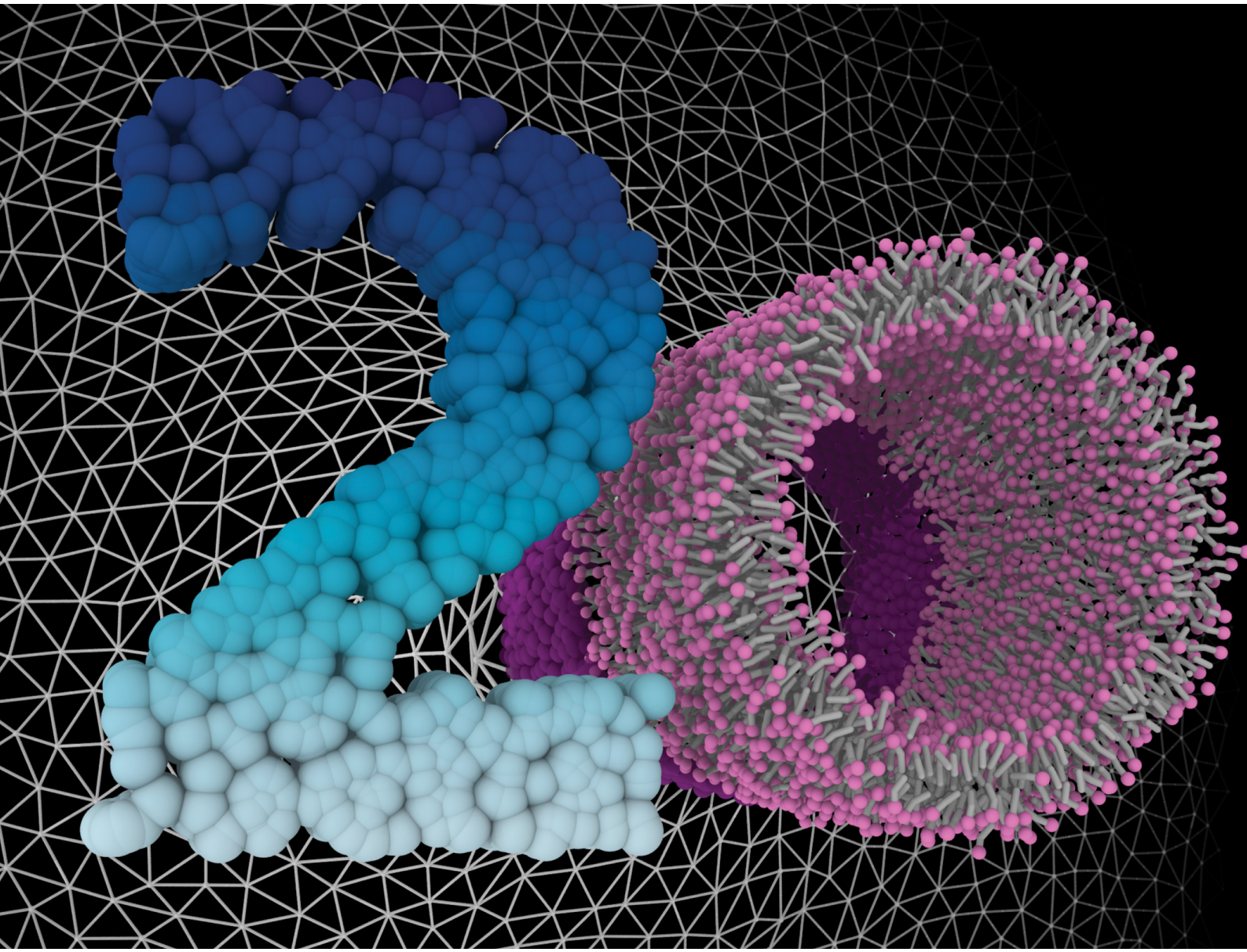
Volume 21
Number 40
28 October 2025
Pages 7729–7896

Soft Matter

rsc.li/soft-matter-journal



ISSN 1744-6848

ROYAL SOCIETY OF CHEMISTRY

TUTORIAL REVIEW
Anđela Šarić *et al.*
A tutorial for mesoscale computer simulations of lipid membranes: tether pulling, tubulation and fluctuations

20 YEARS ANNIVERSARY

Soft Matter

TUTORIAL REVIEW

View Article Online
View Journal | View Issue

Cite this: *Soft Matter*, 2025, **21**, 7736

A tutorial for mesoscale computer simulations of lipid membranes: tether pulling, tubulation and fluctuations

Maitane Muñoz-Basagoiti, †[a] Felix Frey, †[a] Billie Meadowcroft, †[ab] Miguel Amaral, †[a] Adam Prada †[a] and Anđela Šarić *[a]

Lipid membranes and membrane deformations are a long-standing area of research in soft matter and biophysics. Computer simulations have complemented analytical and experimental approaches as one of the pillars in the field. However, setting up and using membrane simulations can come with barriers due to the multidisciplinary effort involved and the vast choice of existing simulations models. In this review, we introduce the non-expert reader to coarse-grained membrane simulations at the mesoscale. Firstly, we give a concise overview of the modelling approaches to study fluid membranes, together with guidance to more specialized references. Secondly, we provide a conceptual guide on how to develop mesoscale membrane simulations. Lastly, we construct a hands-on tutorial on how to apply mesoscale membrane simulations, by providing a pedagogical examination of membrane tether pulling, shape and mechanics of membrane tubes, and membrane fluctuations with three different membrane models, and discussing them in terms of their scope and how resource-intensive they are. To ease the reader's venture into the field, we provide a repository with ready-to-run tutorials.

Received 12th February 2025,
Accepted 21st July 2025

DOI: 10.1039/d5sm00148j

rsc.li/soft-matter-journal

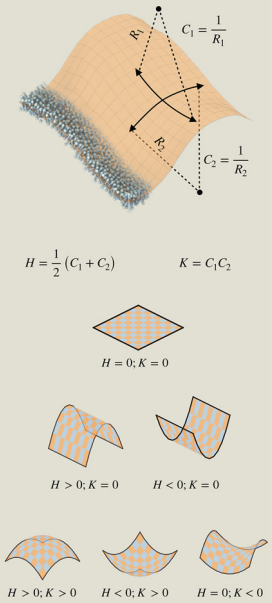
1 Introduction

One of the most astonishing results of biological evolution is the genesis of fluid lipid membranes, which have evolved to be remarkably versatile and ubiquitous in biological systems. The relevance of this evolutionary step can be best seen in the outer plasma membrane and the compartmentalization of biological cells, which are fundamentally achieved *via* fluid lipid membranes. This includes the membrane-enclosed organelles such as the endoplasmic reticulum, the Golgi apparatus, mitochondria or the cell nucleus.[1] One could naïvely assume that in order to shield the interior of a compartment from the exterior, membranes will be rigid and stable, but in fact the opposite is true.[2] Cellular membranes exhibit a rich phenomenology of shapes that are drastically remodelled across different length-scales and in various cellular processes such as endo- and exocytosis, cell adhesion, cell migration and cell division.[3] This shows that fluid lipid membranes combine complementary physical properties: on the one hand, they spontaneously self-assemble from lipid molecules in an aqueous environment, forming closed surfaces that define and separate the inner from the outer; on the other hand, due to the relatively weak forces acting between the lipids that form them, lipid membranes typically operate in the fluid phase in the cell environment and are thus an extremely flexible and malleable material.[4]

Due to their biological relevance, fluid lipid membranes have been studied for decades using experimental, theoretical and computational approaches. As a result, the field of membrane biophysics has experienced many breakthroughs and it is relatively mature compared to other areas of biophysics. It has been extensively reviewed from the experimental,[6–10] theoretical[3,11–14] as well as computational perspectives.[15–18] Nonetheless, the field is far from being saturated. With the advent of new imaging concepts such as cryo-electron microscopy and super-resolution microscopy, the experimental sciences have recently experienced a resolution revolution.[19,20] Biological phenomena that take place on membranes can now be visualized with nanometer resolution, and investigation of previously unobservable processes is becoming tractable. For example, it is now possible to understand with an unprecedented resolution how supramolecular protein assemblies such as the dynamic ESCRT-III filaments and the clathrin coats remodel membranes,[21,22] or how changing the membrane composition can lead to membrane fission.[23] Likewise, in recent years the material properties of fluid lipid membranes have gathered the interest of the soft matter community, with membranes being used as platforms for self-assembly[24,25] and biomimetic design.[26]

[a] *Institute of Science and Technology Austria, Am Campus 1, 3400 Klosterneuburg, Austria. E-mail: andela.saric@ista.ac.at*
[b] *University College London, Gower St, London WC1E 6BT, UK*
† MM-B, FF, BM, MA and AP contributed equally to this work.

Box I: Basics of membrane physics



• Geometry of surfaces: a lipid membrane can be described as a 2D surface embedded in a 3D space. To describe its shape, it is necessary to define the two principal curvatures, C_1 and C_2, at any point of the surface. The magnitude of $|C_i| = 1/R_i$ and $C_i > 0$ if the surface bulges outwards and *vice versa*. Two combinations of the principal curvatures known as the mean curvature $H = (C_1 + C_2)/2$ and Gaussian curvature $K = C_1C_2$ are used to describe the geometry of a surface and classify its type: planes and cylinders have zero Gaussian curvature ($K = 0$), spheres have positive Gaussian curvature ($K > 0$) and saddles have negative Gaussian curvature ($K < 0$); similar to the plane, saddle shapes can also have zero mean curvature ($H = 0$).

• Energetics of membranes as surfaces: the Helfrich-Hamiltonian $\mathscr{H}$ is a function, or, precisely, a surface based energy functional,[4] which assigns an energy density to every point of the surface in a lipid membrane based on the values of H and K; it results in the membrane shape energy when integrated along the surface. At equilibrium, a lipid membrane adopts a shape that minimizes $\mathscr{H}$. To find such a minimum, variational calculus, the Euler–Lagrange equations or various numerical methods can be applied. This minimization process can also be subject to additional constraints by using Lagrange multipliers. For example, one can also choose to minimise the surface area by introducing membrane tension γ as a constraint.

• Triangulated meshes: a surface can be represented using a discrete collection of vertices, edges and faces referred to as a mesh. The vertices of a mesh are points in 3D space which are connected by edges, which then delimit the faces. In a triangulated mesh, the faces of the mesh are (not-necessarily equilateral) triangles, and each face has an associated perpendicular unit vector which defines the orientation of the corresponding triangle. The relative angle between the normal vectors of adjacent faces is given by their scalar product and it can be used to describe the local curvature of the surface.[5]

The study of biological problems often requires the creation of sophisticated models – ideally ones that are easy to use, modify and extend. Fortunately, technological and scientific progress allows us to perform ever more complex and resource-intensive calculations. Therefore, computer simulations have found a wide range of applications including the study of membrane pores,[27] membrane trafficking,[28] membrane remodelling,[18,29] membrane fission and fusion,[30] membrane deformation in flow[17,31] and membrane organelle shapes.[32]

State-of-the-art modelling of the open problems in membrane biophysics requires a variety of expertise, ranging from cell biology and membrane biophysics to programming. It can be challenging for beginners and non-experts to start developing and applying membrane models due to the multidisciplinary character of the questions, the long-standing tradition of the field, and the wealth of knowledge and models. However, choosing a model should be a conscious decision which is, ideally, not influenced by mere technical hurdles or lack of expertise. Before one chooses an appropriate membrane model it is therefore indispensable to know about the limitations of the different options and common pitfalls. The ambition of this review is to guide the reader in their choice of models when they want to start simulating fluid lipid membranes. This review is specifically targeted to a broad audience of researchers that are interested either in starting with lipid membrane simulations, such as computational physicists, or in getting a better understanding of the computational methods, such as experimental biologists with less experience in computer simulations. We limit the scope of the review to problems that involve mesoscale membrane deformations, which we define for the scope of this review
to act on lengthscales roughly between 20 nm and 10 μm. On that scale coarse-grained modelling approaches are most suitable.

We aim to accomplish three tasks. First, for the sake of pedagogical completeness, we give a concise overview of fluid lipid membrane models and corresponding simulation techniques. Second, we present a conceptual guide on how to develop coarse-grained membrane simulations. Third and most importantly, we provide a hands-on tutorial on how to apply mesoscale membrane simulations to three classical membrane problems: the force required to extrude a membrane tube, the equilibrium radius of a membrane tube, and membrane fluctuations. Our focus is on a comparison of different models and a pedagogical examination of them in terms of what these models can achieve and how resource-intensive they are. A repository with ready-to-run tutorials is provided alongside the review.[33] Whenever possible, we point the reader to other relevant reviews which expand on specific topics and technical details.

2 Overview of membrane models

In order to describe fluid lipid membranes, two main approaches have been developed over the last five decades: membranes are either represented based on continuous surfaces (2.1) or based on discrete building blocks (2.2). In the following sections we briefly outline the two modelling approaches.

2.1 Surface-based models

Surface-based models neglect the thickness of the membrane, and instead describe the membrane as a 2D surface in a 3D space. The surface is represented either by a continuous function (2.1.1) or by a discrete mesh approximating a continuous surface (2.1.2). While analytic continuum models have been successfully used to derive many classical results in membrane biophysics, in general, when the surface cannot be parametrised by any basic geometry such as a (nearly) flat plane, a cylinder, a sphere, or combinations of those, it is often easier to represent the surface by a triangulated mesh to be able to carry out the calculations.

2.1.1 Continuum models. Continuum models for fluid lipid membranes have been the subject of several excellent reviews.[11–13] In this section we only introduce the general idea behind the models and discuss several classical problems of membrane biophysics which we later use as a reference for comparison with our computer simulations.

