3D mesh processing using GAMer 2 to enable reaction-diffusion simulations in realistic cellular geometries

Christopher T. Lee[2,◕], Justin G. Laughlin[2,◕], Nils Angliviel de La Beaumelle[2], Rommie E. Amaro[1], J. Andrew McCammon[1], Ravi Ramamoorthi[3], Michael J. Holst[4], and Padmini Rangamani[2*]

1 Department of Chemistry and Biochemistry, University of California, San Diego, La Jolla, CA, 92093 USA
2 Department of Mechanical and Aerospace Engineering, University of California, San Diego, La Jolla, CA, 92093 USA
3 Department of Computer Science, University of California, San Diego, La Jolla, CA, 92093 USA
4 Department of Mathematics, University of California, San Diego, La Jolla, CA, 92093 USA

◕ These authors contributed equally to this work.
* prangamani@ucsd.edu

Abstract

Recent advances in electron microscopy have enabled the imaging of single cells in 3D at nanometer length scale resolutions. An uncharted frontier for *in silico* biology is the ability to simulate cellular processes using these observed geometries. Enabling such simulations requires watertight meshing of electron micrograph images into 3D volume meshes, which can then form the basis of computer simulations of such processes using numerical techniques such as the Finite Element Method (FEM). In this paper, we describe the use of our recently rewritten mesh processing software, `GAMer 2`, to bridge the gap between poorly conditioned meshes generated from segmented micrographs and boundary marked tetrahedral meshes which are compatible with simulation. We demonstrate the application of a workflow using `GAMer 2` to a series of electron micrographs of neuronal dendrite morphology explored at three different length scales and show that the resulting meshes are suitable for finite element simulations. This work is an important step towards making physical simulations of biological processes in realistic geometries routine. Innovations in algorithms to reconstruct and simulate cellular length scale phenomena based on emerging structural data will enable realistic physical models and advance discovery at the interface of geometry and cellular processes. We posit that a new frontier at the intersection of computational technologies and single cell biology is now open.

Author summary

3D imaging of cellular components and associated reconstruction methods have made great strides in the past decade, opening windows into the complex intracellular organization. These advances also mean that computational tools need to be developed to work with these images not just for purposes of visualization but also for biophysical simulations. In this work, we present our recently rewritten mesh processing software, `GAMer 2`, which features both mesh conditioning algorithms and tools to support simulation setup including boundary marking. Using a workflow that consists of

other open-source softwares along with `GAMer 2`, we demonstrate the process of going from electron micrographs to simulations for several scenes of increasing length scales. In our preliminary finite element simulations of reaction-diffusion in the generated geometries, we reaffirm that the complex morphology of the cell can impact processes such as signaling. Technologies such as these presented here are set to enable a new frontier in biophysical simulations in realistic geometries.

List of Acronyms

BPAP	Back Propagating Action Potential
EM	Electron Microscopy
EPSP	Excitatory Postsynaptic Potential
ER	Endoplasmic Reticulum
FEA	Finite Element Analysis
FEM	Finite Element Method
FIB-SEM	Focused-ion Beam Milling Scanning Electron Microscopy
LST	Local Structure Tensor
NMDAR	N-methyl-D-aspartate Receptor
PDE	Partial Differential Equation
PM	Plasma Membrane
PSD	Postsynaptic Density
SBF-SEM	Serial Block-Face Scanning Electron Microscopy

Introduction

Understanding structure-function relationships at cellular length scales (nm to μm) is one of the central goals of modern cell biology. While structural determination techniques are routine for very small and large scales such as molecular and tissue, high-resolution images of mesoscale subcellular scenes were historically elusive [1]. This was primarily due to the diffraction limits of visible light and the limitations of X-ray and Electron Microscopy (EM) hardware. Over the past decade, technological improvements such as improved direct electron detectors have enabled the practical applications of techniques such as volume electron microscopy [2–6]. Advances in microscopy techniques in recent years have opened windows into cells, giving us insight into cellular organization with unprecedented detail [7–13]. Volumetric EM enables the capture of 3D ultrastructural datasets (i.e., images where fine structures such as membranes of cells and their internal organelles are resolved, as shown in Fig. 1A). Using these geometries as the basis of simulations provides an opportunity for the *in silico* animation of various cellular processes and the generation of experimentally testable hypotheses. Popular biophysical simulation modalities such as the finite element method [14, 15], particle-based stochastic dynamics [16–19], and the reaction-diffusion master equation [20–26] among many others require discretizations or meshes representing the geometry of the domain of interest. Moreover, in biological systems, the localization of molecular species is often heterogeneous [27, 28] which necessitates the need for boundary and region marking to represent this heterogeneity. To realize these simulations, therefore, a workflow is necessary to go from images to high-quality and annotated 3D meshes compatible with different numerical simulation modalities. As we review

below, significant community effort has been invested in the imaging and segmentation steps, as well as mesh generation for graphics and visualization (Fig. 1A-C). In an effort to bridge advances in these fields, in this work we describe `GAMer 2`; software which takes input meshes generated from contour-tiling and segmentation, applies mesh conditioning algorithms from the graphics community, and mark faces to demarcate boundary conditions. `GAMer 2` is developed for the common biophysicist and features an easy to use user interface implemented as an add-on to 3D modeling software `Blender` along with a Python API `PyGAMer`. These user interfaces allow for the definition of marked boundaries corresponding to molecular localizations. The output of `GAMer 2` is a boundary marked and simulation compatible surface or volume mesh (Fig. 1D and E) on which one can run finite element-based biophysical simulations (Fig. 1F).

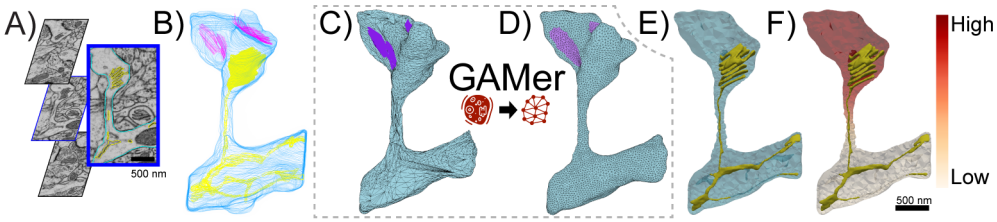

Fig 1. Pipeline from electron microscopy data to a reaction-diffusion finite element simulation on a well-conditioned unstructured tetrahedral mesh. A) Contours of segmented data overlaid on raw slices of electron microscopy data, B) Stacked contours from all slices of segmented data, C) Primitive initial 3D mesh reconstructed by existing `IMOD` software, D) Surface mesh after processing with our system; note the significantly higher quality of the mesh. The steps from C to D are the core contributions of this manuscript. E) Unstructured tetrahedral mesh suitable for finite element simulation obtained with `TetGen` software linked in `GAMer 2`, F) Reaction-diffusion model simulated using `FEniCS` software.

Workflow Steps from Image to Model

In order to develop models from image datasets, a series of steps must be executed. Starting with image acquisition and ending with a simulation compatible mesh, we summarize the typical workflow steps and highlight potential difficulties along the way.

Image acquisition and segmentation

Sample preparation begins with either cell culture or the harvesting of biological tissues of interest. Subsequent preparation steps can vary depending upon the particular volume EM imaging modality used but primarily include sample dehydration, fixation/staining, embedding, and imaging through the different cross-sections [29–32].

Once the images are captured, they are post-processed to improve properties such as contrast and alignment/registration across the stack. From the processed image stack, the boundaries of interesting features are traced or segmented. To the best of our knowledge, the state-of-the-art for segmenting electron micrographs of cells remains reliant upon the expertise of biologists for recognition of organelles and membrane domains in cells. During the segmentation process, the algorithm or researcher must carefully separate boundary signal from noise. Various schemes ranging from manual tracing, thresholding and edge-detection, to deep-learning based approaches have been employed to perform image segmentation [32, 33].

The resulting segmentations from volume EM can be visualized as stacks of contours (Fig. 1B). This provides an initial glimpse into the 3D shapes of objects of interest. In order to enable modeling using the shapes represented by the contours, geometric meshes compatible with numerical methods can be constructed. However, a myriad of complexities often confound this process and necessitate flexible approaches for mesh generation.

Meshing challenges

A variety of challenges for meshing and subsequent physical simulations can arise at each step. Even with near-perfect experimental execution, and despite the enhanced surface contrast from heavy metal stains, images of cellular and organelle membranes are often poorly behaved and contain sharp and otherwise irregular geometries that are difficult to segment. In more serious cases, thinly sliced samples can tear or become contaminated during handling. Methodological errors are also possible. For example, Serial Block-Face Scanning Electron Microscopy (SBF-SEM) datasets in optimum conditions may have 3 nm lateral (x,y) resolution but 25 nm axial (z) resolution, limited by the slicing capability of the ultramicrotome [29]. Anisotropic resolution in tandem with variable slice thickness can cause loss of axial detail.

There are many EM softwares that post-process image stacks to correct for these and other artifacts. Most of our datasets have been manually segmented and corrected in software such as `IMOD` [34], `ilastik` [35], or `TrackEM2` [36]. `IMOD` and other tools such as `ContourTiler` in `VolRoverN` [37] among others [38, 39] have the capacity to perform contour-tiling operations to generate a preliminary surface mesh suitable for basic 3D visualization. If the end goal is visualizing the geometry of the cellular structure, then such meshing operations are often sufficient.

Meshes generated in this manner, however, are often not directly suitable for physical simulations due to various mesh artifacts as described below (Fig. 1C). Some of these include jagged boundaries, non-manifold features, and high aspect ratio faces, as shown in Fig. 2. These problems must be corrected to produce a conditioned surface mesh that is compatible with physical simulations (Fig. 1D). For simulations that track concentrations in the volume, the conditioned surface mesh is tetrahedralized (Fig. 1E). We note that although there exist advanced tetrahedral mesh generation tools, such as `TetWild` [40], and others [41–46], which can generate Finite Element Analysis (FEA) compatible volume meshes from these poor quality initial surfaces, these tools are general purpose and currently not adapted to the length scales of single cells and subcellular structures. Mesh defects such as disconnects in the Endoplasmic Reticulum (ER) arising from the limited resolving powers of EM or errors in segmentation (e.g., Fig. 2C, D) require more careful and often manual curation. Currently, manual curation of cellular data sets remain the gold standard for identifying and matching cellular structures. A specialist trained in imaging modalities is capable of subjectively matching the identity of a surface with the underlying micrograph along with some history of observations from training to determine if an error is likely.