The fundamental assumption of continuum models is that fluid lipid membranes, which are only 4–5 nm thick but can span over micrometer scales, can be represented by a 2D surface without thickness in a 3D space. Due to the assumed separation of lengthscales (difference between thickness and size), the validity range of such models is unlimited at the upper end (*cf.* Fig. 1). At the lower end, although it is hard to give specific numbers, continuum models are useful down to lengthscales of several multiples of the thickness of the membrane (20 nm).[12] If we are able to express the energy of the membrane, we can find its equilibrium shape by minimising this energy. The membrane shape energy functional $\mathscr{H}$ is based on the observation that lipid membranes are incompressible fluids, *i.e.* it is hard to stretch or compress a membrane within the membrane plane, and due to the high diffusivity of lipids within this plane, the membrane is fluid (*i.e.*, it has a vanishing shear modulus). As a consequence, the only relevant energetic change when deforming a fluid lipid membrane is the out-of-plane bending of the surface, which is usually quantified through membrane curvatures (*cf.* Box I). The membrane shape energy $\mathscr{H}$ is then defined as a function of the mean curvature H and the Gaussian curvature K. The most commonly used membrane Hamiltonian (energy functional) is the so-called Helfrich Hamiltonian (or Helfrich–Canham–Evans Hamiltonian),[94] which expands the energy up to the leading order in H and K. This is the second order for H and first order for K as can be seen from the equation of this Hamiltonian (K carries the units of 1 over length squared similar to H^2)

$$\mathscr{H} = 2\kappa\int(H-H_0)^2\mathrm{d}A + \bar{\kappa}\int K\mathrm{d}A + \gamma\int\mathrm{d}A. \qquad (1)$$

In eqn (1), κ quantifies the membrane's resistance to bending and $\bar{\kappa}$ the resistance to changes in the membrane topology, *e.g.* during membrane fusion or fission. Typically, for fluid membranes the membrane bending rigidity is in the range of $\kappa \in [10, 100]\ k_\mathrm{B}T$, which can be measured by pulling membrane tethers or analysing membrane fluctuations.[9] By contrast, the Gaussian curvature modulus is hard to measure since any attempt requires that the membrane changes its topology. Using continuum arguments and coarse-grained simulations, the Gaussian curvature modulus is expected in the range of[12,73] $\bar{\kappa} \in [-0.5, -1]\ \kappa$. The quantity H_0 is the preferred or spontaneous membrane curvature, and it describes the membrane's tendency to curve spontaneously, due to, *e.g.* an asymmetric lipid composition or proteins. The last term in eqn (1) imposes a constraint on the area of the membrane, where changing the membrane area comes with an energetic cost given by the membrane tension γ. From the physical point of view, γ acts as a Lagrange multiplier or the chemical potential for the membrane area[12,95] and it is of the order of[96] $\gamma \in [1\times10^{-6}\text{–}5\times10^{-4}]$ N m^{-1}. The membrane shape energy can be computed by integrating all terms in eqn (1) over the surface area of the membrane. Eqn (1) is the simplest representation of a membrane shape energy. However, more elaborate membrane energy functionals have been developed in order to represent, for instance, the asymmetry between the two phospholipid leaflets that make up most membranes due to effects such as asymmetric lipid compositions, membrane proteins or preferential protein binding. Such effects can be taken into account by adding a non-local curvature energy, like the area-difference-elasticity.[13] In addition, extensions of eqn (1) have been devolved to represent lipid tilt,[97] lipid twist,[98] and lipid splay-tilt coupling.[99] For the scope of this review, however, we focus on eqn (1) in order to keep the discussion as simple as possible.

The Helfrich Hamiltonian has been successfully used to study several classical problems of membrane biophysics (see first row in Fig. 1). The equilibrium shapes of membrane vesicles can be calculated by finding the membrane configuration that has the smallest energy.[39] In this specific case it means that by minimizing the energy functional under the constraint of enclosed volume, the corresponding Euler–Lagrange equation – typically called the shape equation – predicts the equilibrium membrane shape.[100] In general the shape equation is a non-linear partial differential equation (PDE) that cannot be easily solved. However, by assuming rotational symmetry of the membrane shape it is possible to remove some of the complexity of the problem and simplify the PDE to a system of non-linear ordinary differential equations (ODEs). Such systems of ODEs have been solved numerically to predict the shape diagram of lipid membrane vesicles[39] (*cf.* Fig. 1). Other similar uses of continuum models include the study of vesicle adhesion,[42,44] of shape transitions of vesicles and functionalised vesicles,[40,41] of domain induced budding[46] and of vesicle domains[45] on the micrometer scale (*cf.* Fig. 1).

Aside from predicting equilibrium vesicle shapes, continuum theory can determine the response of membranes in standard experimental assays typically used to characterize the membrane properties – see ref. 9 for a review on testing continuum membrane models experimentally. For example, continuum theory connects the properties of a membrane to its height fluctuation spectrum, relating the amplitude of the fluctuations (h) in the Fourier space to the bending rigidity and the membrane tension as $h^2(q) \propto k_\mathrm{B}T/(\kappa q^4 + \gamma q^2)$, where q is the

SURFACE-BASED MODELS

Type	Representation	Applicable scale range	Typical applications	Available software
Top-Down	Continuum	20 nm ↓ Unlimited	Fluctuation spectra [34] Particle uptake [35,36] Uptake dynamics [37,38]; Vesicle shapes [39] Shape transitions [40,41]; Tether pulling [42,43] Adhering vesicles [42,44]; Vesicle domains [45] Domain-induced budding [46]	Mathematica MATLAB Julia FEniCS ...
	Mesh	20 nm ↓ Unlimited	Particle uptake [47,48]; Membrane-filament interactions [49]; Membrane and active particles [50,51]; Tether pulling [52-54]	Surface Evolver [55] Mem3DG [56] PyMembrane [57] TriMem [5] Flippy [58] ...

PARTICLE-BASED MODELS

Type	Representation	Applicable scale range	Typical applications	Available software
Bottom-Up	*All-atom and CG all-atom* 1 lipid ~ 100 particles	Å ↓ 100 nm	Lipid phase behaviour [59,60]; Studies on lipid types [61-63]; Membrane channels and ion transport [64,65]; Transmembrane proteins [66,67]	CHARMM-GUI [68] GROMACS [69] NAMD [70] ...
Bottom-Up *or* Top-Down	*Several-beads-per-lipid models* 1 lipid ~ 10 particles	10 nm ↓ 10 μm	Self-assembly of structured complexes [71]; Mechanical properties [72,73]; Membrane domain formation [74]; Membrane fusion & pore formation [75]	LAMMPS [90] HOOMD [91] ESPResSO [92] ...
Top-Down	*One-bead-per-lipid-patch* 10 lipids ~ 1 particle	20 nm ↓ 10 μm	Particle wrapping & uptake [24,76-78]; Vesicle shapes [79-82]; Whole cell membranes [83-86]; In-plane mechanics [87-90]	LAMMPS [91] HOOMD [92] ESPResSO [93] ...

Fig. 1 Classification and applications of fluid membrane models. Continuum models have been used to study membrane fluctuation spectra,[34] particle uptake,[35,36] uptake dynamics,[37,38] vesicle shapes,[39] shape transitions,[40,41] tether pulling,[42,43] adhering vesicles,[42,44] vesicle domains,[45] and domain induced budding[46] among others. Examples of available software packages to solve continuum models are Mathematica, MATLAB, Julia or FEniCS. Applications of mesh models are particle uptake,[47,48] membrane–filament interactions,[49] membranes and active particles[50,51] and tether pulling.[52–54] Available mesh model software are the Surface Evolver,[55] Mem3DG,[56] PyMembrane,[57] TriMem[5] or Flippy.[58] Particle based models representing 1 lipid as a collection of 10–100 particles have been applied to study lipid phase behavior,[59,60] studies on lipid types,[61–63] membrane channels and ion transport[64,65] and transmembrane proteins.[66,67] These models can be simulated using software like CHARMM-GUI,[68] GROMACS[69] or NAMD.[70] Simulations with lipids represented as 1–10 particles (which we refer to as several-beads-per-lipid models in Section 2.2.3), and focused on the 10 nm–10 μm scale have been applied to studying the self-assembly of lipids and other biomolecules into structured complexes,[71] mechanical properties,[72,73] membrane domain formation[74] and membrane fusion and pore formation.[75] Simulation models where a patch of membrane is represented as a single particle (one-bead-per-lipid-patch models in Section 2.2.4) have been exploited to probe phenomena at larger scales (20 nm–10 μm) such as particle wrapping and uptake,[24,76–78] vesicle shapes,[79–82] whole cell membranes[83–86] and in-plane mechanics,[87–90] among others. Available software to simulate these types of models are LAMMPS,[91] HOOMD[92] or ESPResSO.[93] Images in the figure reproduced with permission from ref. 36, 39, 44, 45, 47, 49, 51, 53, 60, 61, 65, 67, 71, 72, 74, 75, 76, 80, 83 and 89.

wave number.[101] It also predicts the force $f = 2\pi\sqrt{2\gamma\kappa}$ required to pull a membrane tube and its corresponding equilibrium radius $R_{eq} = \sqrt{\kappa/(2\gamma)}$.[43] In addition, continuum theory has also been used to study particle wrapping by membranes, *e.g.* for predicting the shape of the membrane during particle uptake[35,36] and its dynamics[37,38] (*cf.* Fig. 1), or for determining the regions in phase space in which membrane adhesive particles are completely, partially or not at all wrapped as a function of bending stiffness, adhesion energy and membrane tension.[35] Membrane particle uptake has been reviewed in detail previously.[102]

2.1.2 Mesh models. Analytic solution of continuum membrane models is mostly limited to cases that show rotational symmetry. In more complex systems, as when the membrane is coupled to discrete proteins or cytoskeletal filaments, or when it interacts with extracellular structures, PDEs describing the membrane shape can be obtained, but these are difficult to solve without further assumptions. An alternative approach to a straightforward PDE solution is to determine the membrane shape by generating configurations that gradually minimise the energy described by eqn (1). A particularly powerful approach to carry out such an energy minimisation is *via* the representation of the surface as a triangulated network or mesh of vertices connected by edges (*cf.* Box I). Below we give only a brief introduction to triangulated mesh models for lipid membrane simulation, and we direct the reader to in-depth reviews for further details.[18,105] A hands-on application of triangulated meshes for a fluid lipid membrane simulation is then provided in Section 3.2.

In a triangulated network, bending energy can be accounted for in different ways. To simulate tethered membranes, which are characterized by a fixed network connectivity, bending costs can be implemented by combining bond and dihedral angle potentials in order to keep the area of each triangle in the mesh approximately constant. Nonetheless, to simulate fluid membranes a dynamic discretization of the bending energy, introduced in eqn (1), is needed. For large shape changes of the membrane, the connectivity of the network must be regularly updated in order to achieve their characteristic in-plane fluidity. Such discretization can be approached in different ways. Frequently used schemes are based on the angle formed between normal vectors of adjacent triangles in the mesh (*cf.* Box I),[106,107] although more sophisticated discretisations of the bending energy that include surface-related corrections are also available.[108,109] On top of the bending energy, the length of the edges in the network must be also constrained to prevent the mesh from self-intersecting,[110] and the ratio between the rates of edge connectivity flips and vertex displacements determines the membrane viscosity.[17] The minimal modelling of a triangulated fluid membrane is completed by incorporating area and volume constraints into the simulation.[5,111] An increase in membrane area can be penalized through a discretized version of the linear constraint in eqn (1) and a membrane tension $\gamma > 0$, where the total membrane area is equal to the sum of the area of each triangle in the network. Alternatively, it is also possible to prescribe a desired membrane area A_0, and penalize deviations from such value using a quadratic constraint like

$$E_A = k_A\left(\frac{A - A_0}{A_0}\right)^2, \quad (2)$$

where A is the actual membrane area, and k_A is a constant that enforces the constraint. A similar quadratic constraint can be used to prescribe a desired volume V_0 in simulations of closed surfaces such as lipid vesicles. Constraining the membrane area and enclosed volume opens the door to controlling the surface-to-volume ratio of simulated lipid vesicles.[111]

Membrane shape and dynamics using triangulated membrane models can be explored with a variety of simulation techniques such as Monte Carlo (MC) or Molecular Dynamics (MD) (*cf.* Box II), which can be also combined in Hybrid Monte Carlo (HMC) schemes. In addition, membrane shapes that minimise the surface energy can be also determined using numerical techniques like the Surface Evolver.[55] In pure MC simulations, both the positions of the vertices in the mesh and the connectivity of the mesh are updated through trial moves. If those moves increase the energy of the system, they are rejected with some probability, which can make the sampling of the configuration space slow. HMC simulations combine MC bond-swap moves to update mesh connectivity with MD time evolution to compute the dynamics of the mesh vertices. This speeds up the process of finding configurations that minimize the energy of the membrane, as long as the equations of motion are correctly integrated.[109,112]

Triangulated membrane models have been successfully applied to study membrane wrapping of ellipsoidal[47] and non-spherical particles,[48] tether pulling,[52,113] membrane–filament interactions,[49] self-assembly of colloids on fluid surfaces[114] and

Box II: Sampling configurations using molecular dynamics and Monte Carlo simulations

Molecular dynamics (MD) is a simulation method based on the numerical integration of Newton's equations of motion for a group of particles to obtain their trajectories in real time. These trajectories can then be used to read off thermodynamic observables (characterising the state and properties of the system) such as the temperature and pressure, or dynamic observables (characterising the trajectories) like the diffusivity and time correlations of particles. The time evolution of the equations of motion relies on the calculation of the forces acting on all particles in the system for the given timestep, which depend on the interaction potentials between particles and the specific particle positions. Unlike the MD simulations, the Monte Carlo (MC) simulations do not rely on the propagation of equations of motion but are instead based on efficiently sampling the equilibrium distribution of states that characterises the system. This means that the evolution of the system in the simulation time does not necessarily bear relevance to the physical motion of the system. Although they can be used to obtain dynamical properties in certain limits, they are typically used to generate equilibrium configurations. An excellent introduction to MD and MC simulations can be found in ref. 103. Established software to conduct MD and MC simulations include LAMMPS[91] (see ref. 104 for introductory tutorials), HOOMD[92] or ESPRESSO.[93]

interactions with active colloids[50,51] (*cf.* second row in Fig. 1). While it is possible to simulate meshes with changing topologies,[115,116] the difficulty in measuring the Gaussian curvature modulus makes it challenging to choose meaningful parameters for the simulation.[73,99] Volume control is easy to achieve, since the mesh defines a clear surface, and therefore, the enclosed volume can be simply calculated.

Numerous software packages implementing triangulated membrane models are currently available, such as Surface Evolver,[55] Trimem,[5] Mem3DG,[56] PyMembrane[57] and Flippy.[58]

2.2 Particle-based models

Particle-based models follow a conceptually different approach to that of continuum and surface based models. Instead of describing the membrane as a surface, the constituents of the membrane are represented as particles, and the mechanical properties of lipid membranes are integrated into the system through inter-particle interaction potentials. In these models, a particle or bead can represent an atom, a collection of atoms,[117,118] a section of a lipid molecule,[119] or a section of the membrane surface.[120]

It is important to note that not only the resolution but also the underlying modelling philosophy varies among the different particle-based models. Generally bottom-up models retain chemical specificity of the lipid molecules and associated macromolecules. In contrast, in generic top-down models[119,120] the particle properties and inter-particle interactions capture the emergent physical properties of the membrane, but do not automatically retain chemical details. Therefore, the spirit of the latter models is similar to the surface-based models of the previous section. In what follows we describe the representative cases of each flavour of models.

2.2.1 Bottom-up particle models. The most straightforward class of model to understand in terms of modelling philosophy within this category are all-atom models, where each atom in a lipid molecule is described as a particle. Coarse-grained bottom-up models group individual atoms into single particles, while still retaining chemical specificity. A common approach is to group ~4–10 atoms together into a single particle, as done in *e.g.* Martini models[117,118] and other similarly coarse-grained representations,[15,64,121–124] which can lead to a considerable speed-up compared to all-atom simulations. We refer to these types of models, all-atom and coarse-grained representations of all-atom as chemically specific models from now on. Parameters in these models can be a derived from atomistic simulations, for instance in a systematic manner that preserves statistical mechanical properties of lipid bilayers,[15] but can also include empirical information derived from experiments, formally mixing the bottom-up and top-down approaches. Chemically-specific lipid models have been relevant for instance for the study of lipid phase behaviour,[59] the impact of membrane proteins on the membrane,[66,67] and the interaction of membrane channels and ion pumps with the membrane[64,65] (*cf.* Fig. 1). It is important to note that while such models capture processes at the nanoscale, mesoscale and bulk membrane properties might not be always correctly captured.[125] Details on chemically specific models can be found in ref. 126, an overview of the field in ref. 127, and a pedagogical review of chemically-resolved coarse-grained models can be found ref. 128. Timescales and lengthscales in bottom-up models typically do not capture mesoscale membrane deformations,[15] and are beyond the scope of this review.

2.2.2 Top-down particle models. These models typically contain a much smaller number of particles than those developed *via* bottom-up approaches. In several-beads-per-lipid models (Section 2.2.3) individual lipids are described as a collection of typically 3–7 particles.[129] In a more coarse manner, in one-bead-per-lipid-patch models (Section 2.2.4) a small patch of membrane can be modelled as a single particle.[120] In these models the interactions between particles are not derived bottom-up, but are typically chosen empirically such that mesoscale membrane properties (*e.g.* fluidity, deformability) are correctly captured and match experimentally observed behaviour. Particle-based models have been reviewed extensively before,[29,30,32] and despite partially sacrificing lipid-level resolution they have proved very valuable for exploration of mesoscale membrane deformations. Therefore, they will be the focus of this tutorial review.

2.2.3 Several-beads-per-lipid models. Here we introduce models that represent each lipid as several (usually 3–7) particles. A detailed review on such models can be found in ref. 129. Several-beads-per-lipid models are generic, top-down models that do not address specific detail of the lipids. In these models, a lipid is represented by one hydrophilic head particle connected to a chain of hydrophobic tail beads, mimicking the structure of many common lipids. The main conceptual idea behind these models is to reproduce key emergent membrane properties, such as the self-assembly of single lipids into a fluid membrane as a result of hydrophobic interactions with the solvent, and a bending rigidity of a few to a few tens of k_BT. For this purpose, two main approaches have been developed, which can be broadly classified as explicit or implicit solvent models.

The main player of explicit models is dissipative particle dynamics (DPD).[130–132] DPD-based membrane models explicitly consider solvent-like particles in addition to lipid particles, where DPD solvent particles are coarse-grained representations of a volume of fluid. While explicitly simulating a solvent can incur a large computational cost, DPD models mitigate such cost by using soft potentials to account for the interaction between pairs of particles in the simulation. In contrast to hard potentials, which diverge when particles get close, soft potentials remain finite at vanishing inter-particle distances. This allows for larger time steps during the simulation. However, DPD thermostats can be challenging to handle, which often limits the step-size advantage of DPD.[133] An alternative approach to DPD models is known as the phantom solvent model,[134] in which phantom solvent particles are introduced to generate the hydrophobic interactions between lipids and solvent. To speed-up the simulations, however, solvent particles do not interact with each other, since these interactions are computationally the most numerous and hence the most costly.

The second approach are implicit solvent models. In this case, instead of having explicit solvent particles, special interactions are introduced that effectively mimic the hydrophobic effect. It should be noted that using implicit solvent neglects the momentum transfer of the fluid, which is important, among others, for membranes in shear flow (see Section 2.2.3). The first model of that kind uses multi-body density-dependent attraction[135] and models a lipid as a sphere with three parts. The idea of a multi-body density-dependent attraction was later refined[136] to penalize lipids with few neighbours because these lipids are exposed to the (implicit) solvent. In this model, a lipid consists of three beads. Additional models have been developed, using attractions between lipid beads to mimic hydrophobicity with an implicit solvent.[74,119,137–141] Importantly, some of these models apply Lennard-Jones potentials, but with an extended range, to mimic the hydrophobic interaction.[74,119,137,139] By tuning model parameters like the bond angle stiffness[139] or the lipid–lipid interaction strength,[119] it is possible to control the self-assembly of lipids into lipid bilayers, and to control membrane properties such as the bending stiffness and fluidity. The 3-beads-per-lipid model by Cooke and Deserno[119] is applied to simulate and study membrane fluctuations in Section 3.4.

Several-beads-per-lipid models have enough detail to allow for the simulation of lipids with variable interactions, which has led to studies on lipid domain formation and phase separation.[74,132] As the membrane leaflet thickness is explicitly modelled, these models are also suitable to study the physics of membrane fusion and pore formation.[75,142] Additionally, given their highly coarse-grained nature, several-beads-per-lipid models enable the simulation of membranes on longer timescales ($\sim 10^{-1}$ ms) and larger lengthscales (~ 10 μm) than those available to chemically specific models, making them an attractive choice for biologically measurable events which involve large-scale deformations such as particle uptake.[143] Lastly, the combination of the computational efficiency and lipid details allows several-beads-per-lipid models to be used to interrogate mechanical properties of membranes such as the bending and Gaussian curvature moduli, and bilayer asymmetry.[72,73,144] Nonetheless, for the simulation of larger length- and longer timescales, like experiments with GUVs (giant unilamellar vesicles) or whole cells, more coarse-grained models are required.

2.2.4 One-bead-per-lipid-patch models. To simulate membranes at the largest possible scale with particle-based models, a high level of coarse-graining has been achieved by representing a patch of lipids by just one bead. Several similar one-bead models deploying anisotropic potentials have been developed.[120,145–147] One-bead models are unique in that they have a low computational cost due to extensive coarse-graining and, as the beads are not connected (*i.e.* models are meshless), they can easily undergo topological changes such as the neck fission during endocytosis. These model features have led to simulations over a large membrane area with large deformations, and to modelling phenomena where membranes are cut (*cf.* Fig. 1). For example, one-bead membrane models have been used to understand experimental results of temperature-mediated (large) liposome transport in hydrogels,[86] to study vesicles[24,80,148] and membrane–cytoskeleton interactions,[76] to explore endocytosis and budding phenomena,[24,149,150] and even whole cell membrane reshaping.[83–85,151,152] Among the different available models, the implementation of the YLZ one-bead model[120] in the free, open-source software package LAMMPS[91,151] makes it a particularly attractive choice for MD simulations of highly coarse-grained fluid lipid membranes. This model is applied in detail in Section 3.3 to study membrane tube relaxation.

Membranes can be simulated with one-bead models on lengthscales of ~ 10 μm and over timescales of up to minutes, making them suitable for modelling large biological membrane phenomena. However, these models have a few drawbacks. Although the ease with which one-bead-type membranes undergo scission and produce pores contributes to their success over mesh-based models, their rupture is also not always physical.[153] In fact, *in vivo* or *in vitro* membranes do not typically cut as readily as these models would suggest.[25] Additionally, implementing volume control within enclosed membrane surfaces in these models is not trivial. Attempts to implement volume control using one-bead models include introducing coarse-grained solvent particles inside or outside membranes,[80,151] or introducing a harmonic potential for the target volume which is used to compute the force that acts on a single membrane bead after triangulation of the membrane surface.[154]

Nonetheless, the sensitivity and extent of volume manipulation in these methods is overall limited. Furthermore, as with mesh-based and continuum models, the bilayer thickness is not represented: the volume exclusion of the beads representing the membrane endows it with an artificial thickness, which makes these models unsuitable for looking at phenomena at scales <20 nm. Additionally, there is some discussion whether the kinetics of in-plane diffusivity *versus* out-of-plane mobility has the correct ratios in these models. This means, for example, that caution should be exercised where binding kinetics are being modelled alongside membrane deformations. An attempt to rectify these inaccuracies by including partially correct hydrodynamics without introducing implicit solvent has been previously made.[155]

2.2.5 Top-down models with explicit treatment of solvent hydrodynamics. The role of solvent hydrodynamics, namely, the effect of the dynamics of the surrounding fluid on the membrane, can be thought of in two parts. The first part is the local interactions of the membrane with solvent particles. Without introducing explicit solvent particles, these interactions are usually represented by two additional forces on the lipid beads in simulation: a viscous drag (velocity-dependent) term and a random force (temperature-dependent) term. All several-beads-per-lipid and one-bead-per-lipid-patch models with implicit solvent incorporate these dynamics to account for the local effects of solvent on the membrane, in addition to lipid–lipid interaction potentials designed to mimic the hydrophobic interactions between lipids and solvent (see Section 2.2.1).

Box III: Guiding questions for model development
Below we present a series of questions that the reader might want to ask themselves to systematically develop a coarse-grained computational model that involves mesoscale membrane deformations.

1 Relevant membrane properties
– What is the scale of the deformation?
– Is the explicit bilayer structure needed?
– Is the membrane heterogeneity important?
– Does the process involve a membrane topology change?
– Is membrane dynamics important?
– Should the membrane area or the enclosed volume be constrained?
– Is membrane hydrodynamics relevant?

2 Computational implementation
– How are the external players, *e.g.* proteins represented?
– What is the appropriate membrane geometry (*e.g.* vesicle or flat sheet)?
– What are the boundaries of the simulation box (*e.g.* periodic or closed)?

The second part is related to membrane and solvent interactions arising from the momentum transfer of the fluid, which cannot be handled by the aforementioned local force terms. This can be important when global fluid flows occur near or at the membrane surface resulting in unique membrane shapes and dynamics. Developing sophisticated techniques to couple coarse-grained models with accurate global solvent hydrodynamics has been the focus of multiple studies in the last two decades.[131,151,155–158] Some techniques use coarse-grained explicit solvent particles and combine these with hydrodynamic equations to speed up the integration in simulations, as in DPD and multi-particle collision dynamics (MPC) models.[159,160] Alternatively, other techniques solve the hydrodynamic equations of the solvent and integrate the solutions with membrane surfaces by treating membrane–solvent boundary carefully, as in the immersed boundary method or lattice Boltzmann method.[161,162] The aforementioned methods for considering hydrodynamics are then simulated with the membrane model of choice, which can be a several-beads-per-lipid model, one-bead-per-lipid-patch model or a mesh-based model. These methods have proven crucial to understand the behaviour of vesicles in shear flow[156,158] which is particularly relevant for studying red blood cell shape and dynamics in capillaries.[152,157,163–166]

3 Hands-on tutorial: simulating mesoscopic membrane deformations with three model examples

In the first part of this section we provide the reader with some general advice for the development of mesoscale membrane simulations. We build on this in the hands-on tutorials of Sections 3.2–3.4, where we measure the force required to extrude a membrane tube using a tether with a mesh model, determine the equilibrium radius of a membrane tube as a function of membrane tension using a one-bead-per-lipid-patch model, and extract the fluctuation spectrum of a fluid membrane with a three-bead-per-lipid model.

3.1 General advice for the development of mesoscale membrane simulations

By definition, a model is a simplified representation of reality. Deciding how to develop a model, or which model to use depends on (1) the system of interest, (2) the hypothesis to test, (3) the available computational resources and, potentially, (4) coding experience. Points (3) and (4) are of a more mundane character and we hope to address them in Sections 3.2–3.5. By contrast, points (1) and (2) require more intellectual input from the modeller. In this section we provide a conceptual guide to show the ways in which (1) and (2) can be addressed in a coarse-grained, mesoscale membrane model simulation. A brief summary can be found in Box III, which suggests a series of questions to guide the reader in designing their computational coarse-grained model to study a biophysical problem featuring a fluid lipid membrane.

Given a specific problem, one must first identify which membrane properties need to be accurately represented in the study of the system. These properties depend both on the scale of the whole phenomenon and on the scale of the membrane deformations. Trivially, it can be too computationally costly to simulate large membranes using a fine-grained model (*e.g.* many beads per lipid); mesoscale properties might not have been benchmarked for such models either. Conversely, the membrane will not exhibit the correct behaviour if one tries to interpret fine details of membrane behaviour using a very coarse-grained model (*e.g.* a triangulated mesh or one-bead-per-lipid-patch models). If the membrane thickness is thought to contribute in an important way to the studied process, mesh-based and one-bead-per-lipid-patch models are likely not suitable, as membrane thickness does not have physical interpretation in these models and these approaches do not capture any properties on the scale of a single lipid.

Another important consideration is in the representation of membrane dynamics. Is it sufficient to obtain a static (*e.g.* a free energy minimum) picture of the membrane shape, or does the phenomenon of interest have important dynamical features? Membrane dynamics often comes into play through bending fluctuations and in-plane lipid diffusivity, but it may also be important for some large membrane shape changes. In these cases continuum approaches might be too complicated to solve analytically and therefore should be disregarded. Instead, numerical approaches such as mesh-based or particle-based models should be favoured, which can deal with large membrane shape changes and dynamics, and can be customised to the specificities of the system under investigation. It should be noted that typical mesh-based models combined with MC or HMC schemes can correctly sample equilibrium configurations of fluid membranes. By contrast, lipid diffusion might not be properly captured, because MC schemes do not model dynamics in general. To properly capture

lipid dynamics, one must turn to particle-based MD models of appropriate coarse-graining level.

In processes where membrane fusion or fission plays a role, one may need to consider whether and at what level of detail topological changes are captured. As discussed above, mesh based models do not easily undergo topological changes whereas particle based models readily do, although some of them may not capture correctly the exact point of breaking or the two leaflets of bilayer membranes. Other considerations include the overall state of the membrane, like constraints on the enclosed volume or membrane area. These are straightforward to include in mesh-based models and more challenging to capture in particle-based models. Depending on the problem at hand, one needs to decide whether implicit or explicit solvent is more appropriate. When hydrodynamics are of high importance as, for instance, when studying the deformations of vesicles due to fluid flow or the transport of media through membrane pores, then models which incorporate the full solvent hydrodynamics should be considered. In addition, most membrane remodelling processes in cells involve assemblies of proteins such as protein coats or filaments such as the cytoskeleton that are coupled to the membrane. Such protein–membrane interactions can be included either implicitly through external forces acting on the membrane or through explicitly modelling, *e.g.*, filaments in simulations.

Despite all these considerations, it is important to note that often several distinct approaches can be applied to the same biophysical system. This is highlighted in Fig. 1, and discussed in the following sections, where we describe three simulation tests using three different membrane models.

3.2 Force required to extrude a membrane tube with a triangulated-mesh model

As tube extrusion using a tether is relatively straightforward to perform experimentally, it has been extensively used to mechanically characterize lipid membranes.[9] In this section we describe a simulation protocol to study the formation of a membrane tether using a dynamically triangulated mesh.[54,113] We perform MC simulations, but the protocol we describe is general and can be used with any other triangulated mesh model.[5,56–58] In particular, together with the MC code, we provide the option of conducting tube pulling experiments using TriLMP,[167] an in-house software that couples the TriMEM[5] mesh-based software with the LAMMPS[91] MD simulation package to perform HMC simulations.

3.2.1 Simulation set-up. Membrane tubes extruded by pulling a tether have been shown to follow predictions of the Helfrich analytic theory, namely, a radius of $R_{eq} = \sqrt{\kappa/(2\gamma)}$ and a required force for tether extrusion $f = 2\pi\sqrt{2\kappa\gamma}$.[43] Using typical values for bending rigidity κ and membrane tension γ, we expect tube radii below optical resolution, ranging from tens to hundreds of nanometres. Therefore, the membrane area that moves into the tube is small in comparison to the total area of the cells and vesicles they are experimentally extruded from. For this reason, it should be sufficient to restrict our simulation to a membrane patch rather than simulating a full vesicle, as this is the scale relevant for the deformation.