Curating simulation metadata

Once a suitable mesh is generated, other steps may be necessary to facilitate successful simulation. For example, when modeling biochemical signal transduction, receptors and other molecules may be localized to particular regions of a scene [47]. Realistic simulations must be able to represent the observed localization to effectively predict cellular behavior [48–50]. In a simulated model, the confinement of molecules can be presented as boundary conditions on a marked mesh region (Fig. 1D). Depending on the situation, such localizations can be arbitrarily or randomly assigned for hypothesis testing [51] Alternatively, the regions of confinement may be informed by the electron micrographs

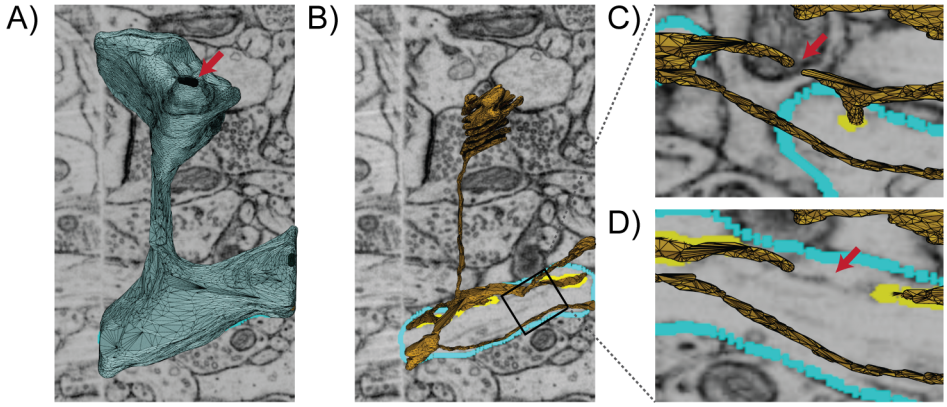

Fig 2. Initial surface mesh model of a single spine scene with subcellular organelles imaged by Focused-ion Beam Milling Scanning Electron Microscopy (FIB-SEM) (overlaid), courtesy of Wu et al. [7], contains many mesh artifacts and is not compatible with physics-based simulations. A) The blue surface represents the Plasma Membrane (PM) which contains a hole indicated by the red arrow. B) The yellow surface represents the membrane of the Endoplasmic Reticulum (ER). C, D) are two views rotated and zoomed-in on B showing a disconnected region of the ER proofed against micrographs taken at different z-axis locations. An untraced ghost, indicated by the red arrow, appears between the disconnected segments which suggests a possible error in the segmentation.

themselves or correlated from another experimental approaches. A robust mesh generation tool capable of handing and resolving problems across all workflow steps including boundary marking and other metadata curation is necessary to support simulations from images of subcellular scenes (Fig. 1E).

Here, we introduce our recently redesigned software, `GAMer 2` (Geometry-preserving Adaptive MeshER version 2), which features mesh conditioning algorithms and simulation setup tools. In this redesign, we focused on the following software design criteria:

- Easy cross-platform code compilation and distribution.
- Runtime stability with meaningful error messages.
- Easy interactivity for the biophysicist user base.
- Version tracking for code provenance.

The algorithms in `GAMer 2` for mesh conditioning remain those described by Yu *et al.* [52, 53], by Gao *et al.* [54, 55], and Chen and Holst [56]. As described in the original manuscripts, the algorithms seek to preserve mesh features while producing smooth surfaces. We will show in our illustrative examples that `GAMer 2` approximately preserves volume. In this redevelopment, we have introduced the capability to perform local refinements and now provide end-users with access to new meta-parameters such as the number of neighbor rings to use in the calculation of the local structure tensor.

Details of the rewrite including the development of a new Python interface `PyGAMer`, and `GAMer Blender` add-on called `BlendGAMer` follow. In addition, we summarize new geometric capabilities such as the estimation of curvatures on meshes.

Methods

GAMer 2 Development

`GAMer 2` is a complete rewrite of `GAMer` in C++ using the `CASC` data structure [57] as the underlying mesh representation. Prior versions of `GAMer` were susceptible to segmentation faults under certain conditions, which is now fixed in this major update. In addition to improving the run-time stability, we have added error handling code to produce actionable notes for the convenience of the end-user. `GAMer 2` continues to be licensed under LGPL v2.1 and the source code can be downloaded from GitHub (`https://github.com/ctlee/gamer`) [58].

In this update, we have also redeveloped the Python interface for `GAMer` and the `Blender` add-on. The `GAMer 2` Python API, called `PyGAMer` is generated using `pybind11` [59] as opposed to `SWIG` [60] used by `GAMer`. `pybind11` provides superior wrapping of C++ template objects which enables `PyGAMer` to interact with nearly all elements of `GAMer 2` in a Python environment. The `GAMer 2-Blender` add-on, now called `BlendGAMer`, has been rewritten to use the `PyGAMer` interface. `Blender` not only provides a customizable mesh visualization environment, but also tools such as sculpt mode, which allows users to flexibly manipulate the geometry [61]. With great care to remain truthful to the underlying data, `Blender` sculpting tools can be useful for fixing topological issues such as disconnected ER segments. We briefly review the concepts behind the mesh processing algorithms from Yu et al. [52, 53].

Mesh Processing

Local Structure Tensor

The mesh processing operations in `GAMer` are designed to improve mesh quality while preserving the underlying geometry of the data. We use a Local Structure Tensor (LST) to account for the local geometry [62–64]. The LST is defined as follows,

$$T(\mathbf{v}) = \sum_{i=1}^{N_r} \mathbf{n_i} \otimes \mathbf{n_i} = \sum_{i=1}^{N_r} \begin{pmatrix} n_i^x n_i^x & n_i^x n_i^y & n_i^x n_i^z \\ n_i^y n_i^x & n_i^y n_i^y & n_i^y n_i^z \\ n_i^z n_i^x & n_i^z n_i^y & n_i^z n_i^z \end{pmatrix}, \tag{1}$$

where $\mathbf{v}$ is the vertex of interest, N_r is the number of neighbors in the r-ring neighborhood, and $n_i^{x,y,z}$ form the normal of the ith neighbor vertex. Vertex normals are defined as the weighted average of incident face normals. Performing the eigendecomposition of the LST, we obtain information on the principal orientations of normals in the local neighborhood [65]. The magnitude of the eigenvalue corresponds to the amount of curvature along the direction of the corresponding eigenvector. Inspecting the magnitude of the eigenvalues gives several geometric cases:

- Planes: $\lambda_1 \gg \lambda_2 \approx \lambda_3 \approx 0$
- Ridges and valleys: $\lambda_1 \approx \lambda_2 \gg \lambda_3 \approx 0$
- Spheres and saddles: $\lambda_1 \approx \lambda_2 \approx \lambda_3 > 0$

Feature preserving mesh smoothing

Finite element simulations are sensitive to the mesh quality [66, 67]. Poor quality meshes can lead to numerical error, instability, long times to solution, and non-convergence. Generally, triangular meshes with high aspect ratios produce larger errors compared with equilateral elements [68].

To improve the conditioning of the surface meshes derived from microscopy images, we use an angle-weighted Laplacian smoothing approach, as shown in Fig. 3A. This scheme is an extension of the angle weighted smoothing scheme, formulated for 2D and described by Zhou and Shimada, to three dimensions [69]. In essence, this algorithm applies local torsion springs to the 1-ring neighborhood of a vertex of interest to balance the angles.

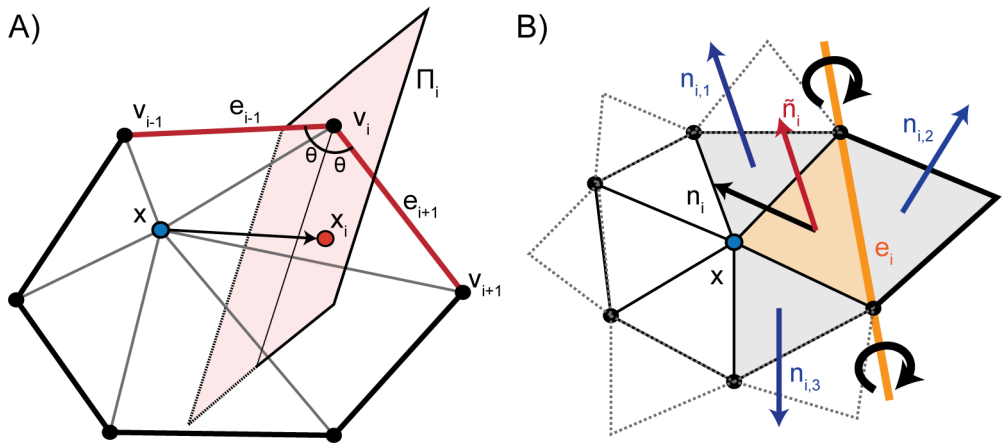

Fig 3. Schematic illustrating `GAMer` mesh conditioning algorithms. A) angle-based surface mesh conditioning, and B) anisotropic normal smoothing algorithms which are previously described by Yu et al. [52, 53] and implemented in `GAMer`.

Given a vertex $\mathbf{x}$ with the set of 1-ring neighbors $\{\mathbf{v}_1, \ldots, \mathbf{v}_N\}$, where N is the number of neighbors, ordered such that $\mathbf{v}_i$ is connected to $\mathbf{v}_{i-1}$ and $\mathbf{v}_{i+1}$ by edges. The 1-ring is connected such that $\mathbf{v}_{N+1} := \mathbf{v}_1$ and $\mathbf{v}_{-1} := \mathbf{v}_N$. Traversing the 1-ring neighbors, we define edge vectors $\mathbf{e}_{i-1} := \overrightarrow{\mathbf{v}_i\mathbf{v}_{i-1}}$ and $\mathbf{e}_{i+1} := \overrightarrow{\mathbf{v}_i\mathbf{v}_{i+1}}$. This algorithm seeks to move $\mathbf{x}$ to lie on the perpendicularly bisecting plane Π_i of $\angle(\mathbf{v}_{i-1}, \mathbf{v}_i, \mathbf{v}_{i+1})$. For each vertex in the 1-ring neighbors, we compute the perpendicular projection, $\mathbf{x_i}$, of $\mathbf{x}$ onto Π_i. Since small surface mesh angles are more sensitive to change in $\mathbf{x}$ position than large angles, we prioritize their maximization. We define a weighting factor, $\alpha_i = \frac{\mathbf{e}_{i-1}\cdot\mathbf{e}_{i+1}}{|\mathbf{e}_{i-1}|\cdot|\mathbf{e}_{i+1}|}$, which inversely corresponds with $\angle(\mathbf{v}_{i-1}, \mathbf{v}_i, \mathbf{v}_{i+1})$. The average of the projections weighted by α_i gives a new position of $\mathbf{x}$ as follows,

$$\bar{\mathbf{x}} = \frac{1}{\sum_{i=1}^{N}(\alpha_i + 1)} \sum_{i=1}^{N} (\alpha_i + 1)\mathbf{x}_i. \tag{2}$$

There are many smoothing algorithms in the literature; the angle-weighted Laplacian smoothing algorithm described here can outperform other popular smoothing strategies such as those described in [70–73] which are primarily focused on optimizing the smoothness of surface normals for computer graphics applications and not mesh angles. Our goal is not to provide an elaborate comparison against existing algorithms in this manuscript but to demonstrate the utility of our pipeline for biological images, with a specific goal of using EM-generated images for computational biology simulations.