To conduct the simulation, we initialize the system as a flat triangulated mesh (Fig. 2A), which was generated by arranging a collection of points as a two-dimensional triangular lattice and connecting the nearest-neighbour vertices. The number of vertices N_V must be sufficiently large to avoid finite size effects, and it is governed by the trade-off between simulation resolution and the computational time. Vertices in the mesh will be divided into edge N_E and bulk vertices N_B; the latter are defined by selecting a circular patch on the lattice (Fig. 2A). Edge vertices will remain fixed during the simulation, while bulk vertices can move. In this tutorial $N_V = 5600$ vertices, of which $N_B = 3452$. Triangulated mesh models constrain the length of the edges in the mesh between a maximum l_{max} and minimum l_{min} elongation to prevent the mesh from self-intersecting.[110] While we do not expect mesh self-intersection in the simulations, we set the edge length constraints to typical values in the literature, namely $l_{max}/\sigma = \sqrt{3}$ and $l_{min} = \sigma$, where σ is the diameter of a hard-sphere particle placed on a mesh vertex and is also the simulation unit of length. These edge length constraints also contribute to keeping the regularity of the triangles in the mesh, which is important for the calculation of the bending energy. The choice of σ sets the lengthscale of the system, and the average bond length $\langle l\rangle/\sigma$ at which the system is initialized determines the initial area of the membrane patch. All bonds in the mesh satisfy $l < l_{max}$, except for bonds connecting edge and bulk vertices, for reasons we comment below.

Once the initial conditions of the simulation are set, we must define the membrane energy function. This will dictate how the vertices in the mesh fluctuate and respond to deformations. The energy function used to describe the membrane is based on a discretized version of the Helfrich bending energy[94] which uses the dot product of the face normals (see discussion in Section 2.1.2) together with contributions that penalise area increase of the form γA. We are simulating a flat membrane patch, so no volume constraint applies, and we only need to set the bending rigidity, which we choose as $\kappa = 20k_BT$,[9] where k_BT sets the energy scale of the system, and the membrane tension, which we take as $\gamma\sigma^2 = 1k_BT$.

3.2.2 Sampling membrane configurations. We use MC simulations to sample equilibrium membrane configurations. A typical simulation run consists of a sequence of two stages: bond-swaps and vertex moves, both of which are accepted according to a Metropolis-MC criterion.[110] During the bond-swap stage, mesh vertices remain immobile as we try to sequentially update the connectivity of the network (Fig. 2A). All edges in the network can be swapped throughout the simulation, provided that the new bond $l < l_{max}$.

3.2.3 Boundary conditions and membrane tension. The theoretical expectation for the plateauing force f_0 is derived under the assumption that the membrane is at constant tension,[43] which is justified by the observation of lipid

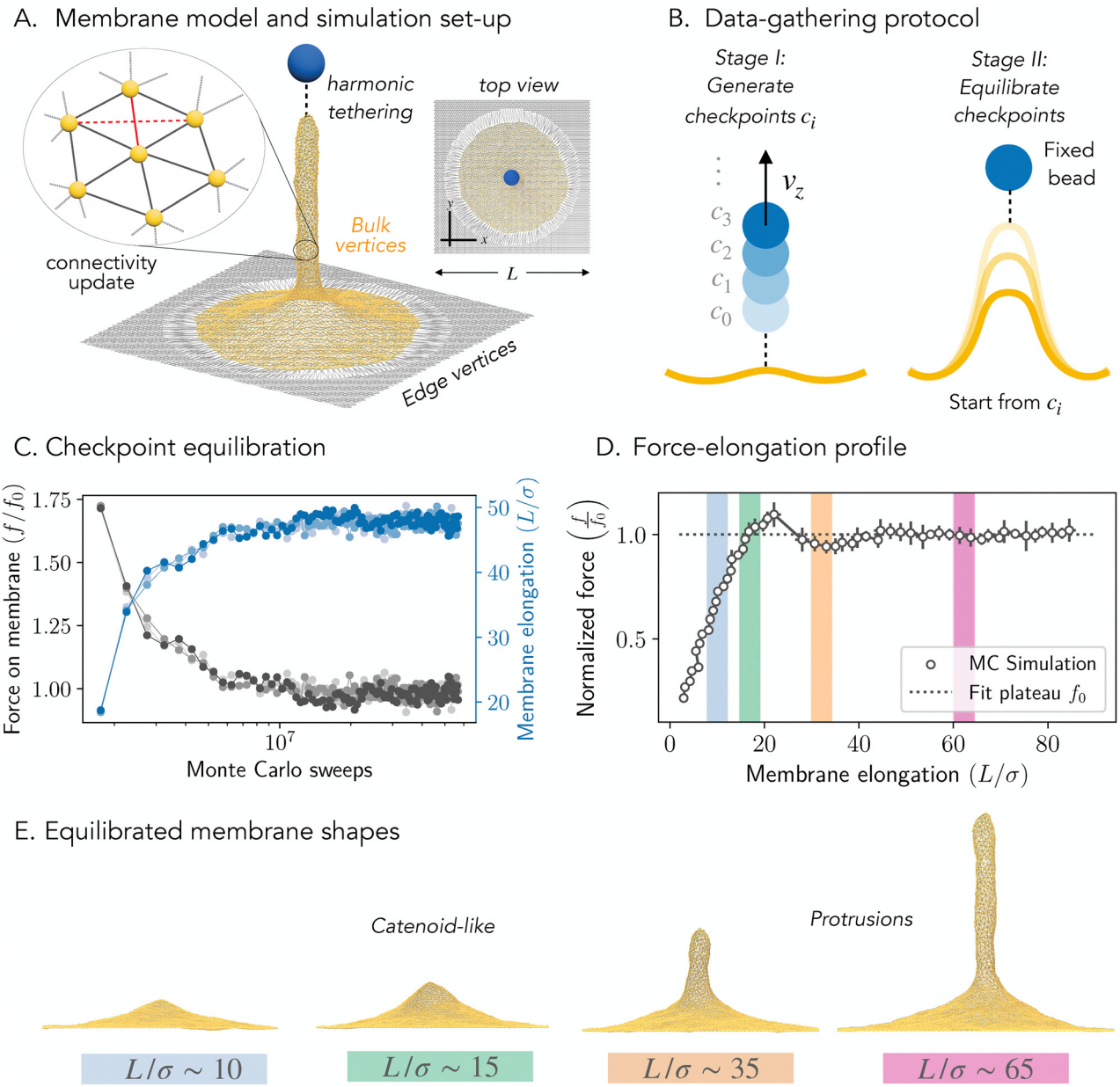


Fig. 2 Tether pulling Monte Carlo simulation using a mesh model. (A) In a dynamically triangulated mesh, membrane fluidity is ensured by regularly swapping bonds in the mesh. To extrude a membrane tube, we set-up a flat membrane patch consisting of edge (fixed) and bulk vertices and tether the central membrane vertex to a bead (blue particle) *via* a harmonic spring. (B) To measure the force required to extrude a tether, we develop a two-stage protocol: first, we move the blue particle to generate simulation checkpoints; second, we reinitialize the system at each checkpoint to let the membrane relax for a fixed position of the blue bead. (C) Example of relaxation curves for the force pulling on the membrane (shades of black) and membrane elongation (shades of blue) obtained during the second protocol step for a membrane with $\kappa = 20k_BT$, $\gamma = 1k_BT/\sigma^2$ and average initial bond length $\langle l \rangle/\sigma = 1.5$. The blue bead is fixed at $z \approx 80\sigma$ from the initial membrane plane. As the simulation progresses, the pulling force on the membrane equilibrates to a non-zero value. (D) Force–elongation profile. The curve is obtained by averaging over the final equilibrated force in three replica simulations. The curve is normalised by the plateauing force f_0 measured in simulations. (E) Representative simulation snapshots for different membrane elongations shown as coloured bars in panel (D).

reservoirs in experimental systems.[168] In simulations, the membrane tension can be kept constant by allowing the projected area of the membrane to fluctuate. This is realised by updating the size of the simulation box according to a Metropolis-MC criterion in combination with periodic boundary conditions.[110] Likewise, it is also possible to keep the membrane tension constant by allowing the number of vertices in the mesh to fluctuate *via* grand canonical MC simulations.[53] Alternatively, as we propose in this tutorial, we can separate the mesh into bulk and edge vertices, connected by bonds that can extend

beyond l_{max}. The membrane is then able to draw area from the 'edge region' at a cost γ, as prescribed in the membrane Hamiltonian. We note that this set-up does not require the implementation of periodic boundary conditions or additional MC moves in the simulation.

3.2.4 Protocol to measure the extrusion force. Mimicking the optical tweezer experiments,[9,54] the desired membrane deformation can be achieved by tethering a bead to the membrane patch and pulling on it (Fig. 2A). In our tutorial, we use a bead that is tethered to the central vertex of the membrane through a harmonic bond. It is important to choose the elastic constant of the bond to be weak, so that one can simulate adiabatic tube extrusion.[113] Here we use $\sigma_B = 10\sigma$ for the diameter of the bead, $k = 1k_BT/\sigma^2$ as the elastic constant of the bond and set the rest length of the bond to $\tilde{\sigma} = 1/2(\sigma + \sigma_B) = 5.5\sigma$.

To study the extrusion force in a simulation, we develop a two-step protocol that allows us to equilibrate the system and obtain the force–elongation profile in an efficient manner (see Fig. 2B). After allowing the bulk vertices of the membrane to relax, we first run a long simulation where the bead tethered to the membrane is displaced along the direction perpendicular to the membrane patch. We do so by updating its position by $\delta x = 10^{-4}$ every MC sweep. As the bead moves, we record simulation checkpoints at a given frequency. We next reinitialize new simulations from all recorded checkpoints. In these new simulations, the bead position is fixed and we allow the membrane to relax. Fig. 2C shows an example of a set of relaxation curves when the pulling bead is fixed at some $z = z_0$ above the initial plane of the membrane. The membrane equilibrates by elongating towards the pulling bead, which minimizes the energy stored in the harmonic bond that connects them. For a given bead position, the force required to deform the membrane is calculated from the deviation of the harmonic bond from its rest length once the system has equilibrated.

3.2.5 Results: analysing the force–elongation profile. The force–elongation profile during tether extrusion of a fluid membrane can be divided into several regimes that have been experimentally measured.[54,113] At small elongations, one finds an elastic regime where the force required to deform the membrane is proportional to the elongation; here the surface adopts the minimum energy shape of a catenoid if the membrane is tensionless.[169] At larger deformations, the profile may exhibit an overshoot,[43,54] followed by a saturation regime where the force plateaus. The plateau, which emerges when a tube with radius $R_{eq} = \sqrt{\kappa/2\gamma}$ forms, indicates the presence of a membrane reservoir that buffers changes in membrane tension. The force will further increase if the reservoir is depleted. A similar observation can be made when tubes are pulled from cells.[168]

The force–elongation profile obtained in our simulations is shown in Fig. 2D. This is computed by extracting the membrane equilibrium elongation value, which we take as the position of the membrane vertex that the pulling bead is bonded to, and the equilibrium force that the membrane exerts on the bead, for different simulation checkpoints. Representative equilibrated membrane shapes as a function of membrane elongation L/σ are shown in Fig. 2E. As described above, the force–elongation profile has first an elastic regime, followed by a plateau where the tube forms and the force required to extrude the tube is independent of the membrane elongation (Fig. 2D). We note that the results in Fig. 2D do not exhibit the plateauing at the expected theoretical value of f, which suggests that while our simulation set-up keeps membrane tension constant, there is a cumulative membrane tension that results from the imposed boundary conditions. In particular, the plateauing force we measure in simulations corresponds to $f_0 = 32.8 \pm 0.4k_BT/\sigma$, while the theoretical result is $f \approx 40k_BT/\sigma$ for $\kappa = 20k_BT$ and membrane tension $\gamma\sigma^2 = 1k_BT$. The discrepancy is expected as it is not straightforward to map the emergent membrane tension γ that appears in the theoretical expression for the force $f = 2\pi\sqrt{2\kappa\gamma}$ to the prefactor that penalizes area changes in the MC Hamiltonian.[170,171]

3.3 Membrane tube equilibration using the YLZ potential

In this section, we simulate membrane tubes in equilibrium of varying bending rigidity at different tensions using the YLZ model,[120] and compare the simulation results to the analytic theory of Section 2.1.1. We note that while membrane tube equilibration could be studied using the same procedure as described in Section 3.2 for a mesh model, here we detail an alternative approach that exploits periodic boundary conditions and a barostat to control in-plane membrane tension.

3.3.1 The YLZ potential. One of the most widely used one-bead-per-lipid-patch models is the YLZ model; developed in 2010 and named after its authors.[120] This set-up models a biological membrane as a single layer of particles interacting *via* a 2-body potential dependent on the relative distance and orientation of the beads (Fig. 3A). When this potential is implemented in MD simulations, beads self-assemble into sheets, tubes or vesicles, and exhibit membrane properties such as biologically relevant fluidity and bending rigidity, which can be tweaked *via* the bead-bead potential parameters. In the simulations presented in this section, we used the values of YLZ parameters given in Table 1, unless stated otherwise. How these parameters relate to the underlying potential can be seen in the original article introducing the potential[120] or the LAMMPS documentation section pair_style ylz.

3.3.2 Simulation set-up and protocol to measure the equilibrium radius. To set up the initial state, we put YLZ particles on a regular trigonal lattice on the surface of a cylinder with the edge length roughly corresponding to the minimum of the radial potential ($\approx 1\sigma$, where σ sets the lengthscale of the system). The orientation vectors are all taken to point outside of the tube, normal to the surface of the cylinder. The simulation box with length L_z, width L_y and height L_x is chosen such that the tube ends join onto themselves *via* the periodic boundary conditions. The other two dimensions of the box are irrelevant, as long as the tube does not interact with images of itself at any point in the simulation (Fig. 3A).