Conceptually the fidelity of the local geometry can be maintained by restricting vertex movement along directions of low curvature. This constraint is achieved by anisotropically dampening vertex diffusion using information contained in the LST. Although the weighted vertex smoothing scheme, as described, will reasonably preserve geometric structure, the structure preservation can be further improved by using the LST. Computing the eigendecomposition of the LST, we obtain eigenvalues $\lambda_1, \lambda_2, \lambda_3$ and eigenvectors $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3$, which correspond to principal orientations of local normals.

We project $\bar{\mathbf{x}} - \mathbf{x}$ onto the eigenvector basis and scale each component by the inverse of the corresponding eigenvalue,

$$\hat{\mathbf{x}} = \mathbf{x} + \sum_{k=1}^{3} \frac{1}{1+\lambda_k}[(\bar{\mathbf{x}} - \mathbf{x}) \cdot \mathbf{E}_k]\mathbf{E}_k. \tag{3}$$

This has the effect of dampening movement along directions of high curvature i.e., where λ is large. In this fashion, our algorithm improves triangle aspect ratios while preserving local geometric features. We note that our actual implementation iterates between rounds of vertex smoothing and conventional angle based edge flipping to achieve the desired smoothing effect. Edge flips are common in mesh processing, and provide a mechanism for both improving angles and reducing the valency of vertices [74]. A comparison of the angle-weighted smoothing algorithm with and without LST correction is shown in Fig. 4.

Feature preserving anisotropic normal-based smoothing

To remove additional bumpiness from the mesh, we use a normal-based smoothing approach [75, 76], as shown in Fig. 3B. The goal is to produce smoothly varying normals across the mesh without compromising mesh angle quality. Given a vertex $\mathbf{x}$ of interest, for each incident face i, with normal $\mathbf{n}_i$ we rotate $\mathbf{x}$ around a rotation axis defined by opposing edge e_i such that $\mathbf{n}_i$ aligns with the mean normal of neighboring faces $\bar{\mathbf{n}}_i = \sum_{j=1}^{3} \mathbf{n}_{ij}/3$. We denote the new position which aligns $\mathbf{n}_i$ and $\bar{\mathbf{n}}_i$ as $R(\mathbf{x}; e_i, \theta_i)$. Summing up the rotations and weighting by incident face area, a_i, we get an updated position,

$$\bar{\mathbf{x}} = \frac{1}{\sum_{i=1}^{N_1} a_i} \sum_{i=1}^{N_1} a_i R(\mathbf{x}; e_i, \theta_i). \tag{4}$$

This is an isotropic scheme that is independent of the local geometric features; meaning that many iterations of this algorithm may weaken sharp features.

Instead, we use an anisotropic scheme [76, 77] to compute the mean neighbor normals,

$$\bar{\mathbf{n}}_i = \frac{1}{\sum_{j=1}^{3} e^{K(\mathbf{n}_i \cdot \mathbf{n}_{ij})}} \sum_{j=1}^{3} e^{K(\mathbf{n}_i \cdot \mathbf{n}_{ij})} \mathbf{n}_{ij}, \tag{5}$$

where K is a user defined positive parameter which scales the extent of anisotropy. Under this scheme, the weighting function decreases as a function of the angle between $\mathbf{n}_i$ and $\mathbf{n}_{ij}$ resulting in the preservation of sharp features.

Feature preserving mesh decimation

The number of degrees of freedom in the mesh influences the computational burden of subsequent physical simulations. One strategy to reduce the number of degrees of freedom is to perform mesh decimation or simplification.

There are many strategies for decimation, some reviewed here [78, 79], including topology preserving Euler operators, other algorithms such as vertex clustering which may not guarantee topological invariance [80], and remeshing [81]. It is typically desirable to preserve the mesh topology for physical simulation based applications. Conventional Euler operations for mesh decimation include vertex removal, edge collapse, and half-edge collapse. As noted earlier, finite element simulations are sensitive to angles of the mesh. Edge and half-edge collapses can sometimes lead to vertices with high or low valency and therefore poor angles. Although algorithms to detect topology-changing edge collapses have been developed [82], we avoid this problem by employing a

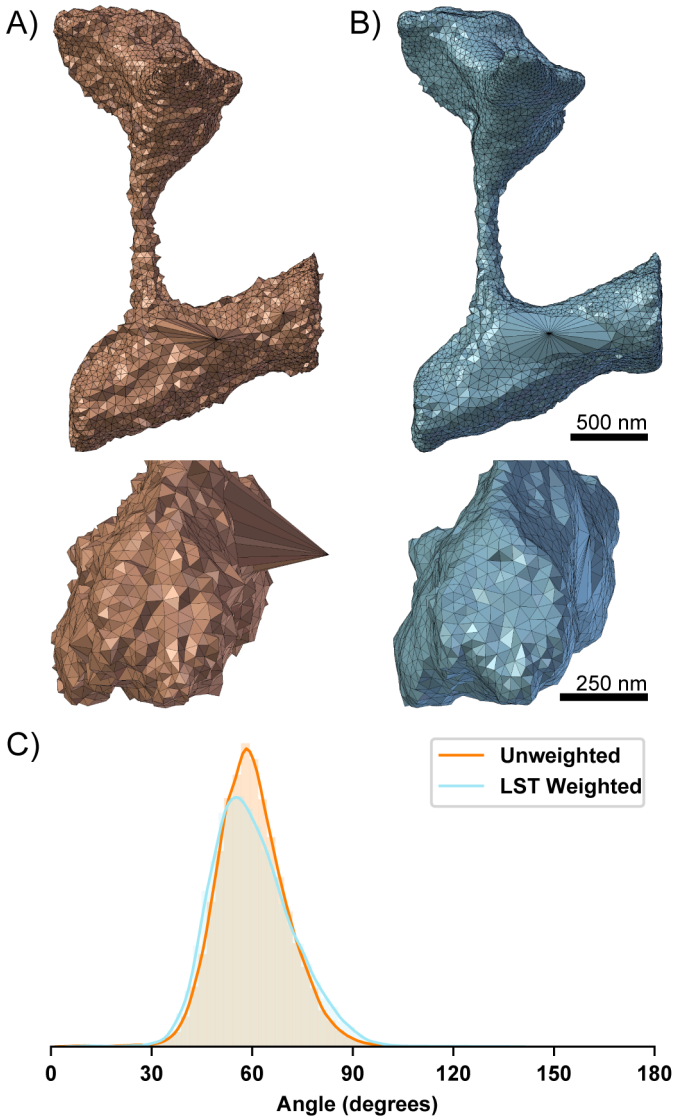

Fig 4. Comparison of 50 iterations of angle weighted smoothing algorithm. A) without and B) with Local Structure Tensor (LST) based correction. The LST helps to preserve the geometric structure albeit with slight degradation to the mesh angles. It is a simple metric to capture local geometric information which can be used to constrain conditioning operations. C) Mesh angles are improved in both the LST weighted and unweighted meshes. The distribution of the mesh angles with LST correction are left shifted from 60 degrees.

vertex removal algorithm. First, vertices to be decimated are selected based upon certain criteria discussed below. We then remove the vertex and re-triangulate the resulting hole. This is achieved using a recursive triangulation approach, which heuristically balances the edge valency. Given the boundary loop, we first connect vertices with the fewest incident edges. This produces two resulting holes that we then fill recursively using the same approach. When a hole contains only three boundary vertices, they are connected to make a face. We note that while this triangulation scheme balances vertex valency, it may degrade mesh quality. We solve this by running the geometry preserving smoothing algorithm on the local region.

We employ two criteria for selecting which vertices to remove. These criteria can be used

in isolation or together. First, to selectively decimate vertices in low or high curvature regions, information from the LST can be used. Comparing the magnitudes of the eigenvalues of the LST provides information about the local geometry near a vertex. For example, to decimate vertices in flat regions of the mesh, given eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3$, vertices can be selected by checking if the local region satisfies,

$$\frac{\lambda_2}{\lambda_1} < R_1, \tag{6}$$

where R_1 is a user specified flatness threshold (smaller is flatter). In a similar fashion, vertices in curved regions can also be selected. However, decimation of curved regions is typically avoided due to the potential for losing geometric information.

Instead, to simplify dense areas of the mesh, we employ an edge length based selection criterion,

$$\frac{\max_{i=1}^{N_1} d(\mathbf{x}, \mathbf{v}_i)}{\overline{D}} < R_2, \tag{7}$$

where N_1 is the number of vertices in the 1-ring neighborhood of vertex $\mathbf{x}$, $d(\cdot,\cdot)$ is the distance between vertices $\mathbf{x}$ and $\mathbf{v}_i$, $\overline{D}$ is the mean edge length of the mesh, and R_2 is a user specified threshold. This criterion allows us to control the sparseness of the mesh. We note that the aforementioned criteria are what is currently implemented in `GAMer 2`, however the vertex removal decimation scheme can be employed with any other selection criteria.

Boundary marking and tetrahedralization

To support the definition of boundary conditions on the mesh, it is conventional to assign boundary marks or identifiers which correspond to different boundary definitions in the physical simulation. In simplified and idealized geometries it is possible to define functions to assign boundary values. However, in subcellular scenes where the geometry may be tortuous, local receptor clusters can be arbitrarily distributed on the manifold, and the resolution may be insufficient to resolve these features, boundary definition is a non-trivial challenge. In many scenarios, the most biologically accurate boundary definition may be based off of a biologist's understanding of the specimen which transcends the particular datset. For example, a particular image may not be able to resolve how receptors are distributed but knowledge of relevant immunogold labeling studies may allow the biologist to propose a physiologically meaningful receptor distribution. The `BlendGAMer` add-on supports the facile user-based definition of boundary markers on the surface [61]. Users can utilize any of the face selection methods (e.g., circle selection demonstrated in Fig. 5) which `Blender` provides to select boundaries to mark. Boolean operations and other geometric strategies provided natively in `Blender` can also be used for selection. Boundary markers are associated with a unique material property which helps visually delineate marked assignments. After boundaries are marked, stacks of surface meshes corresponding to different domains can be grouped and passed from `GAMer 2` into `TetGen` for tetrahedralization [83].