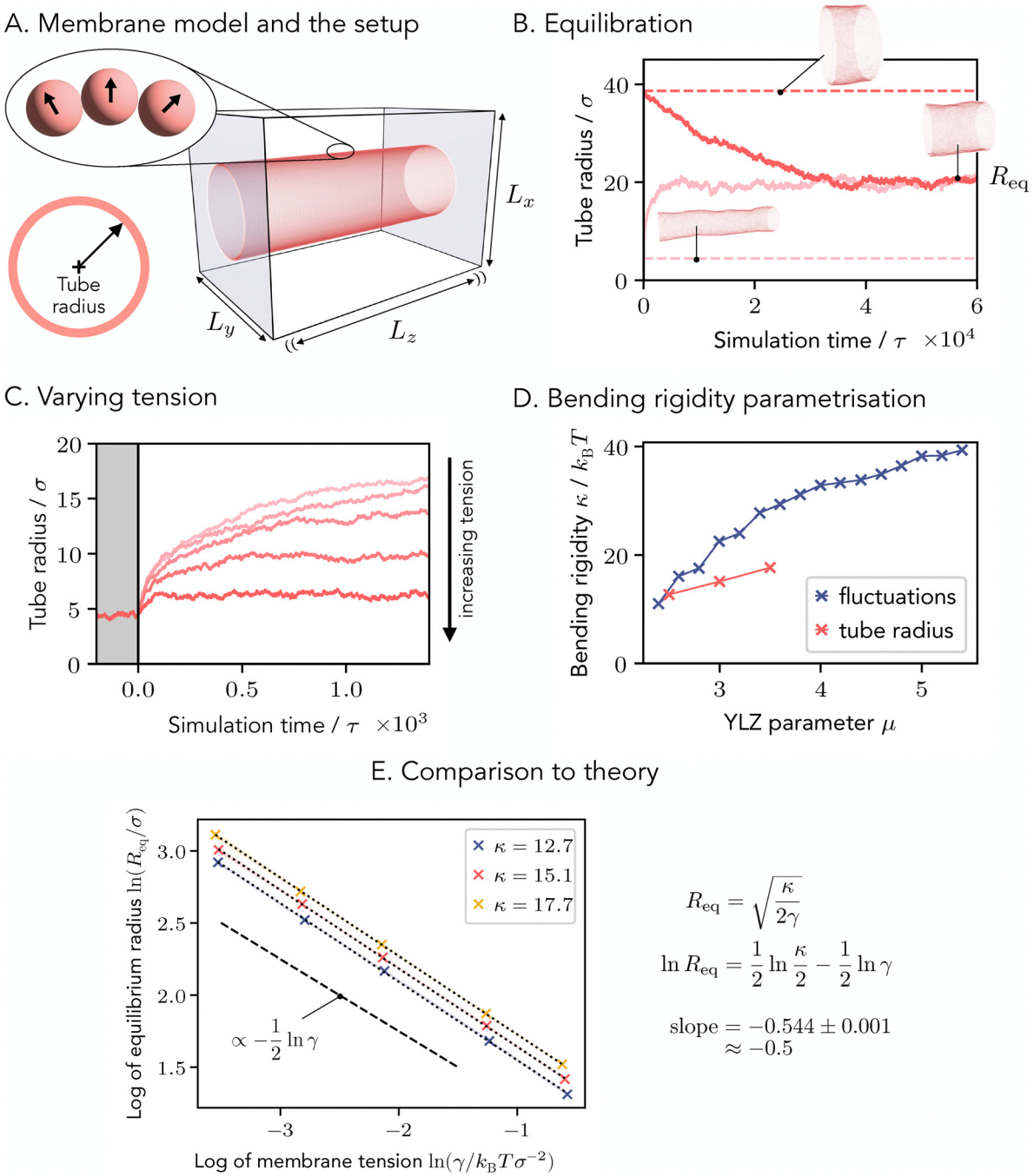


Fig. 3 Membrane tube equilibration using the YLZ potential. (A) The YLZ model models the membrane as a single layer of interacting beads. A tube of an arbitrary radius is constructed out of a regular trigonal lattice in a simulation box with periodic boundary conditions with ((↔)) showing the barostat-coupled z box dimension L_z, which can vary. (B) The size of the box along the cylinder axis is allowed to vary using a modified NPH Nose–Hoover barostat. Regardless of the initial state, under the action of the barostat, the membrane always equilibrates to the same radius if we set the tension and membrane parameters to be the same. (C) As predicted by the theory, the same membrane tube equilibrates to a different radius under different tensions with radius decreasing with increasing tension. (The shaded part shows a fixed-box equilibration.) (D) The parameter μ of the YLZ model changes the bending rigidity of the membrane, κ. The value of κ was estimated for three different values of μ using the tube radii at different tensions (red). This is compared to the estimates from the membrane fluctuation spectrum[120] (blue). (E) The Helfrich theory predicts that the radius of a thermodynamically stable tube is proportional to $1/\sqrt{\gamma}$. The simulations adhere to a power law nearly perfectly, with the exponent being close to that given by the theory. The equilibrium value of the radius for each set of parameters was obtained by averaging over time and three different random seeds. The three seeds were also used for an estimation of the error, which, however, is negligible (see the barely visible shaded area). All quantities in the figure are expressed in simulation units.

To measure the equilibrium tube radius, we use the following protocol. The tube is first equilibrated to a constant temperature at a constant volume using the standard velocity Verlet algorithm in combination with the Langevin thermostat

Table 1 Parameters used for the YLZ potential. All values are in simulation units

Parameter	Value
ε	4.34
μ	2.5, 3, 3.5
ζ	4
$r_{\min}$	$\sqrt[6]{2}$
r_c	2.6
θ_0	0

as implemented in LAMMPS. The tube then relaxes and adopts an average radius corresponding to a specific bending rigidity and membrane tension as given by the theory $R_{eq} = \sqrt{\kappa/(2\gamma)}$. The radius of the equilibrated tube is then measured by fitting a circle to its cross section (Fig. 3B). In the next phase of the simulation, the membrane tube is equilibrated under the action of a Nose–Hoover barostat as implemented in LAMMPS, giving rise to different tension values. The equilibrium radius decreases with increasing membrane tension (Fig. 3C). Measurement of the stress tensor can provide the value of the membrane tension as shown in ref. 172. The LAMMPS code has been modified to redefine the pressure estimator, as outlined in the next section.

3.3.3 Regulating membrane tension in a simulation. The main conceptually difficult part of the simulation is the use of a barostat to control the tension of the membrane. As can be derived from the definition of a surface tension, the membrane tension of a tube can be related to its equilibrium radius and the force in the direction of the tube axis exerted by the membrane as

$$\gamma = \frac{f}{4\pi R_{eq}}. \tag{3}$$

The same formula can also be obtained from the Helfrich Hamiltonian, as done in ref. 172. From the definition of the pressure and stress tensors (in the convention used by LAMMPS), the same force can be obtained as

$$f = -P_{zz}L_xL_y = \frac{\sigma_{zz}}{L_z}, \tag{4}$$

where P_{ij} is the pressure tensor, σ_{ij} is the stress tensor and i, j refer to the three dimensions x, y, z. To control membrane tension with a barostat, we defined a quantity P' that can be used by the barostat and that is easily relatable to the membrane tension. This quantity is defined as

$$P'_{zz} = \frac{P_{zz}V}{N} = -\frac{\sigma_{zz}}{N}, \tag{5}$$

where $V = L_xL_yL_z$ is the box volume and N is the number of membrane particles. This is basically the negative of the average stress per YLZ particle. We can relate P' to γ as

$$\gamma = -\frac{P'_{zz}N}{4\pi R_{eq}L_z} = -\frac{P'_{zz}}{2a}, \tag{6}$$

where we used the area of the membrane cylinder as $A = Na = 2\pi RL_z$, with a being the membrane area per YLZ particle. This allows us to calculate the value of the membrane tension for our simulations. In addition, if we assume that a is approximately independent on the tension, then $\gamma \propto -P'_{zz}$. Empirically, the membrane area per YLZ particle has not varied over our simulations more than ±5%. However, since the area per YLZ particle can be also obtained from the simulation, one does not have to rely on it being constant for the calculation of tension.

3.3.4 Results: comparison with theory and computation of the bending rigidity κ. The membrane bending rigidity κ is controlled by the parameter μ in the YLZ model. This dependence was measured previously by analysing membrane fluctuations[120] (Fig. 3D, blue). In the following, we use μ to change the membrane rigidity. Informed by the analytical formula for $R_{eq}(\gamma)$, we plot $\ln R_{eq}$ as a function of $\ln\gamma$ and observe that these two quantities adhere extremely well to a power law, which can be represented as

$$R_{eq} = \left(\frac{\kappa}{2\gamma}\right)^x. \tag{7}$$

The equation fitted to the data in Fig. 3E is

$$\ln R_{eq} = x\ln\frac{\kappa}{2} - x\ln\gamma. \tag{8}$$

The simulations are consistent in the value of x and only slightly deviate from the theoretical value of 1/2 (Fig. 3E). We varied μ and calculated the bending rigidity κ for each value using eqn (8) (Fig. 3E). We plot the resulting values for κ together with the values obtained in ref. 120 (Fig. 3D, red). The results are noticeably different, which highlights that the YLZ potential is not a mere numerical solution of the Helfrich Hamiltonian (eqn (1)), but rather a coarse-grained potential mimicking the membrane properties in a less straightforward way. Taken together, the results of this set of simulations are internally consistent and agree with the theoretical expectations, but comparison to other types of simulations shows some of the limitations of the model.

3.4 Analysing membrane fluctuation spectrum using the Cooke model

A lipid membrane is thin and soft enough for the thermal energy to manifest as shape fluctuations. In this section, we describe a simulation setup to measure height fluctuation spectra using the coarse-grained Cooke[119] model for lipid membranes and implemented in LAMMPS.

For a flat, tensionless square membrane patch of side of length L and on the xy plane, shape fluctuations result in height fluctuations in the z direction. For small deformations, one can use the small deformation approximation known as the Monge Gauge, where the surface of the membrane is defined by a varying height field $h(x, y)$ with respect to a flat plane and model the ensemble spectrum of $h(x, y)$. As introduced in Section 2.1.1, the mean square amplitude in the Fourier space of the mode with number n follows

$$\langle h_q^2\rangle = \frac{k_BT}{\kappa L^2q^4}, \tag{9}$$

where $q = 2\pi n/L$ is the wave number. Eqn (9) shows that shape

fluctuations provide a connection between the scale of the thermal deformations and the bending modulus κ.

3.4.1 The three-beads-per-lipid model. One of the most recent and widely used several-beads-per-lipid models is the Cooke membrane model,[119] where the solvent is modelled implicitly. We refer the reader to its seminal paper[119] for specifics, and give only a brief overview here.

In the model, each lipid consists of a three-bead-long chain, a head bead b_1 followed by two tail beads, b_2, b_3 (see Fig. 4A). The bonds are finite extensible nonlinear elastic (FENE) bonds, *i.e.* they keep consecutive beads at near fixed distance, and the lipids are kept straight by a harmonic angle potential on the angle formed by $\angle b_1 b_2 b_3$.[119] All beads repel each other *via* a purely repulsive Lennard-Jones potential that becomes zero when particles are 1σ apart, where σ is our distance unit; using typical area per lipid values this means 1σ maps roughly to 1 nm. To represent the solvent implicitly, the hydrophobic interaction between tails is modelled as an added attractive potential with a tunable range w between tail beads of different lipids. The tail-to-tail attractive interaction compresses the tail region, and would lead to spontaneous curvature without further modifications, which for a minimal system introduces extra terms in the Hamiltonian and is thus undesirable. To compensate for this effect and prevent the lipids bound in a membrane from inducing spontaneous curvature we follow the approach of Cooke and Deserno.[119] We set the pair interaction between a head bead and any other bead to a rescaled version of the repulsive part of the interaction potential between tail beads. Effectively, this makes the head beads smaller than tail beads by a factor of 0.95. In equation form:

$$U_{\text{head to any}}(r) = U_{\text{tail to tail, repulsive part}}\left(\frac{r}{0.95}\right). \tag{10}$$

Consistently, we draw lipid head beads with a smaller radius.[73] As a consequence, lipids self-assemble into bilayer membranes. The Cooke model is typically limited to lengthscales of $\sim$μm and timescales of seconds.

3.4.2 Simulation set-up. The seminal paper for the Cooke model[119] tuned two parameters, the tail interaction range w which we set to 1.6σ, and the potential depth of the hydrophobic interactions ε which is also our unit of energy that we set to $k_B T/\varepsilon = 1.1$. Picking these parameters guarantees that our system will be in the liquid phase and allows comparison with other published works, which have used the same parameters.

Cooke and Deserno determined that to match in-membrane lipid diffusion to real lipid diffusion one must set the time unit $\tau \approx 10$ ns, however this makes lipid flip-flopping rates become orders of magnitude higher than those of real lipids. We refer readers to the model's paper[119] for possible modifications if interested in accurate dynamics; here we only require measuring ensemble averages and thus faster flip-flopping dynamics are beneficial since they speed up equilibration. To simulate this system, we pre-assemble a flat membrane in a periodic box with dimensions (L, L, L_z), where we initially pick $L = 60\sigma$. L_z is fixed to 120σ, large enough to avoid self-intersection due to membrane fluctuations. We chose to place each lipid molecule in a hexagonal grid with two layers, one per leaflet, oriented so that their head beads point away from the membrane. We then setup a Noose–Hover barostat, with the relaxation constant of 10τ, whose function is to scale the simulation box length $L = L_x = L_y$ to enforce zero lateral pressure $P_x = P_y = 0$. This is so that the membrane is neither stretched nor compressed, or equivalently, so that membrane tension is kept zero; while this is only partially accurate, since the membrane will fluctuate and thus it will not exactly be aligned with the simulation box horizontal xy plane, in practice as we will show a tensionless theory of fluctuations suffices to explain and fit well our measurements (Fig. 4). Nevertheless, to ensure negligible tension we use half the original timestep, setting it to $dt = 0.005\tau$. To keep the system thermalised at constant temperature, we also setup a Langevin thermostat with the relaxation constant damp = 1τ. We picked the duration of our simulation to fully equilibrate the measurements of average squared amplitude for each fluctuation mode. In practice, we reserved $10 \times 10^3\tau$ for initial equilibration, followed by $110 \times 10^3\tau$ for measurements, for the total of $120 \times 10^3\tau$. Additionally, to be able to distinguish what might seem like an equilibrated observable from one that is kinetically trapped or varying too slowly to be observed, we ran four replicas of the simulation with different seeds of the random number generator used by the thermostat to draw velocities. We setup simulation output so that every 100τ we record the average potential energy, the global temperature and pressure for each box axis, the simulation box dimensions, and the particle coordinates.