Results

As a demonstration, we apply `GAMer 2` to build simulations from electron micrographs and segmentations from Wu et al. [7] which were graciously shared by De Camilli and coworkers. In their work, Wu et al. imaged dendritic spines from neurons taken from the mouse cerebral cortex or nucleus accumbens using Focused-ion Beam Milling Scanning Electron Microscopy (FIB-SEM) [7]. In addition to their important role in synaptic and structural plasticity, these cellular structures

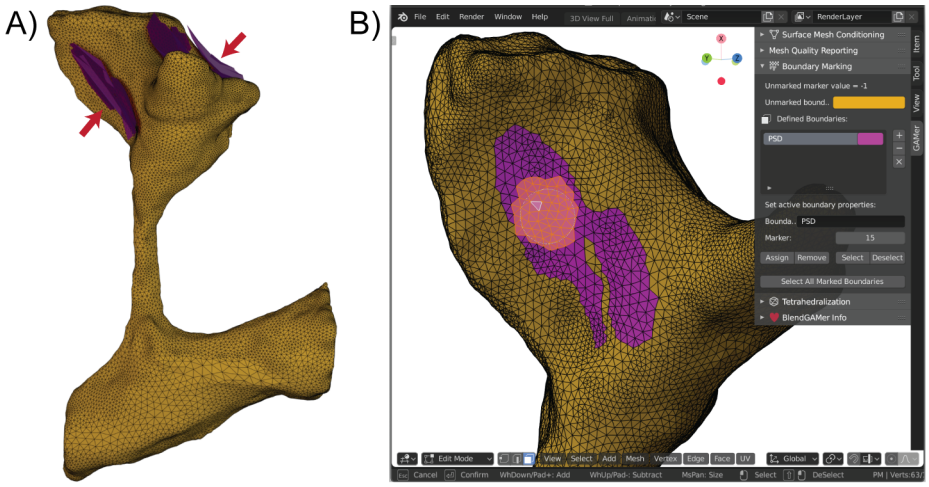

Fig 5. Marking boundaries using `BlendGAMer`. A) The Postsynaptic Density (PSD) is annotated as a contour in the segmentation and represented as a patch (purple) neighboring the plasma membrane (yellow). B) Screenshot of boundary marking using `BlendGAMer` v2.0.5 in `Blender` 2.80. The circle select tool is used to select faces of the plasma membrane mesh in proximity to the Postsynaptic Density (PSD) patches. A user interface allows the user to name boundaries, assign and unassign face membership, along with setting the marker value.

demonstrate highly tortuous morphologies, high surface-to-volume ratios, and a geometric intricacy that serves as a good test-bed for our approach. Here we consider several scenes of increasing length scale: the ER of the single spine geometry which requires nanometer precision and the Plasma Membrane (PM) of the single spine which has a length scale of a couple of microns (Fig. 6A), the two spine geometry, a few microns (Fig. 6B), and the dendrite with about 40 spines, with a length scale in the tens of microns (Fig. 6C). For each of these geometries, using the segmentations produced by Wu et al., we generated preliminary meshes using the `imod2obj` utility included with `IMOD` [34]. The output initial meshes have units of pixels and were scaled to µm using an 8 nm isotropic voxel size. Each initial mesh was then processed using algorithms described in §Mesh Processing and implemented in `GAMer 2` [58]. We note that for some meshes, features such as disconnections of the ER, were manually reconnected using `Blender` mesh sculpting features. We will provide a candid discussion of the manual curation steps in the following paragraphs discussing each scene. Boundaries were marked using `BlendGAMer`, although `PyGAMer` and `GAMer 2` can be scripted to assign boundary labels as well, and the conditioned surface meshes were tetrahedralized using `TetGen` [83].

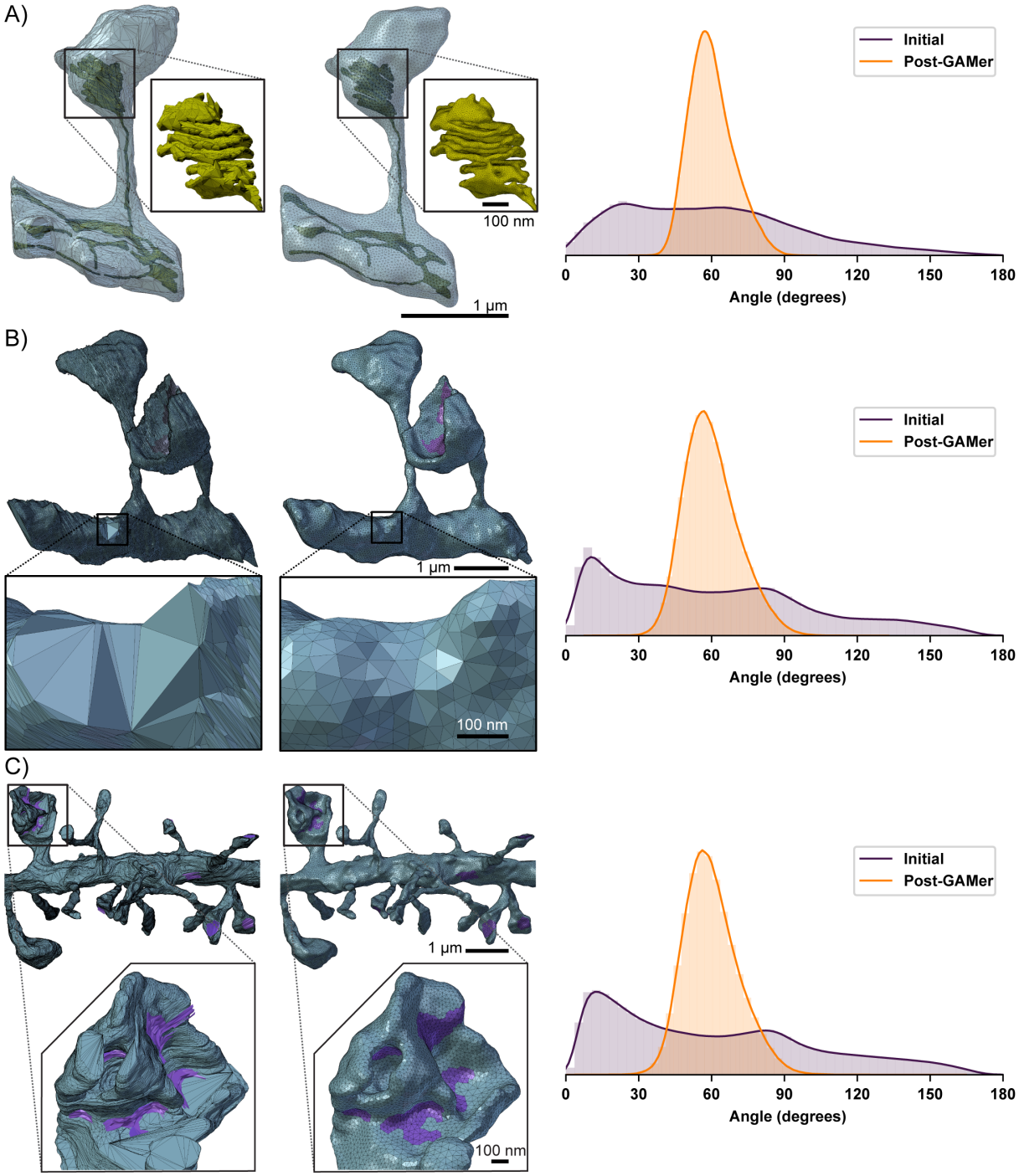
A)
100 nm
1 µm
Initial
Post-GAMer
0
30
60
90
120
150
180
Angle (degrees)
B)
1 µm
100 nm
Initial
Post-GAMer
0
30
60
90
120
150
180
Angle (degrees)
C)
1 µm
100 nm
Initial
Post-GAMer
0
30
60
90
120
150
180
Angle (degrees)

Fig 6. Quantification of mesh quality pre- and post-GAMer 2 processing for several geometries. Data at varying spatial scales can be processed via the GAMer 2 framework. Surface meshes of dendritic spine geometries before (left) are compared with their mesh after GAMer 2 processing (middle). The shift in distribution of angles highlights the improvement in mesh quality (right). A) Surface meshes of a single dendritic spine; the PM is colored cyan and the ER yellow. Inlay: close-up of the spine apparatus. The standard deviation of the distribution of triangular angles before is 35.9 and after 9.1. B) Surface mesh of PM of two dendritic spines. Faces marked as purple are the PSD. Inlay: close-up of a region with a large variance in angle distribution before GAMer 2 processing. The standard deviation of the distribution of triangular angles before is 41.1 and after 11.1. C) Surface mesh of PM of a dendrite segment with many spines. Inlay: GAMer 2 preserves the intricate details of a highly curved spine head with multiple regions of PSD. The standard deviation of the distribution of triangular angles before is 41.9 and after 11.1.

The one spine geometry contains two separate membranes: the PM and the ER Fig. 6A – each with their own problems. The PM contained several large holes on the surface which correspond to the top and bottom of the image stack. As the dendrite meanders throughout the tissue block, the cutting planes of the experiment or sample block may result in the artificial truncation of the image. We have remedied this truncation by triangulating the holes and processing using GAMer 2 algorithms.

The ER mesh contained many more initial artifacts. This is because the detailed and variable nature of the ER membrane can be poorly resolved by the imaging method. Nixon-Abel and coworkers found using superresolution fluorescence microscopy that ER tubules have a diameter of 50 to 100 nm and sheet-like structures at the cellular periphery can be much finer [84]. Some ER morphology cannot even be resolved by the powers of EM [7]. For example, if the ER undergoes large spatial variation between z-slices then tubules may appear disconnected. Alternatively, the boundaries of the ER membrane may have poor contrast and can sometimes be missed during segmentation. We have manually reconnected the ER segments manually in Blender under the guidance of the underlying EM micrographs. An animated comparison of the initial mesh with the stack of images is shown in Movie S2. This type of topological artifact arising from errors in segmentation or poor imaging resolution remains a major challenge to the field. In this work, to produce a model faithful to the biological context, we have employed human judgement to detect and rectify similar problems. As imaging technology and machine recognition algorithms improve, we anticipate that the need for manual curation will be reduced. The decision to manually curate, or not, is at the discretion of the end-user. BlendGAMer provides only the option for the tight integration of sculpting and automated mesh processing.