3.4.3 Analysis pipeline. We generally follow the analysis provided in ref. 119, informed by a latter work which focuses on fluctuation spectrum analysis.[173] Fig. 4B provides an overview of our analysis. First, we determine if the equilibration stage is complete by plotting together the 4 replicas. We verify qualitatively that the simulation has equilibrated within the initial $10^4\tau$ of the simulation by monitoring the box length L rather than temperature or energy of the system, both of which equilibrate much faster. By visually rendering the simulation, we verify the integrity of the membrane for the rest of the trajectory since lipids can occasionally evaporate and be adsorbed by the membrane. For each trajectory frame, we then apply three transformations. First, we cluster the simulation particles, grouping together particles that were within 1.5σ of each other, and then take the largest cluster as the membrane and exclude the rest, thus removing the gas phase. Secondly, we compute the membrane height field h_{ij} by computing the 2D histogram of the particles z coordinates with bins covering the xy section of the simulation box, and then taking the average height $h_{ij} = \langle z \rangle_{ij}$ in each bin. Here we pick the number of bins per axis to be both a power of two (for a faster Fourier transform), and such that each bin square at least covers $3^2\sigma^2$, which for $L = 60\sigma$ yields 16 bins per axis. Finally, we take the 2D discrete real Fourier transform of h_{ij}, obtaining H_{ij}. We note that while L is not fixed during our simulation, the theory of height fluctuation spectrums expects a fixed L. Nonetheless, because the changes in L are small, we simply use the average $\langle L \rangle$ when fitting the spectrum. For each mode, identified by its 2D wave vector $\vec{n} = (i, j)$, we can then compute the average and standard deviation of

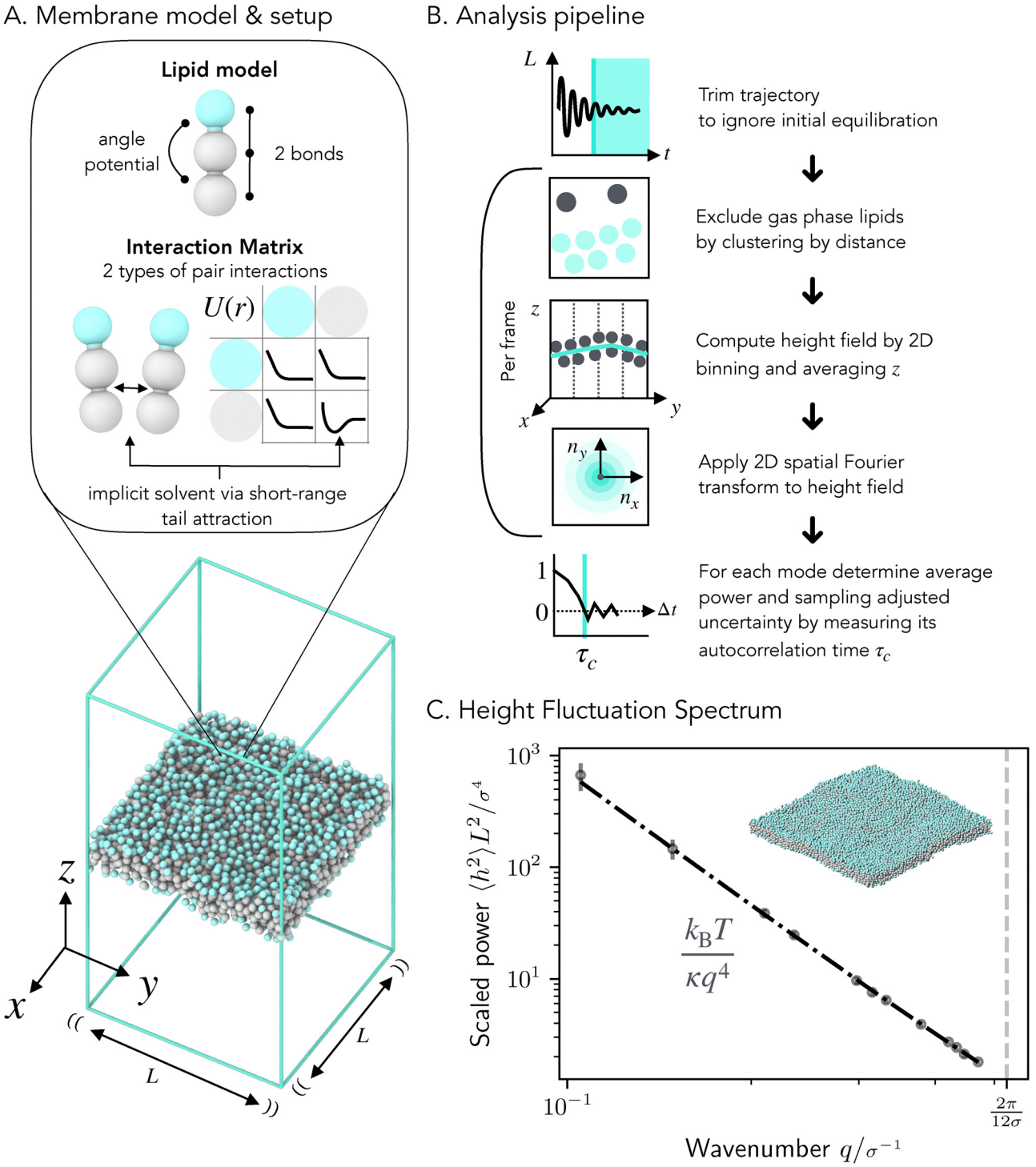


Fig. 4 Membrane fluctuation simulation using the Cooke and Deserno model. (A) (top) Diagram of Cooke bilayer lipid and interaction matrix showing pair potential sketches. (bottom) Simulation setup for a small membrane patch (small size chosen for clarity), with ((↔)) showing the barostat-coupled x, y box dimensions $L_x = L_y = L$, which can vary. (B) Analysis pipeline: skip over initial equilibration determined by the evolution of the box length L in time, then for the rest of the trajectory consider L to be $\langle L \rangle$; process equally spaced frames to obtain for each the instantaneous height fluctuation spectrum h_{ij}; finally for each mode of the spectrum compute its average power $\langle h_q{}^2 \rangle$ in Fourier space and corresponding uncertainty as the adjusted standard deviation $\sqrt{\mathrm{Var}(h_q{}^2)(2\tau_c/\tau + 1)/N}$. (C) Ensemble average of the height fluctuation power spectrum, in log–log space, with snapshot of simulation. The spectrum is linear in these coordinates (fit as dashed black line), up to the cut-off shown as a vertical line at the 12σ wavelength.

the squared amplitude, $\langle |h_q{}^2| \rangle$ and $\mathrm{Var}(|h_q{}^2|)$. Because we are interested in estimating the error of the mean value, we then analyse the time series of the complex amplitude to determine its correlation time. To do this, we compute the statistical inefficiency

g_{ineff} (see ref. 174) for its norm squared $|h_q|^2$ and phase, taking the largest of the two as the effective g_{ineff}. If N is the number of points in the time series, the effective sample side, *i.e.* the number of uncorrelated observations, is N/g_{ineff}. The error of the mean is then $\text{Var}(|h_q{}^2|)/g_{\text{ineff}}$.

3.4.4 Results: analysing membrane fluctuations. In Fig. 4C, we show the membrane height fluctuation spectrum. While it is possible to try to pick the maximum q threshold based on the fitness of the fit, here we took a simpler approach of setting it so the equivalent wavelength is 12σ, or roughly twice the membrane thickness.

By fitting the spectrum in the initial linear region to eqn (9), we can obtain the bending modulus $\kappa = 14.27 \pm 0.3\ k_BT$, with a goodness of fit $Q > 0.009$ (defined in ref. 173). We can compare this result to the original paper, which yields a range between $13–20k_BT$. A more recent work that simulates membrane cylinders (equivalent to tethers obtained by tube pulling)[172] report a value of $\kappa = 11.7 \pm 0.2\ k_BT$ and $12.5 \pm 1\ k_BT$ from the height fluctuation method. We attribute this last mismatch to differences in the fitting expression used for the spectrum.

We note that the key bottleneck in this measurement is the sampling of the mode with smallest wave number of interest because the relaxation time of a mode scales as q^{-4}.[119,175]

3.5 Model comparison: on computational time

The choice of a specific membrane model can ultimately be influenced by the computational resources available. As with any MD and MC simulation, computational time generally increases with the number of elements included in the model: for example, in an MD simulation, the more particles the system contains, the more equations of motion must be integrated, and thus the more expensive it becomes. Both YLZ and Cooke models run on code that is fully parallelised, which aids in reducing the total physical simulation time. As the code used to perform the MC simulations in Section 3.2 is not parallelised, we use the TriLMP software[167] as an alternative representative for the mesh models in this section. TriLMP, which relies on the software TriMem[5] coupled to LAMMPS, is partially parallelised as it updates the network connectivity in a serial manner using LAMMPS functions.

Fig. 5A shows the different CPU timescales required to simulate and study typical membrane deformation phenomena such as fluctuations, endocytosis, tube extrusion or cell division. While the figure suggests a model like YLZ[120] to be optimal for simulating membrane phenomena from the computational-time point of view, this is accompanied by a loss of resolution that the model entails compared to the Cooke model.[119] Likewise, it is important to know which parts of the computational model can be parallelised, and if so, how efficient this parallelisation is to establish a meaningful comparison between models. The parallelisation efficiency is defined as $T_1/(T_n\ n)$, where T_1 is the run time in a single CPU core and T_n is the runtime on n CPU cores. In Fig. 5B we compare the parallelisation efficiency of TriLMP, YLZ and Cooke models as a function of the number of CPUs.

4 Discussion, perspective and open questions

From cells to organelles to vesicles, the role of membranes as shielding and structuring barriers, and as a platform for protein encounters and assembly, is ubiquitous in biological systems. Therefore, membrane remodelling is fundamental for many cellular processes such as cell division, cell migration and endocytosis.[3] Despite decades of remarkable progress in understanding these processes, numerous unsolved problems remain in the field of membrane biophysics. For example, understanding membrane bound protein self-organization and self-assembly processes, along with coupling these to non-equilibrium driving forces such as ATP/GTP hydrolysis, is the focus of much of today's biophysical research and reconstituted biology.[176–179] By complementing novel experimental research in cell and membrane biophysics, computational membrane models have become a highly valuable tool to elucidate the key mechanisms at play in these systems. However, entering the field of computational membrane physics can be challenging due to the multidisciplinary character of the subject. This pedagogical tutorial aims to lower the entry barrier by combining a state-of-the-art overview of the field with practical guidance to advise on how to pick and apply membrane models at the mesoscale.

Throughout the review, we have emphasized how mesoscale membrane models are built on different representations (surface *vs.* particle-based), cover different scales (from single lipids up to whole cell membranes) and come with specific (dis)advantages. We have shown that the choice of an appropriate membrane model depends significantly on the biophysical system under investigation. Indeed, by providing a hands-on tutorial for three distinct membrane models at the mesoscale, and comparing the three representations using criteria such as performance, adaptability and ability to represent the biophysical regime, we have demonstrated that choosing the 'best' model is often highly context-dependent. Nevertheless, one of the key points of our review is the notion that the process of finding a suitable model for a problem of interest can be systematised. Likewise, the breadth of models does not mean the reader must choose and stick to a single one for their project: we argue in favour of testing various membrane models under the same conditions to ensure that the results are independent of the used modelling techniques.

It is essential to know the limitations of the different models, and how to approach questions that lie precisely in the spaces which are difficult to model, as these are the spaces where new and exciting research takes place. Focusing on the limitations of a model sheds light on the gaps in knowledge in the field and can bring clarity to future perspectives and challenges. As one of the future research directions, structural biologists and atomistic MD modellers have argued that the ultimate model for the membrane and for the full biological cell is the so-called digital twin.[180] Such a model is very far from possible,[181] as we do not know the exact composition of every (or arguably any) cell or cellular membrane, and their

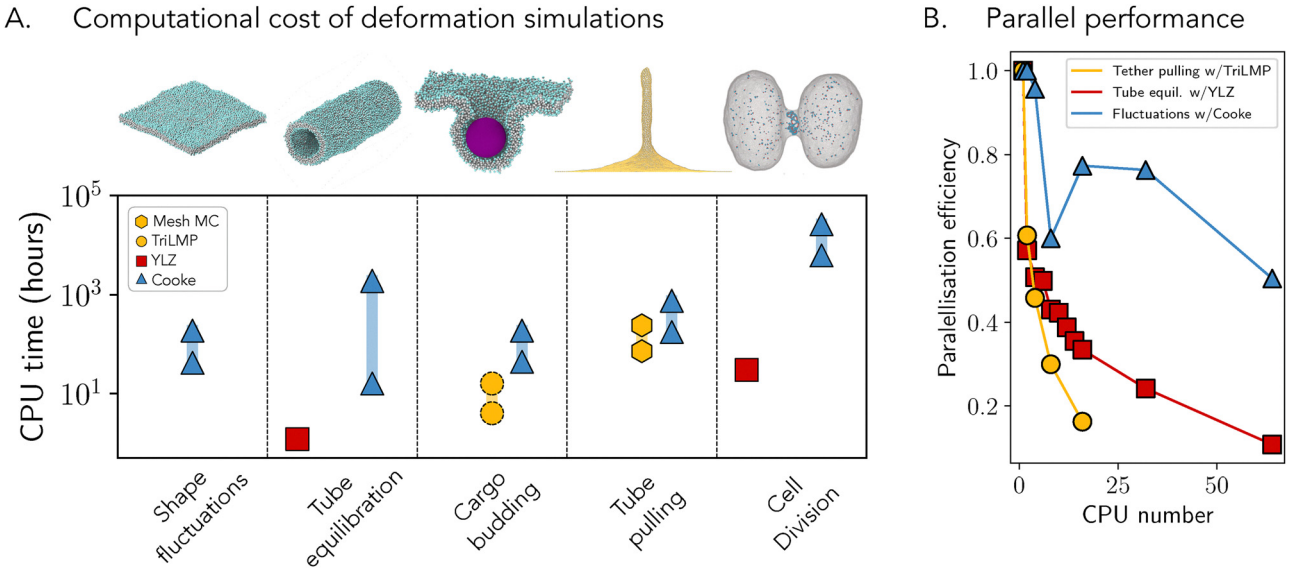


Fig. 5 Model comparison by computational time. (A) CPU time (hours) spent in simulating different membrane deformations using various membrane models. Mesh simulations: cargo budding simulations with a mesh model were performed with the parallelised software TriLMP,[167] using two vesicles with $N_1 = 2562$ (with diameter $\sigma_1/\sigma \approx 30$) and $N_2 = 10\,242$ ($\sigma_2/\sigma \approx 60$) vertices in the mesh; two cargo particles were tested with diameters $\sigma_{c,1}/\sigma_1 = 6$ and $\sigma_{c,2}/\sigma_2 = 6$, and the strength of the interaction was chosen to ensure sufficiently strong adhesion ($\varepsilon/k_BT = 10$). Dashed lines around symbols indicate that the model cannot accommodate the topological changes required to complete the deformation. The tube pulling results for the mesh MC model correspond to the time required to equilibrate the membrane and obtain the curves presented in Section 3.2. YLZ simulations: the CPU time for the tube equilibration was that for the simulations presented in Section 3.3. The time for the cell division was for a vesicle of 50 000 YLZ particles following a reliable division protocol requiring ~800 000 time steps. Cooke simulations: the CPU time ranges were obtained from simulation datasets appropriate for each measurement of shape fluctuations, tube equilibration and cargo budding; for this latter case we included the CPU time required for the equilibration of the final state. For tube pulling and cell division, we used estimates based on scaling the budding simulation by a factor of respectively 4 and 140, the latter corresponding to a cell of area 1 μm^2. (B) Comparison of the performance of three openly available, parallelised membrane models. The drastic fall in efficiency for the mesh model TriLMP is due to the partially serial nature of the code: bond update moves, although computed in parallel,[5] are currently serially implemented in TriLMP. Parallelisation in LAMMPS depends on how the simulation box is subdivided. The YLZ tube simulation was parallelised only by dividing the box along one axis, hence the monotonic curve. The Cooke fluctuation simulation was divided along two axes, which explains the dip at 8 CPUs.

dependence on cell cycle state and cellular environment, limiting building of such models even if they would be computationally possible. Furthermore, it is also unclear how much such a high resolution modelling alone can improve our understanding of the (emergent) mechanisms at play in biological systems.