After curation and processing with GAMer 2, the geometric detail of the one spine scene is preserved. An animated comparison of the meshes output by GAMer 2 is shown in Movie S3. Notably, this spine contains a specialized form of ER termed the spine apparatus, Fig. 6A, inset, which consists of seven folded cisternae. This highly organized structure bears geometrical similarities to a parking garage structure and helicoidal geometries [85–88]. The geometric detail of the spine apparatus is preserved by the conditioning process in our pipeline.

In Fig. 6, we also show the distribution of the triangular angles of the surface mesh before and after conditioning. One metric of a well-conditioned mesh is that all the surface triangles are nearly equilateral [68]. Prior to conditioning, the angle distribution is spread out and contains many large and small angles. After processing using GAMer, the angles of triangles of the one spine PM mesh are improved, as indicated by the peaked distribution around 60 degrees. Although the ER structure is significantly more complex, the angles of triangles of the mesh are also improved, albeit to a lesser extent than the PM. In scenarios such as this where the length scales of interest are closer to the

acquisition resolution, it may be necessary to increase the number of triangles to accurately capture the fine details with high mesh quality. Table 1 summarizes the number of vertices and triangles in the initial vs conditioned meshes as well as vertices and tetrahedra in the resultant volumetric meshes. To accurately capture the curvature of the PM mesh in Fig. 6A about 48% more triangles were needed compared to the ER mesh in Fig. 6A, inset, which required 270% more triangles, both relative to the initial mesh.

The approach described here is also applicable for larger systems as we demonstrate with two spines and a full dendrite. The two spine geometry shown in Fig. 6B is a few microns in length. Based on the length scales we would expect a well conditioned mesh for this geometry to contain approximately double the number of triangles in the single spine mesh; however, the orientation of z-stacks in this mesh is different from that in the single spine geometry which led to an abnormally large number of triangles: 320,976 versus just 9,330 in the mesh of PM in the single spine. After `GAMer 2` conditioning algorithms were applied, the number of triangles was reduced to 36,050, a much more reasonable count. As demonstrated, our pipeline is robust and can handle cases where the initial mesh either generates too few or too many triangles as required for capturing geometric details. The ER of the two spine geometry was processed in a similar manner to that of the one spine.

At the tens of microns length scale, we constructed a mesh of a full dendritic segment. We show a zoomed in section of the mesh before and after conditioning in Fig. 6C. As in the one and two spine cases, our system robustly handles artifacts such as poor quality triangles and intersecting faces; Fig. 6C shows that the distribution of the angles post conditioning are comparable to the one and two spine examples, showing that size does not alter the capability to produce well-conditioned meshes. Fig. 6C shows an intricate spine head with many different regions of PSD shown in purple; this geometry is preserved post-conditioning and the PSD is marked with `BlendGAMer` to denote a boundary condition.

For all meshes the initial surface area is greater than that of the final result (Table 1. This is due to the jagged nature of the initial meshes which reflects small deviations in the alignment and registration of the micrographs and segmentation. As the surfaces are smoothed, the surface area is therefore reduced. On the other hand, the initial and final volumes of each geometry remain similar. This is a good indication of the feature-preserving nature of the algorithms.

In Fig. 7 we compare the meshes generated by `GAMer 2` with other softwares including `TetWild` [40], `CGAL 3D Mesh Generation` (referred to as `CGAL` in the subsequent text) [44–46], Hu et al. Remesh [89], and `VolRoverN` [37]. For this analysis, we performed a best faith effort to use these tools using recommended default settings where possible. We do not make any claims of software supremacy and instead highlight feature differences between the codes. Shown in Fig. 7A are the distribution of triangular angles for the final surface mesh. We find that `GAMer 2` produces a mesh with more equilateral triangles than other tools.

The radius-ratio is a useful metric for determining the quality of both triangles and tetrahedra, it is defined as $\frac{nr_i}{r_o}$ where n is the geometric dimension (i.e. 2 for triangles, 3 for tetrahedra), r_i is the radius of the largest n-sphere which can be inscribed within the shape, and r_o is the radius of the smallest n-sphere which circumscribes the shape. An equilateral triangle and an equilateral tetrahedron both correspond to a radius-ratio of one [90]. As the radius-ratio approaches zero the element approaches degeneracy which can affect numerical accuracy, stability, and convergence [66, 67].

Fig. 7B compares different codes and their resultant distributions of radius-ratios when applied to the single spine PM surface mesh. One important difference is that `TetWild`, `CGAL`, and Hu et al. Remesh are fully automated algorithms which employ constraints to guarantee preservation of input features. We note that all methods tested here, except `TetWild`, require a watertight and manifold surface mesh as input. For the sake of this comparison, the same manually curated watertight and

Table 1. Vertex and Element Counts for Meshed Geometries.

		Surface Mesh			Volume Mesh*		
		# Vertices	# Triangles	Area [μm^2]	# Vertices	# Tetrahedra	Volume [$5\mu m^3$]
Single Spine	Initial PM	4,695	9,330	6.99	6,726	27,581	0.64
	Conditioned PM	6,924	13,844	6.52	13,734	62,557	0.65
	Initial ER	6,546	19,654	2.70	–	–	0.028*
	Conditioned ER	36,294	72,620	2.39	53,134	211,018	0.027
Two Spines	Initial PM	160,733	320,976	26.09	–	–	2.82*
	Conditioned PM	18,027	36,050	23.58	28,989	122,082	2.94
	Initial ER	20,111	40,370	10.90	–	–	0.13*
	Conditioned ER	80,419	161,050	9.73	101,033	352,741	0.13
Dendritic Segment	Initial PM	207,448	410,896	139.1	–	–	10.403*
	Conditioned PM	126,336	252,668	110.0	194,848	798,626	10.611

*Non-manifold and other mesh artifacts prevent the tetrahedralization of these meshes. Volumes reported are computed using the corresponding surface mesh.

manifold meshes were used as input for all methods. Noisy features such as the jagged boundaries resulting from misaligned micrographs or segmentation are often preserved in the surface meshes generated by these codes. As a result, the triangular angles often deviate from equilateral in order to represent these preserved fine features. `GAMer 2` on the other hand does not strictly preserve the fine features of the input mesh. The LST is a metric of the local curvature averaged over a mesh patch. Thus, `GAMer 2` generated meshes are smoother but deviate, from the input, more than those generated by `TetWild`, `CGAL`, and Hu et al. Remesh. `VolRoverN`, on the other hand, implements the Level Set Boundary-Interior-Exterior (LBIE) algorithm which employs a geometric flow. The LBIE algorithm is known to work well for spherical geometries and was designed for smoothing biomolecular meshes constructed as the union of hard spheres [37, 91, 92]. Given the complex geometry of the dendritic spine, `VolRoverN` does not preserve the geometry well.

The tetrahedral mesh qualities of meshes produced by each code are compared in Fig. 7C and D. Notably, no volume meshes are shown for Hu et al. Remesh and `VolRoverN` since Hu et al. is a surface remeshing code only and `VolRoverN` failed to produce a valid volume mesh when called from the software's graphical user interface. The other codes `TetWild`, `CGAL`, and `GAMer 2` all produced quality tetrahedral meshes as output. The distribution of tetrahedral radius-ratios are shown in Fig. 7D. We find that `CGAL` under-performs in comparison to the other two codes. This observation may be due, in part, to our usage of `CGAL`. The 3D mesh generation subproject of `CGAL` is provided as a library along with several example C++ scripts. For this comparison, we have applied the bundled script to our mesh of interest with minimal modification. We note that the default settings in the script may not be ideal for our application. It is possible that by setting stricter mesh quality targets for optimization, `CGAL` could perform better.

GAMer 2 Operations Preserve Mesh Topology

As aforementioned, where we deemed appropriate, manual curation was used to modify the topology of the input meshes. In order to differentiate between the manual modifications and to verify whether `GAMer 2` operations change the topology of the mesh we have computed Betti numbers at several mesh processing stages. Betti numbers are topological invariants and thus can be used to distinguish between topological spaces. The kth Betti number β_k describes the number of k-dimensional holes on a surface. Intuitively β_0 is the number of connected components, β_1 is the number of circular holes, and β_2 is the number of enclosed voids.

The computation of Betti numbers is performed using a modified algorithm by Definado-Edelsbrunner [93]. In their original work, they describe an algorithm to compute the Betti numbers of tetrahedral meshes by iterating over a filtration. Since `GAMer 2` is primarily a surface mesh conditioning library, there are no tetrahedra and therefore the Delfinado-Edelsbrunner algorithm is not applicable. The challenge lies in the determination of when the addition of a triangular face (i.e., 2-simplex) to a filtration forms an enclosed volume. In the original algorithm, the tetrahedral simplices are colored such that all adjacent tetrahedra not separated by a closed boundary will be the same color. The addition of a triangular face in a filtration will produce a void if and only if it completes a boundary such that the interior and exterior tetrahedra can be colored differently. While algorithms such as flood-filling or ray-casting and counting surface crossings have been described, these approaches are often computationally cumbersome especially when applied to meshes with reentrant surfaces.

We simplify the problem at hand by restricting our calculation to apply for only the set of topologically manifold surface meshes. If a mesh is both orientable (i.e., a consistent normal orientation can be assigned for all faces such that no neighboring faces have opposing normals) and all edges belong to two faces, we say that the mesh is a closed manifold. After iterating through a

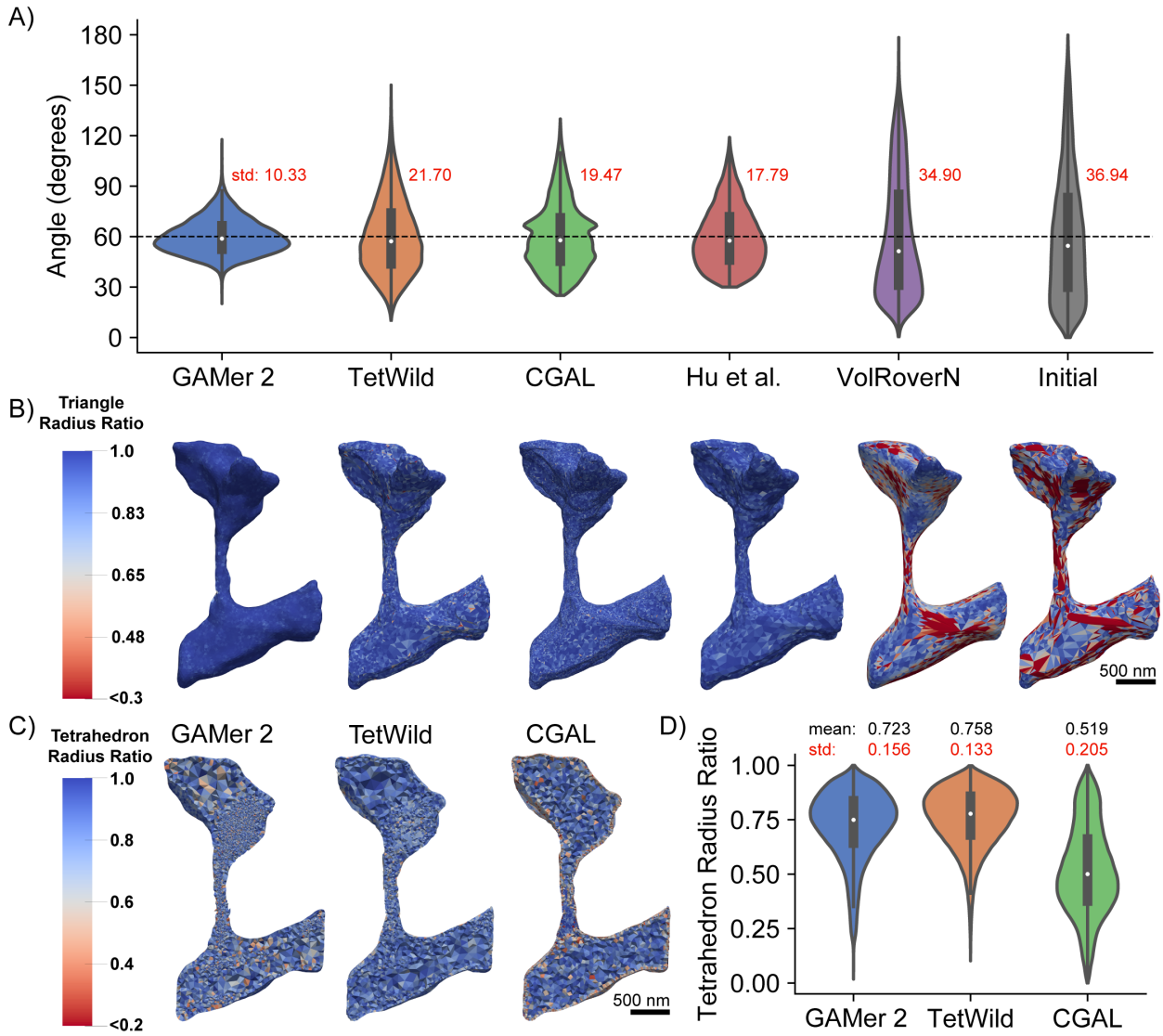

Fig 7. Comparison of `GAMer 2` generated meshes with leading mesh generation softwares. A) Distribution of triangular angles of the final Plasma Membrane (PM) and Endoplasmic Reticulum (ER) surface meshes. B) Quantification of the surface mesh triangle radius-ratios. The order of meshes is identical to the software ordering in panel A. C) Quantification of tetrahedron radius-ratios of the tetrahedral mesh output by each software*. We note that the tetrahedral mesh produced by `GAMer 2` is generated using `TetGen` from a `GAMer 2` conditioned surface mesh. D) Distribution of tetrahedron radius-ratios of the resulting tetrahedral meshes.
*In our best faith effort, we were not able to tetrahedralize the PM and ER mesh using `VolRoverN`. The tetrahedral meshes for `GAMer 2` and `TetWild` include the ER; `CGAL` contains only the PM–we were not able to tetrahedralize both the PM and ER together.

filtration in the manner described by Delfinado-Edelsbrunner, if we find that a mesh is a closed manifold we increment the first and second Betti numbers (β_1, β_2) by one.

Betti numbers of the meshes at several processing stages, produced by the modified Delfinado-Edelsbrunner algorithm, are computed and shown in Table 2. As aforementioned, the initial meshes produced by `imod2obj` often contain disjoint surfaces or holes. Comparing against the underlying

micrographs and data, we have curated each mesh to produce a watertight model. At this point we apply `GAMer 2` meshing operations to produce a conditioned mesh. We find that the topology of the watertight meshes are identical to that of their corresponding conditioned mesh.

Table 2. Computed Betti numbers for each mesh.

Geometry	**Component**	β_0	β_1	β_2
Single Spine	Initial PM†	1	4	0
	Watertight PM*	1	0	1
	Conditioned PM	1	0	1
	Initial ER†	6	18	6
	Watertight ER*	1	18	1
	Conditioned ER	1	18	1
Two Spines	Initial PM†	1	0	1
	Watertight PM*	1	0	1
	Conditioned PM	1	0	1
	Initial ER†	3	110	3
	Watertight ER*	1	110	1
	Conditioned ER	1	110	1
Dendritic Segment	Initial PM†	45	62	0
	Watertight PM*	1	0	1
	Conditioned PM	1	0	1

† Betti number computation in `GAMer 2` only supports manifold surface meshes. Initial meshes have been curated in a best faith effort to preserve the initial topology (i.e., edges connected to three or more faces are cleaned up while holes and disconnected components are untouched)

* These meshes have been edited to be watertight. Furthermore disconnected components which are artifacts of earlier workflow stpes have been reconnected.

Estimating Membrane Curvatures in Realistic Geometries

In addition to the generation of simulation compatible meshes of realistic cell geometries, the meshes from `GAMer 2` are amenable to other geometric analysis. The conditioned meshes can yield improved results for many geometric quantities of interest such as surface area and volume along with other more complex observables such as surface curvature. Membrane curvatures and minimal surfaces have long been of interest to biophysicists and mathematicians alike.

One of the advantages of using electron micrographs of membrane structures in cells is that we can now bridge the gap between membrane mechanics, curvature studies, and realistic geometries. Current studies of membrane mechanics often assume that the initial membrane configuration is flat or spherical. However membranes are rarely so well behaved and to the best of our knowledge, currently no estimates of the curvatures of the plasma membrane or internal membranes exist.

Using the conditioned surface meshes, the curvature can be estimated using methods from discrete

differential geometry. In `GAMer 2`, we have implemented the algorithms to compute curvatures as described by Cazals and Pouget (JETS) [94, 95] and Meyer et al. (MDSB) [96]. We note that the curvature values produced by `GAMer 2` are estimates suitable for qualitative comparison and visualization only. The JETS algorithm fits an osculating jet to a local patch using interpolation or approximation. From this fitted jet, the curvature is calculated. Depending on the fineness of the mesh, the jet order, and the details of patch selection, the error in the curvature may vary. On the other hand, Borelli et al. has previously showed that the estimation of the Gaussian curvature using the angle defect, which is used by the MDSB algorithm, is valid only for regular meshes with a vertex valency of 6 [97]. The robust calculation of curvature estimates from discrete meshes remains an open problem and the accuracy of many approaches are surveyed by Váša et al. in Ref. 98.

Here, we use the meshes produced by `GAMer 2` to calculate the curvature of the geometries using the MDSB algorithm. The principal curvatures κ_1 and κ_2 are shown in Figs. 8 and 9, respectively. We note that we have adopted the sign convention where a negative curvature corresponds to a convex region. A comparison of the MDSB estimate and JETS is shown in Fig. S1. We find that both algorithms produce qualitatively similar results.

Looking at the first principal curvature, Fig. 8, corresponding to the maximum curvature of the local region, we find that the distribution of κ_1 spans both positive and negative values, centered around zero for both the PM and ER. The negative regions of κ_1 are in regions where the membrane is convex and the positive regions are in regions where the membranes are concave.

The second principal curvature, which corresponds to the minimum curvature of the local region, shows a different behavior (Fig. 9). We first observe that for all geometries, this value is primarily negative. The regions of high bending such as the folds of the spine apparatus in the spine head (Fig. 9A, B, left panels) are highly curved and are connected by sheets with low curvature. The curvature along the entire dendrite highlights that the structure is mostly characterized by low curvature throughout with regions of concentrated high curvature (Fig. 9C). The positive regions of κ_2 are in regions where the membrane is convex and the negative regions are in regions where the membranes are concave. Thus using the meshes generated from `GAMer 2`, we are able to quantify the curvatures along the plasma membrane and the internal organelle membranes using tools from discrete differential geometry. In addition to estimating the curvatures, we can use the mesh models to interrogate the impacts of curvature on signaling.

Simulations of a Coupled Volume and Surface Diffusion Model

To showcase how simulations performed on meshes of realistic biological geometries can elucidate structure-function relationships, we reproduce the results of Ref. 99 on a dendritic spine. The one spine geometry was used as the spatial domain for a numerical simulation using the finite element method. Consider the reaction,

$$\mathrm{A} + \mathrm{X} \underset{k_{\mathrm{off}}}{\overset{k_{\mathrm{on}}}{\rightleftharpoons}} \mathrm{B},$$

where A is a cytosolic component which binds to X, a membrane bound component, to produce B, another membrane bound component. The governing equations consist of a volumetric Partial Differential Equation (PDE),

$$\frac{\partial \mathrm{A}}{\partial t} = D_{\mathrm{A}} \Delta \mathrm{A} \quad \text{in} \quad \Omega, \tag{8}$$