In our view, the intellectual value of coarse-grained mesoscale membrane models goes beyond their computational feasibility – rather than being a substitute for computationally unfeasible models, they represent an independent approach whose goal is to reveal key underlying physical principles, which are generalisable and go beyond chemical specificity. Precisely because top-down models deliberately simplify the processes under consideration, they enable us to build a physical intuition for the system, and to uncouple driving forces from one another. Hence, coarse-grained mesoscale membrane models constitute a powerful, highly interpretable, readily available and relatively cheap tool that enables the study of biological matter at multiple scales, and that can be easily interfaced with the increasing amounts of experimental biological data and augmented by machine learning.[182,183]

We predict that in the upcoming years the field of membrane biophysics will become even more multidisciplinary, more frequently coupled to experiments carried out in living cells and synthetic cell-like systems, often characterised by dynamical membrane heterogeneity and asymmetry, and inherently driven out-of-equilibrium. The constant back and forth between experimenters and modellers/simulators will be a norm, in particular due to the ever growing accessibility of quantitative cell biology experiments, controlled synthetic cell systems, and high resolution imaging techniques.[184,185] Models will be needed to thoroughly map out and expand the experimentally accessible behaviour, as well as to predict and test physical mechanisms behind complex cell and cell-like phenomena in order to guide future experiments. Likewise, the expanding number of different modelling techniques will also trigger the need for more consistent mapping between the different models. It is likely that models will incorporate various techniques at once, for instance by simultaneously combining several modelling approaches,[15] or including methods of artificial intelligence. We hope that this guide will serve as a valuable reference point for both experimentalist and modellers in this exciting research space, so that models can be more easily learnt, compared and combined in the future.

Author contributions

All authors wrote and approved the manuscript.

Conflicts of interest

There are no conflicts to declare.

Data availability

Tutorial codes to reproduce the results in Section 3 are given as a repository.[33]

Acknowledgements

We thank Oded Farago, Angelo Cacciuto, Jeriann Beiter and Pietro Sillano for helpful discussions and a critical reading of the manuscript. MMB and AP acknowledge funding by the European Unions Horizon 2020 research and innovation programme under Marie Skłodowska-Curie Grant Agreement No. 101034413. FF acknowledges financial support by the NOMIS foundation. BM and AŠ acknowledge funding by ERC Starting Grant "NEPA" 802960. MA and AŠ acknowledge funding by the Volkswagen Foundation Grant Az 96727.

Notes and references

1 B. Alberts, A. Johnson, J. Lewis, D. Morgan, M. Raff, K. Roberts and P. Walter, *Molecular biology of the cell*, Garland Science, New York, 6th edn, 2015.
2 *Physics of Biological Membranes*, ed. P. Bassereau and P. Sens, Springer International Publishing, Cham, 2018.
3 F. Frey and T. Idema, *Soft Matter*, 2021, **17**, 3533–3549.
4 R. Phillips, J. Kondev, J. Theriot and H. Garcia, *Physical biology of the cell*, Garland Science, New York, 2012.
5 M. Siggel, S. Kehl, K. Reuter, J. Köfinger and G. Hummer, *J. Chem. Phys.*, 2022, **157**, 174801.
6 R. Dimova, *Annu. Rev. Biophys.*, 2019, **48**, 93–119.
7 T. Idema and D. J. Kraft, *Curr. Opin. Colloid Interface Sci.*, 2019, **40**, 58–69.
8 H. Turlier and T. Betz, *Annu. Rev. Condens. Matter Phys.*, 2019, **10**, 213–232.
9 P. Bassereau, B. Sorre and A. Lévy, *Adv. Colloid Interface Sci.*, 2014, **208**, 47–57.
10 H. T. McMahon and J. L. Gallop, *Nature*, 2005, **438**, 590–596.
11 R. Lipowsky, *Adv. Biol.*, 2022, **6**, 2101020.
12 M. Deserno, *Chem. Phys. Lipids*, 2015, **185**, 11–45.
13 U. Seifert, *Adv. Phys.*, 1997, **46**, 13–137.
14 R. Lipowsky, *Nature*, 1991, **349**, 475–481.
15 J. Beiter and G. A. Voth, *Curr. Opin. Cell Biol.*, 2024, **87**, 102831.
16 L. Li, X. Huang and J. Wang, *Giant*, 2024, **18**, 100263.
17 A. K. Dasanna and D. A. Fedosov, *Eur. Phys. J.-Spec. Top.*, 2024, **233**, 3053–3071.
18 G. Kumar, S. C. Duggisetty and A. Srivastava, *J. Membr. Biol.*, 2022, **255**, 757–777.
19 W. Kühlbrandt, *Science*, 2014, **343**, 1443–1444.
20 S. J. Sahl, S. W. Hell and S. Jakobs, *Nat. Rev. Mol. Cell Biol.*, 2017, **18**, 685–701.
21 A.-K. Pfitzner, V. Mercier, X. Jiang, J. Moser von Filseck, B. Baum, A. Šarić and A. Roux, *Cell*, 2020, **182**, 1140–1155.
22 M. Mund, A. Tschanz, Y.-L. Wu, F. Frey, J. L. Mehl, M. Kaksonen, O. Avinoam, U. S. Schwarz and J. Ries, *J. Cell Biol.*, 2023, **222**, e202206038.
23 J. Steinkühler, R. L. Knorr, Z. Zhao, T. Bhatia, S. M. Bartelt, S. Wegner, R. Dimova and R. Lipowsky, *Nat. Commun.*, 2020, **11**, 905.
24 A. Azadbakht, B. Meadowcroft, T. Varkevisser, A. Šarić and D. J. Kraft, *Nano Lett.*, 2023, **23**, 4267–4273.
25 A. Azadbakht, B. Meadowcroft, J. Májek, A. Šarić and D. J. Kraft, *Biophys. J.*, 2024, **123**, 307–316.
26 V. Willems, A. Baron, D. A. M. Fernandez, G. Wolfisberg, E. R. Dufresne, J.-C. Baret and L. Alvarez, *Soft Matter*, 2025, DOI: **10.1039/d5sm00309a**.
27 S. J. Marrink, A. H. De Vries and D. P. Tieleman, *Biochim. Biophys. Acta, Biomembr.*, 2009, **1788**, 149–168.
28 D. Álvarez, J. Sapia and S. Vanni, *Curr. Opin. Cell Biol.*, 2023, **83**, 102205.
29 A. H. Larsen, *Int. J. Mol. Sci.*, 2022, **23**, 8098.
30 H. Noguchi, *J. Phys. Soc. Jpn.*, 2009, **78**, 041007.
31 A. Guckenberger and S. Gekle, *J. Phys.: Condens. Matter*, 2017, **29**, 203001.
32 W. Pezeshkian and S. J. Marrink, *Curr. Opin. Cell Biol.*, 2021, **71**, 103–111.
33 M. Munoz-Basagoiti, F. Frey, B. Meadowcroft, M. Amaral, A. Prada and A. Saric, *Tutorials for mesoscale computer simulations of lipid membranes*, Zenodo repository, DOI: **10.5281/zenodo.14857644**.
34 W. Helfrich and R.-M. Servuss, *Il Nuovo Cimento D*, 1984, **3**, 137–151.
35 M. Deserno, *Phys. Rev. E: Stat., Nonlinear, Soft Matter Phys.*, 2004, **69**, 031903.
36 L. Foret, *Eur. Phys. J. E: Soft Matter Biol. Phys.*, 2014, **37**, 42.
37 F. Frey, F. Ziebert and U. S. Schwarz, *Phys. Rev. Lett.*, 2019, **122**, 088102.
38 F. Frey, F. Ziebert and U. S. Schwarz, *Phys. Rev. E*, 2019, **100**, 052403.
39 U. Seifert, K. Berndl and R. Lipowsky, *Phys. Rev. A: At., Mol., Opt. Phys.*, 1991, **44**, 1182–1202.
40 S. Christ, T. Litschel, P. Schwille and R. Lipowsky, *Soft Matter*, 2021, **17**, 319–330.
41 F. Frey and T. Idema, *Phys. Rev. E*, 2022, **106**, 024401.
42 A.-S. Smith, E. Sackmann and U. Seifert, *Phys. Rev. Lett.*, 2004, **92**, 208101.
43 I. Derényi, F. Jülicher and J. Prost, *Phys. Rev. Lett.*, 2002, **88**, 238101.
44 U. Seifert and R. Lipowsky, *Phys. Rev. A: At., Mol., Opt. Phys.*, 1990, **42**, 4768–4771.
45 F. Jülicher and R. Lipowsky, *Phys. Rev. E: Stat. Phys., Plasmas, Fluids, Relat. Interdiscip. Top.*, 1996, **53**, 2670–2683.
46 F. Jülicher and R. Lipowsky, *Phys. Rev. Lett.*, 1993, **70**, 2964–2967.
47 S. Dasgupta, T. Auth and G. Gompper, *Soft Matter*, 2013, **9**, 5473–5482.

48 S. Dasgupta, T. Auth and G. Gompper, *Nano Lett.*, 2014, **14**, 687–693.
49 H. Ni and G. A. Papoian, *J. Phys. Chem. B*, 2021, **125**, 10710–10719.
50 H. R. Vutukuri, M. Hoore, C. Abaurrea-Velasco, L. van Buren, A. Dutto, T. Auth, D. A. Fedosov, G. Gompper and J. Vermant, *Nature*, 2020, **586**, 52–56.
51 P. Iyer, G. Gompper and D. A. Fedosov, *Soft Matter*, 2023, **19**, 3436–3449.
52 W. Pezeshkian and J. H. Ipsen, *Nat. Commun.*, 2024, **15**, 548.
53 J. Weichsel and P. L. Geissler, *PLoS Comput. Biol.*, 2016, **12**, e1004982.
54 G. Koster, A. Cacciuto, I. Derényi, D. Frenkel and M. Dogterom, *Phys. Rev. Lett.*, 2005, **94**, 068101.
55 K. A. Brakke, *Exper. Math.*, 1992, **1**, 141–165.
56 C. Zhu, C. T. Lee and P. Rangamani, *Biophys. Rep.*, 2022, **2**, 100062.
57 D. A. Matoz-Fernandez, S. Li, M. O. de la Cruz and R. Sknepnek, PyMembrane: A flexible framework for efficient simulations of elastic and liquid membranes, *arXiv*, preprint, 2023, arXiv:2308.12754 [cond-mat, physics:physics], DOI: **10.48550/arXiv.2308.12754**.
58 G. Dadunashvili and T. Idema, flippy: User friendly and open source framework for lipid membrane simulations, *arXiv*, preprint, 2023, arXiv:2303.12305 [cond-mat, physics:physics], DOI: **10.48550/arXiv.2303.12305**.
59 T. S. Carpenter, C. A. López, C. Neale, C. Montour, H. I. Ingólfsson, F. Di Natale, F. C. Lightstone and S. Gnanakaran, *J. Chem. Theory Comput.*, 2018, **14**, 6050–6062.
60 R. S. Davis, P. B. Sunil Kumar, M. M. Sperotto and M. Laradji, *J. Phys. Chem. B*, 2013, **117**, 4072–4080.
61 T. A. Wassenaar, H. I. Ingólfsson, R. A. Böckmann, D. P. Tieleman and S. J. Marrink, *J. Chem. Theory Comput.*, 2015, **11**, 2144–2155.
62 S. Jo, J. B. Lim, J. B. Klauda and W. Im, *Biophys. J.*, 2009, **97**, 50–58.
63 Y. Qi, H. I. Ingólfsson, X. Cheng, J. Lee, S. J. Marrink and W. Im, *J. Chem. Theory Comput.*, 2015, **11**, 4486–4494.
64 S. O. Yesylevskyy, L. V. Schäfer, D. Sengupta and S. J. Marrink, *PLoS Comput. Biol.*, 2010, **6**, e1000810.
65 F. Khalili-Araghi, J. Gumbart, P.-C. Wen, M. Sotomayor, E. Tajkhorshid and K. Schulten, *Curr. Opin. Struct. Biol.*, 2009, **19**, 128–137.
66 T. Mandal, S. E. Spagnolie, A. Audhya and Q. Cui, *Biophys. J.*, 2021, **120**, 3211–3221.
67 M. P. Muller, T. Jiang, C. Sun, M. Lihan, S. Pant, P. Mahinthichaichan, A. Trifan and E. Tajkhorshid, *Chem. Rev.*, 2019, **119**, 6086–6161.
68 S. Jo, T. Kim, V. G. Iyer and W. Im, *J. Comput. Chem.*, 2008, **29**, 1859–1865.
69 M. Abraham, A. Alekseenko, V. Basov, C. Bergh, E. Briand, A. Brown, M. Doijade, G. Fiorin, S. Fleischmann, S. Gorelov, G. Gouaillardet, A. Grey, M. E. Irrgang, F. Jalalypour, J. Jordan, C. Kutzner, J. A. Lemkul, M. Lundborg, P. Merz, V. Miletic, D. Morozov, J. Nabet, S. Pall, A. Pasquadibisceglie, M. Pellegrino, H. Santuz, R. Schulz, T. Shugaeva, A. Shvetsov, A. Villa, S. Wingbermuehle, B. Hess and E. Lindahl, *GROMACS 2024.2 Manual*, 2024, DOI: **10.5281/zenodo.11148638**.
70 J. C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R. D. Skeel, L. Kalé and K. Schulten, *J. Comput. Chem.*, 2005, **26**, 1781–1802.
71 O. Farago and N. Grønbech-Jensen, *J. Am. Chem. Soc.*, 2009, **131**, 2875–2881.
72 S. L. Foley and M. Deserno, *J. Chem. Phys.*, 2024, **160**, 064111.
73 M. Hu, J. J. Briguglio and M. Deserno, *Biophys. J.*, 2012, **102**, 1403–1410.
74 I. R. Cooke, K. Kremer and M. Deserno, *Phys. Rev. E: Stat., Nonlinear, Soft Matter Phys.*, 2005, **72**, 011506.
75 J. Shillcock and R. Lipowsky, *J. Phys.: Condens. Matter*, 2006, **18**, S1191–S1219.
76 X. Jiang, L. Harker-Kirschneck, C. Vanhille-Campos, A.-K. Pfitzner, E. Lominadze, A. Roux, B. Baum and A. Šarić, *PLoS Comput. Biol.*, 2022, **18**, e1010586.
77 W. Bao, F. Tian, C. Lyu, B. Liu, B. Li, L. Zhang, X. Liu, F. Li, D. Li, X. Gao, S. Wang, W. Wei, X. Shi and Y. Li, *Sci. Adv.*, 2021, **7**, eaba2458.
78 J. C. Forster, J. Krausser, M. R. Vuyyuru, B. Baum and A. S. Šarić, *Phys. Rev. Lett.*, 2020, **125**, 228101.
79 D. A. Rower and P. J. Atzberger, *Math. Comput. Simul.*, 2023, **209**, 342–361.
80 C. Vanhille-Campos and A. Šarić, *Soft Matter*, 2021, **17**, 3798–3806.
81 C. Huang, H. Yuan and S. Zhang, *Appl. Phys. Lett.*, 2011, **98**, 043702.
82 H. Yuan, C. Huang and S. Zhang, *Soft Matter*, 2010, **6**, 4571–4579.
83 L. Harker-Kirschneck, A. E. Hafner, T. Yao, C. Vanhille-Campos, X. Jiang, A. Pulschen, F. Hurtig, D. Hryniuk, S. Culley, R. Henriques, B. Baum and A. Šarić, *Proc. Natl. Acad. Sci. U. S. A.*, 2022, **119**, e2107763119.
84 A. Singh and Y. Li, *Advanced Materials: Design, Processing, Characterization and Applications*, Advances in Aerospace Technology, Columbus, Ohio, USA, 2022, vol. 3, p. V003T03A001.
85 M. Becton, R. Averett and X. Wang, *Biomech. Model. Mechanobiol.*, 2019, **18**, 425–433.
86 M. Yu, W. Song, F. Tian, Z. Dai, Q. Zhu, E. Ahmad, S. Guo, C. Zhu, H. Zhong, Y. Yuan, T. Zhang, X. Yi, X. Shi, Y. Gan and H. Gao, *Proc. Natl. Acad. Sci. U. S. A.*, 2019, **116**, 201818924.
87 F. Ahmadpoor, G. Zou and H. Gao, *Mech. Mater.*, 2022, **174**, 104414.
88 V. E. Debets, L. M. C. Janssen and A. Šarić, *Soft Matter*, 2020, **16**, 10628–10639.
89 Y. Zhang, C. Huang, S. Kim, M. Golkaram, M. Dixon, L. Tilley, J. Li, S. Zhang and S. Suresh, *Proc. Natl. Acad. Sci. U. S. A.*, 2015, **112**, 6068–6073.
90 M. Sadeghi and F. Noé, *J. Phys. Chem. Lett.*, 2021, **12**, 10497–10504.
91 A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. I. T. Veld,

A. Kohlmeyer, S. G. Moore, T. D. Nguyen, R. Shan, M. J. Stevens, J. Tranchida, C. Trott and S. J. Plimpton, *Comput. Phys. Commun.*, 2022, **271**, 108171.
92 J. A. Anderson, J. Glaser and S. C. Glotzer, *Comput. Mater. Sci.*, 2020, **173**, 109363.
93 F. Weik, R. Weeber, K. Szuttor, K. Breitsprecher, J. De Graaf, M. Kuron, J. Landsgesell, H. Menke, D. Sean and C. Holm, *Eur. Phys. J.: Spec. Top.*, 2019, **227**, 1789–1816.
94 W. Helfrich, *Z. Naturforsch., C*, 1973, **28**, 693–703.
95 O. Farago and P. Pincus, *Eur. Phys. J. E: Soft Matter Biol. Phys.*, 2003, **11**, 399–408.
96 P. Rangamani, *Biochim. Biophys. Acta, Biomembr.*, 2022, **1864**, 183897.
97 M. Hamm and M. Kozlov, *Eur. Phys. J. E: Soft Matter Biol. Phys.*, 2000, **3**, 323–335.
98 M. C. Watson, E. S. Penev, P. M. Welch and F. L. H. Brown, *J. Chem. Phys.*, 2011, **135**, 244701.
99 M. M. Terzi, M. F. Ergüder and M. Deserno, *J. Chem. Phys.*, 2019, **151**, 164108.
100 O.-Y. Zhong-can and W. Helfrich, *Phys. Rev. A: At., Mol., Opt. Phys.*, 1989, **39**, 5280–5288.
101 W. Helfrich and R. M. Servuss, *Il Nuovo Cimento D*, 1984, **3**, 137–151.
102 A. H. Bahrami, M. Raatz, J. Agudo-Canalejo, R. Michel, E. M. Curtis, C. K. Hall, M. Gradzielski, R. Lipowsky and T. R. Weikl, *Adv. Colloid Interface Sci.*, 2014, **208**, 214–224.
103 D. Frenkel and B. Smit, *Understanding Molecular Simulation*, Academic Press, 2002.
104 S. Gravelle, *LAMMPS tutorials*, 2024, **https://lammpstutorials.github.io/**.
105 G. Gompper and D. M. Kroll, *J. Phys.: Condens. Matter*, 1997, **9**, 8795–8834.
106 Y. Kantor and D. R. Nelson, *Phys. Rev. A: At., Mol., Opt. Phys.*, 1987, **36**, 4020–4032.
107 F. Jülicher, *J. Phys. II*, 1996, **6**, 1797–1824.
108 G. Gompper and D. M. Kroll, *J. Phys. I*, 1996, **6**, 1305–1320.
109 X. Bian, S. Litvinov and P. Koumoutsakos, *Comput. Methods Appl. Mech. Eng.*, 2020, **359**, 112758.
110 G. Gompper and D. M. Kroll, *Statistical Mechanics of Membranes and Surfaces*, World Scientific, 2004, pp. 359–426.
111 A. H. Bahrami and G. Hummer, *ACS Nano*, 2017, **11**, 9558–9565.
112 S. K. Veerapaneni, D. Gueyffier, G. Biros and D. Zorin, *J. Comput. Phys.*, 2009, **228**, 7233–7249.
113 A. Paraschiv, T. J. Lagny, C. V. Campos, E. Coudrier, P. Bassereau and A. Šarić, *Biophys. J.*, 2021, **120**, 598–606.
114 A. S. Šarić and A. Cacciuto, *Phys. Rev. Lett.*, 2012, **108**, 118101.
115 M. Tachikawa and A. Mochizuki, *Proc. Natl. Acad. Sci. U. S. A.*, 2017, **114**, 5177–5182.
116 G. Gompper and D. M. Kroll, *Phys. Rev. Lett.*, 1998, **81**, 2284–2287.
117 S. J. Marrink, A. H. De Vries and A. E. Mark, *J. Phys. Chem. B*, 2004, **108**, 750–760.
118 C. Arnarez, J. J. Uusitalo, M. F. Masman, H. I. Ingólfsson, D. H. de Jong, M. N. Melo, X. Periole, A. H. de Vries and S. J. Marrink, *J. Chem. Theory Comput.*, 2015, **11**, 260–275.
119 I. R. Cooke and M. Deserno, *J. Chem. Phys.*, 2005, **123**, 224710.
120 H. Yuan, C. Huang, J. Li, G. Lykotrafitis and S. Zhang, *Phys. Rev. E: Stat., Nonlinear, Soft Matter Phys.*, 2010, **82**, 011905.
121 S. Izvekov and G. A. Voth, *J. Phys. Chem. B*, 2005, **109**, 2469–2473.
122 M. G. Saunders and G. A. Voth, *Annu. Rev. Biophys.*, 2013, **42**, 73–93.
123 S. Seo and W. Shinoda, *J. Chem. Theory Comput.*, 2019, **15**, 762–774.
124 M. Orsi and J. W. Essex, *PLoS One*, 2011, **6**, 1–22.
125 T. D. Loose, P. G. Sahrmann, T. S. Qu and G. A. Voth, *J. Chem. Theory Comput.*, 2024, **20**, 9190–9208.
126 S. Moradi, A. Nowroozi and M. Shahlaei, *RSC Adv.*, 2019, **9**, 4644–4658.
127 M. Venturoli, M. Maddalena Sperotto, M. Kranenburg and B. Smit, *Phys. Rep.*, 2006, **437**, 1–54.
128 D. J. Smith, J. B. Klauda and A. J. Sodt, *Living J. Comput. Mol. Sci.*, 2019, **1**, 5966.
129 M. Deserno, *Macromol. Rapid Commun.*, 2009, **30**, 752–771.
130 R. Goetz, G. Gompper and R. Lipowsky, *Phys. Rev. Lett.*, 1999, **82**, 221–224.
131 J. C. Shillcock and R. Lipowsky, *J. Chem. Phys.*, 2002, **117**, 5048–5061.
132 M. Laradji and P. B. Sunil Kumar, *Phys. Rev. Lett.*, 2004, **93**, 198105.
133 O. Farago and N. Grønbech-Jensen, *J. Chem. Phys.*, 2016, **144**, 084102.
134 O. Lenz and F. Schmid, *J. Mol. Liq.*, 2005, **117**, 147–152.
135 J.-M. Drouffe, A. C. Maggs and S. Leibler, *Science*, 1991, **254**, 1353–1356.
136 H. Noguchi and M. Takasu, *Phys. Rev. E: Stat., Nonlinear, Soft Matter Phys.*, 2001, **64**, 041913.
137 O. Farago, *J. Chem. Phys.*, 2003, **119**, 596–605.
138 G. Brannigan and F. L. H. Brown, *J. Chem. Phys.*, 2004, **120**, 1059–1071.
139 G. Brannigan, P. F. Philips and F. L. H. Brown, *Phys. Rev. E: Stat., Nonlinear, Soft Matter Phys.*, 2005, **72**, 011915.
140 Z.-J. Wang and D. Frenkel, *J. Chem. Phys.*, 2005, **122**, 234711.
141 J. D. Revalee, M. Laradji and P. B. Sunil Kumar, *J. Chem. Phys.*, 2008, **128**, 035102.
142 Z.-J. Wang and D. Frenkel, *J. Chem. Phys.*, 2005, **123**, 154701.
143 R. Vácha, F. J. Martinez-Veracoechea and D. Frenkel, *Nano Lett.*, 2011, **11**, 5391–5395.
144 W. K. den Otter and W. J. Briels, *J. Chem. Phys.*, 2003, **118**, 4712–4720.
145 H. Noguchi and G. Gompper, *Phys. Rev. E: Stat., Nonlinear, Soft Matter Phys.*, 2006, **73**, 021903.
146 P. Ballone and M. G. Del Pópolo, *Phys. Rev. E: Stat., Nonlinear, Soft Matter Phys.*, 2006, **73**, 031404.

147 T. Kohyama, *Phys. A*, 2009, **388**, 3334–3344.
148 H. Noguchi, *J. Chem. Phys.*, 2013, **138**, 024907.
149 C. Huang, Y. Zhang, H. Yuan, H. Gao and S. Zhang, *Nano Lett.*, 2013, **13**, 4546–4550.
150 E. Weiner, E. Berryman, F. Frey, A. G. Solís, A. Leier, T. M. Lago, A. Šarić and M. S. Otegui, *Proc. Natl. Acad. Sci. U. S. A.*, 2024, **121**, e2409407121.
151 S.-P. Fu, Z. Peng, H. Yuan, R. Kfoury and Y.-N. Young, *Comput. Phys. Commun.*, 2017, **210**, 193–203.
152 P. Appshaw, A. M. Seddon and S. Hanna, *Soft Matter*, 2022, **18**, 1747–1756.
153 S. Wang, S. Ma, H. Li, M. Dao, X. Li and G. E. Karniadakis, *Biophys. J.*, 2024, **123**, 1069–1084.
154 H. Yuan, C. Huang and S. Zhang, *Soft Matter*, 2010, **6**, 4571.
155 M. Sadeghi and F. Noé, *Nat. Commun.*, 2020, **11**, 2951.
156 H. Noguchi and G. Gompper, *Phys. Rev. Lett.*, 2004, **93**, 258102.
157 H. Noguchi and G. Gompper, *Proc. Natl. Acad. Sci. U. S. A.*, 2005, **102**, 14159–14164.
158 H. Shiba, H. Noguchi and G. Gompper, *J. Chem. Phys.*, 2013, **139**, 014702.
159 G. Gompper, T. Ihle, D. M. Kroll and R. G. Winkler, in *Multi-Particle Collision Dynamics: A Particle-Based Mesoscale Simulation Approach to the Hydrodynamics of Complex Fluids*, ed. C. Holm and K. Kremer, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 1–87.
160 P. J. Hoogerbrugge and J. M. V. A. Koelman, *Europhys. Lett.*, 1992, **19**, 155.
161 C. S. Peskin, *J. Comput. Phys.*, 1977, **25**, 220–252.
162 Z. Guo and T. S. Zhao, *Phys. Rev. E: Stat., Nonlinear, Soft Matter Phys.*, 2002, **66**, 036304.
163 D. Fedosov, B. Caswell and G. Karniadakis, *Biophys. J.*, 2010, **98**, 2215–2225.
164 Y. Liu and W. K. Liu, *J. Comput. Phys.*, 2006, **220**, 139–154.
165 M. M. Dupin, I. Halliday, C. M. Care, L. Alboul and L. L. Munn, *Phys. Rev. E: Stat., Nonlinear, Soft Matter Phys.*, 2007, **75**, 066707.
166 A. Rezghi and J. Zhang, *Biophys. J.*, 2022, **121**, 3393–3410.
167 M. Wassermair, M. Munoz-Basagoiti, M. Amaral and A. Saric, *TriLMP Github repository*, **https://github.com/Saric-Group/trimem_sbeady**.
168 D. Raucher and M. P. Sheetz, *Biophys. J.*, 1999, **77**, 1992–2002.
169 T. R. Powers, G. Huber and R. E. Goldstein, *Phys. Rev. E: Stat., Nonlinear, Soft Matter Phys.*, 2002, **65**, 041901.
170 F. Schmid, *Europhys. Lett.*, 2011, **95**, 28008.
171 J.-B. Fournier, *Europhys. Lett.*, 2012, **97**, 18001.
172 V. A. Harmandaris and M. Deserno, *J. Chem. Phys.*, 2006, **125**, 204905.
173 M. F. Ergüder and M. Deserno, *J. Chem. Phys.*, 2021, **154**, 214103.
174 J. D. Chodera, *J. Chem. Theory Comput.*, 2016, **12**, 1799–1805.
175 O. Farago, *J. Chem. Phys.*, 2008, **128**, 184105.
176 M. Loose, E. Fischer-Friedrich, J. Ries, K. Kruse and P. Schwille, *Science*, 2008, **320**, 789–792.
177 F. Wu, J. Halatek, M. Reiter, E. Kingma, E. Frey and C. Dekker, *Mol. Syst. Biol.*, 2016, **12**, 873.
178 Y. Dreher, K. Jahnke, E. Bobkova, J. P. Spatz and K. Göpfrich, *Angew. Chem., Int. Ed.*, 2021, **60**, 10661–10669.
179 L. Baldauf, L. van Buren, F. Fanalista and G. H. Koenderink, *ACS Synth. Biol.*, 2022, 3120–3133.
180 M. Beck, R. Covino, I. Hänelt and M. Müller-McNicoll, *Cell*, 2024, **187**, 545–562.
181 M. Deserno, *Macromol. Rapid Commun.*, 2009, **30**, 752–771.
182 H. Jung, R. Covino, A. Arjun, C. Leitold, C. Dellago, P. G. Bolhuis and G. Hummer, *Nat. Comput. Sci.*, 2023, **3**, 334–345.
183 P. G. Sahrmann and G. A. Voth, *Curr. Opin. Struct. Biol.*, 2025, **90**, 102972.
184 S. Liu, P. Hoess and J. Ries, *Annu. Rev. Biophys.*, 2022, **51**, 301–326.
185 F. J. Verweij, L. Balaj, C. M. Boulanger, D. R. F. Carter, E. B. Compeer, G. D'Angelo, S. El Andaloussi, J. G. Goetz, J. C. Gross, V. Hyenne, E.-M. Krämer-Albers, C. P. Lai, X. Loyer, A. Marki, S. Momma, E. N. M. Nolte-'t Hoen, D. M. Pegtel, H. Peinado, G. Raposo, K. Rilla, H. Tahara, C. Théry, M. E. van Royen, R. E. Vandenbroucke, A. M. Wehman, K. Witwer, Z. Wu, R. Wubbolts and G. van Niel, *Nat. Methods*, 2021, **18**, 1013–1026.