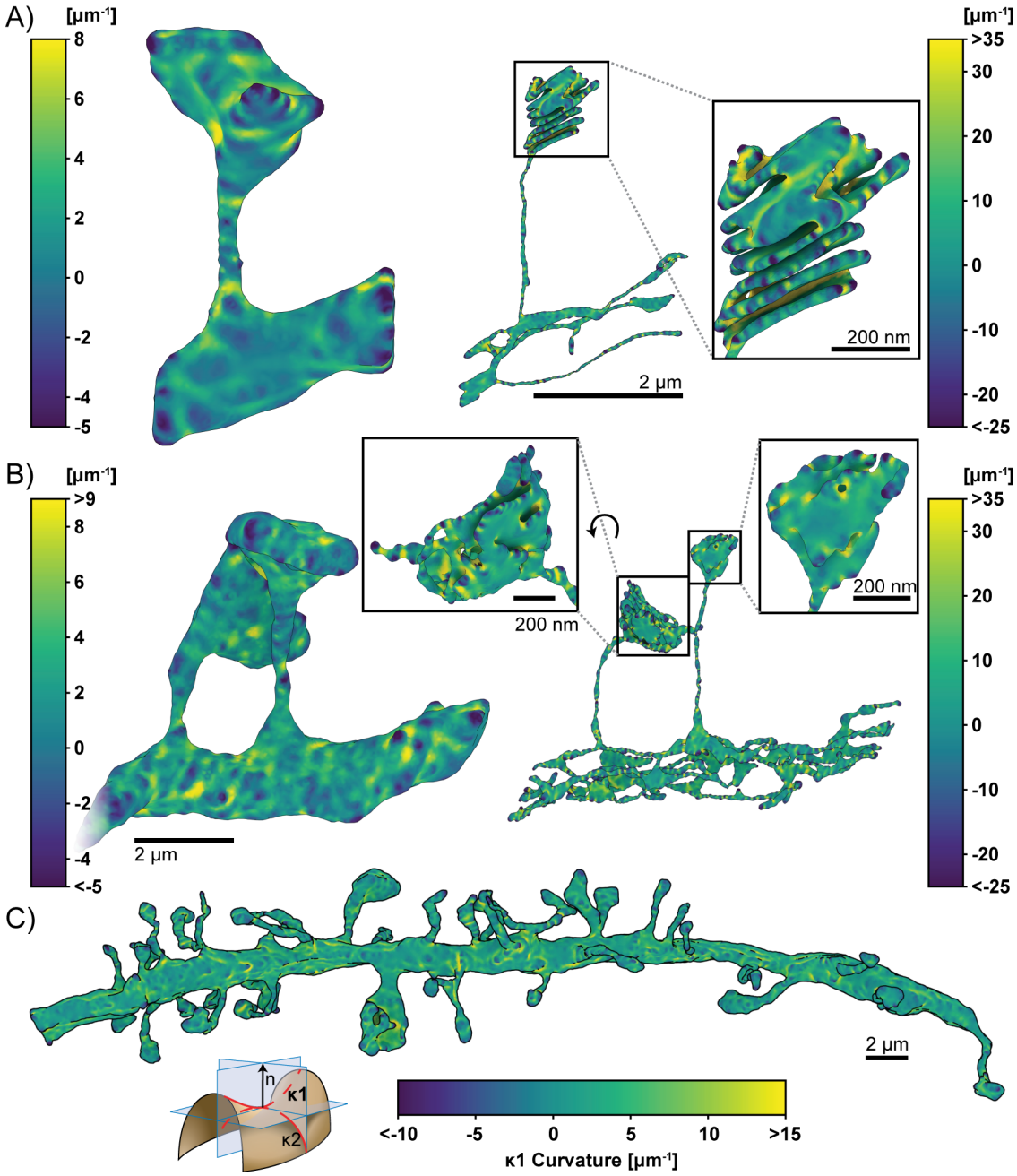

Fig 8. Estimated first principal curvatures of the spine geometries. The signed first principal curvature, corresponding to the maximum curvature at each mesh point, is estimated using `GAMer 2`. Color bars correspond to curvature values with units of μm^{-1}. We have adopted the sign convention where negative curvature values refer to convex regions. Geometries are A) single spine model, left: plasma membrane, right: endoplasmic reticulum; B) two spine model, left: plasma membrane, right: endoplasmic reticulum; and C) plasma membrane of the dendritic branch model. Scale bars: full geometries $2\,\mu$m, inlays: 200 nm. Curvature schematic modified from Wikipedia, credited to Eric Gaba (CC BY-SA 3.0).

two surface PDEs,

$$\frac{\partial \mathrm{X}}{\partial t} = D_{\mathrm{X}}\Delta_S \mathrm{X} - k_{\mathrm{on}}\mathrm{A}|_{\partial\Omega}\mathrm{X} + k_{\mathrm{off}}\mathrm{B} \quad \text{on} \quad \partial\Omega \tag{9}$$

$$\frac{\partial \mathrm{B}}{\partial t} = D_{\mathrm{B}}\Delta_S \mathrm{B} + k_{\mathrm{on}}\mathrm{A}|_{\partial\Omega}\mathrm{X} - k_{\mathrm{off}}\mathrm{B} \quad \text{on} \quad \partial\Omega, \tag{10}$$

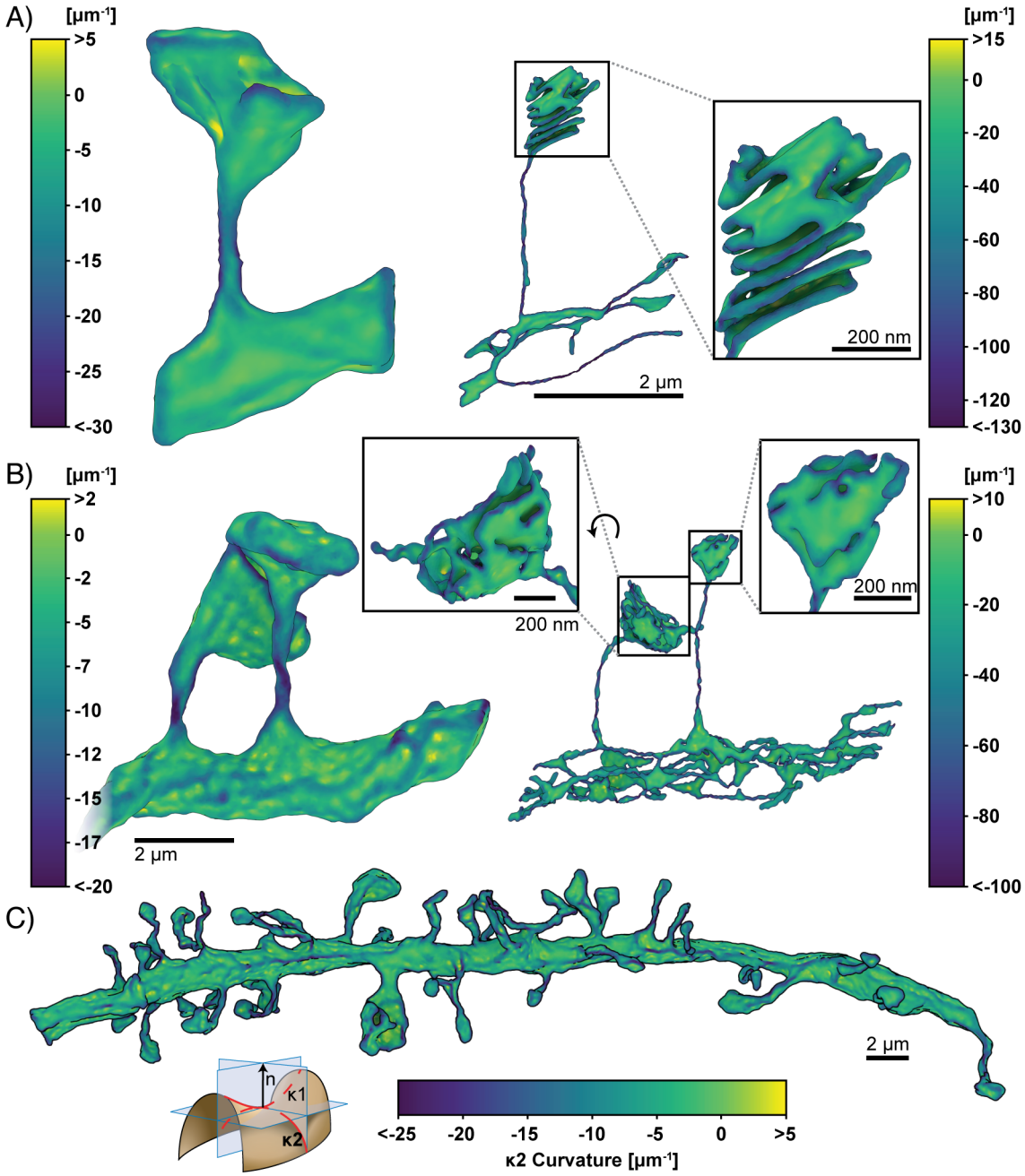

Fig 9. Estimated second principal curvatures of the spine geometries. The signed second principal curvature, corresponding to the minimum curvature at each mesh point, is estimated using `GAMer 2`. Color bars correspond to curvature values with units of μm^{-1}. We have adopted the sign convention where negative curvature values refer to convex regions. Geometries are A) single spine model, left: plasma membrane, right: endoplasmic reticulum; B) two spine model, left: plasma membrane, right: endoplasmic reticulum; and C) plasma membrane of the dendritic branch model. Scale bars: full geometries $2\,\mu$m, inlays: 200 nm. Curvature schematic modified from Wikipedia, credited to Eric Gaba (CC BY-SA 3.0).

and a boundary condition for A which couples all three species at the interface:

$$D_{\text{A}}(\mathbf{n}\cdot\nabla\text{A}) = -k_{\text{on}}\text{AX} + k_{\text{off}}\text{B} \quad \text{on} \quad \partial\Omega. \tag{11}$$

D_{A}, D_{X}, and D_{B} are the diffusion coefficients for A, X, and B respectively. $\mathbf{n}$ is the outwardly-oriented unit normal vector, Δ is the standard Laplacian operator, Δ_S is the Laplace-Beltrami operator, Ω is the volumetric (cytosolic) domain, and $\partial\Omega$ is the surface (plasma membrane) domain (illustrated in Fig. 10A). The parameters used in this system are as follows: $k_{\mathrm{on}} = 1\,\mu\mathrm{M}^{-1}\,\mathrm{s}^{-1}$, $k_{\mathrm{off}} = 0.1\,\mathrm{s}^{-1}$, $D_{\mathrm{A}} = 0.1\ldots300\,\mu\mathrm{m}^2\,\mathrm{s}^{-1}$, $D_{\mathrm{X}} = 0.1\,\mu\mathrm{m}^2\,\mathrm{s}^{-1}$, and $D_{\mathrm{B}} = 0.01\,\mu\mathrm{m}^2\,\mathrm{s}^{-1}$. The initial conditions were set to $\mathrm{A}(t=0) = 1.0\,\mu\mathrm{M}$, $\mathrm{X}(t=0) = 1000\,\mathrm{molecules}\,\mu\mathrm{m}^{-2}$, and $\mathrm{B}(t=0) = 0\,\mathrm{molecules}\,\mu\mathrm{m}^{-2}$. The initial concentration of X was set to a large value such that it would not be a rate-limiting factor.

Multiplying each PDE by a test function, integrating over their respective domains, and applying the divergence theorem results in the variational or weak form of the problem. After discretizing the time derivatives using the backward Euler method with time-step size δt, and decoupling the volumetric and surface PDEs using a first-order operator splitting scheme the system becomes:

$$\int_{\Omega} \frac{\mathrm{A}^{(n+1)} - \mathrm{A}^{(n)}}{\delta t} v_{\mathrm{A}} + D_{\mathrm{A}} \nabla \mathrm{A}^{(n+1)} \cdot \nabla v_{\mathrm{A}}\, d\Omega + \int_{\partial\Omega} k_{\mathrm{on}} \mathrm{A}^{(n+1)} \tilde{\mathrm{X}} v_{\mathrm{A}} - k_{\mathrm{off}} \tilde{\mathrm{B}} v_{\mathrm{A}}\, d\Gamma = 0, \tag{12}$$

$$\int_{\partial\Omega} \frac{\mathrm{X}^{(n+1)} - \mathrm{X}^{(n)}}{\delta t} v_{\mathrm{X}} + D_{\mathrm{X}} \nabla_S \mathrm{X}^{(n+1)} \cdot \nabla_S v_{\mathrm{X}} + k_{\mathrm{on}} \tilde{\mathrm{A}} \mathrm{X}^{(n+1)} v_{\mathrm{X}} - k_{\mathrm{off}} \mathrm{B}^{(n+1)} v_{\mathrm{X}}\, d\Gamma = 0, \tag{13}$$

$$\int_{\partial\Omega} \frac{\mathrm{B}^{(n+1)} - \mathrm{B}^{(n)}}{\delta t} v_{\mathrm{B}} + D_{\mathrm{B}} \nabla_S \mathrm{B}^{(n+1)} \cdot \nabla_S v_{\mathrm{B}} - k_{\mathrm{on}} \tilde{\mathrm{A}} \mathrm{X}^{(n+1)} v_{\mathrm{B}} + k_{\mathrm{off}} \mathrm{B}^{(n+1)} v_{\mathrm{B}}\, d\Gamma = 0. \tag{14}$$

Here, $\tilde{\mathrm{A}}$, $\tilde{\mathrm{X}}$, and $\tilde{\mathrm{B}}$ represent the most recent estimates of $\mathrm{A}^{(n+1)}$, $\mathrm{X}^{(n+1)}$, and $\mathrm{B}^{(n+1)}$. At each time-step Eq. (12) is solved to estimate $\mathrm{A}^{(n+1)}$, this estimate is then used in Eqs. (13) and (14) to obtain an estimate for $\mathrm{X}^{(n+1)}$ and $\mathrm{B}^{(n+1)}$ which are used again in Eq. (12) to further improve the estimate of $\mathrm{A}^{(n+1)}$. This cycle continues until a satisfactory convergence criterion is met.

Note that Eqs. (9) and (10) are PDEs that govern phenomena occurring entirely on the surface $\partial\Omega$, with the Laplace-Beltrami operator Δ_S appearing as the principle spatial differential operator acting on functions that live on the surface $\partial\Omega$. The *metric* γ_{ij} of the surface $\partial\Omega$ encodes the geometry of the surface, and appears as a spatially varying function in the Laplace-Beltrami operator, as well as in the area element of integrals over $\partial\Omega$. This class of PDEs with spatial domains being represented by two-dimensional surfaces, or more generally Riemannian n-manifolds, are known as *geometric PDEs*, and arise in a number of areas of pure mathematics and mathematical physics, as well as in applications in science and engineering. Unfortunately, two distinct challenges arise in developing numerical methods for geometric PDEs with the necessary convergence properties to allow for drawing scientific conclusions from simulations. The first is the necessity of treating the continuous curved spatial manifold only approximately, using some type of computable discrete proxy (such as an interpolatory triangulation of a smooth two-surface), and then accounting for the impact of this domain approximation on the overall error in a numerical simulation. The second difficulty is the need to approximate the metric of the smooth surface that appears in the definition of the Laplace-Beltrami operator itself, using some type of computable approximation (such as a polynomial), and again accounting for the impact of this second distinct approximation on the overall error in the numerical simulation of phenomena on the surface.

Surface finite element methods have emerged [100–103] over the last few years as an approach to developing finite element methods for this class of problems that have well-understood convergence properties, and are both efficient and reliable. The method is formulated on a "flat" triangulated approximation of the curved domain surface, and the error produced by this approximation is then controlled using a "variational crimes" framework known as the *Strang Lemmas*. Our recent work in this area leverages the Finite Element Exterior Calculus framework [104] to provide a more general error analysis framework for surface finite element methods on n-surfaces, for static

linear and nonlinear problems [105, 106], as well as for evolution problems on surfaces [107, 108]. Surface finite element methods for geometric PDE have the advantage of allowing for the use of standard finite element software originally developed for standard (non-geometric) PDE problems in two-dimensional "flat" domains or three-dimensional volumes, after a fairly simple modification to the reference element maps commonly used by such software packages. Our approach here is to make use of the standard finite element software package `FEniCS` [109], and to use surface finite element modifications to `FEniCS` (described e.g. in [14]) for solving our geometric PDE Eqs. (9) and (10) above.

To demonstrate the role of membrane shape in coupled reaction-diffusion simulations of membrane and volume components, we simulated the reaction of a volume component A reacting with membrane bound species X to form membrane bound species B. The volumetric domain, Ω, and the boundary domain, $\partial\Omega$, are labeled in Fig. 10A. Shown in Fig. 10B and C, are the concentrations of species A and B in the volume and on the surface respectively. We found that the shape of the dendrite has a significant effect on the formation of B on the surface and on the depletion of A in the volume. In regions of high curvature, such as the small protrusions in the head, we found that the density of B is lower because of local depletion of A. This effect can be seen very clearly along the spine neck, where the surface area to volume ratio is high and the resulting density of B is lower than in the dendrite. To investigate if the diffusion coefficient of A affects these results, we varied the diffusion of A and analyzed its effects on the surface distribution of B. As expected, we found that as the diffusion coefficient of A is increased, the effects of local depletion are weakened (Fig. 10D). Fig. 10E shows the maximum and minimum concentrations of B plotted with respect to time. We find that there is a large initial difference in rates of B formation, as indicated by the large gap between the maximum and minimum concentrations, subsequently this gap narrows as A is slowly depleted from the volume. Thus, we show that the meshes generated from `GAMer 2` can be used for systems biology with coupled surface-volume interactions in realistic geometries.

Mesh convergence analysis

Next, we illustrate the effects of `GAMer 2` mesh conditioning on FEA results for the model described above. We investigate the performance improvement as a function of rounds of conditioning. A common error metric used in FEA is the L_2 norm of the difference between a solution computed on a coarser mesh (u') and a solution computed on a very fine mesh, which is taken to be the ground-truth (u), i.e.,

$$\varepsilon_{L_2} = \left(\int_{\Omega} (u' - u)^2 \, d\Omega \right)^{\frac{1}{2}}. \tag{15}$$

This is a standard procedure that can be used to measure h-refinement convergence rates; however between iterations of `GAMer 2` algorithms the boundaries of the mesh are perturbed slightly. Attempting to use ε_{L_2} as an error metric is problematic as its integrand is undefined in regions where Ω', the domain of u', and Ω do not intersect.

Therefore, to illustrate the convergence of solutions as the mesh quality is improved using `GAMer 2`, we used an error metric based on the relative difference in total molecules,

$$\varepsilon_{\text{rel}} = \left| \frac{\int_{\Omega'} u' \, d\Omega' - \int_{\Omega} u \, d\Omega}{\int_{\Omega} u \, d\Omega} \right|. \tag{16}$$

Fig. 11 (D-G) shows tetrahedral meshes generated at intermediary steps during the `GAMer 2` refinement process. For each mesh, a given number of surface mesh smoothing iterations was performed; any remaining artifacts that would prevent tetrahedralization such as intersecting

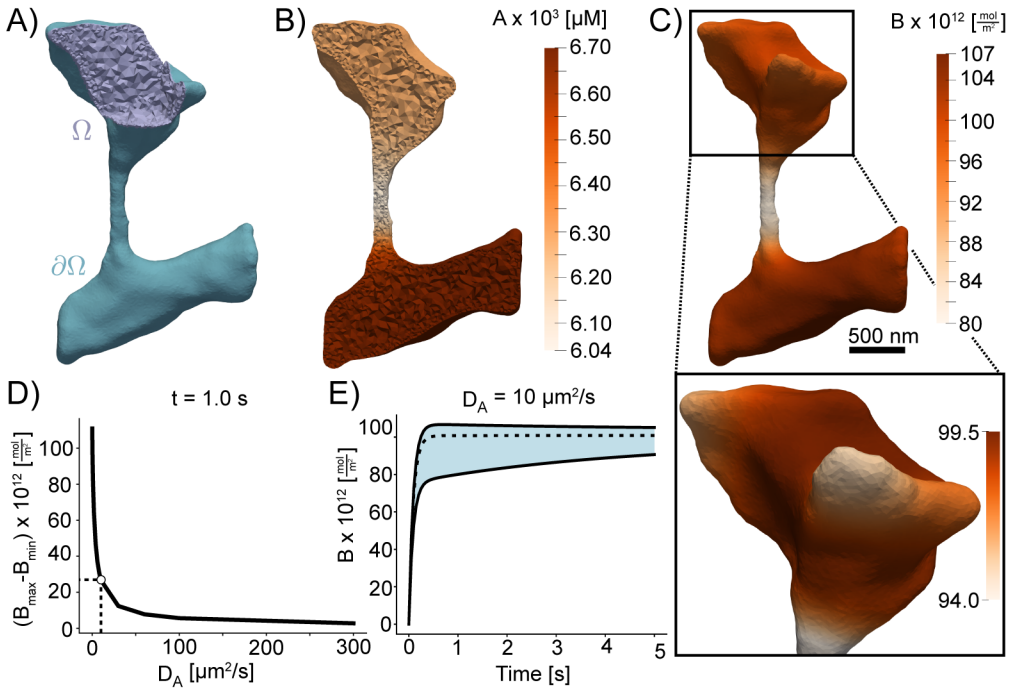

Fig 10. Simulations of coupled surface volume diffusion A) Illustration of the domains for the volume and surface PDEs. B) and C) The concentrations of species A and B, respectively, at $t = 1.0\,\mathrm{s}$ when D_A is set to $10\,\mu\mathrm{m}^2/\mathrm{s}$. D) Difference between maximum and minimum values of B at $t = 1.0\,\mathrm{s}$; the point $D_A = 10\,\mu\mathrm{m}^2/\mathrm{s}$ corresponding to panels B) and C) is marked. E) the minimum, mean, and maximum of B over time when $D_A = 10\,\mu\mathrm{m}^2/\mathrm{s}$; a vertical bar is drawn at $t = 1.0s$. Scale bar: 500nm.

faces were removed and the resultant holes were re-triangulated. The surface meshes were all tetrahedralized using `TetGen` with the same parameters. As the surface mesh Fig. 11 A and B show how as the distribution of surface mesh angles improves, the distribution of radius-ratios in the corresponding tetrahedral mesh improves. The system Eqs. (8) to (11) was solved on the aforementioned tetrahedral meshes for a single time-step and the relative error for B was computed using Eq. (16). The simulation on the most refined mesh (Fig. 11G) was assumed to be the ground-truth. Fig. 11C shows that the relative error consistently decreased as a result of further mesh conditioning in `GAMer 2`. This analysis highlights not only the importance of using a high quality mesh in FEA but also that `GAMer 2` can generate such high quality meshes.

Simulation of reaction-diffusion equations on a dendrite

Using the mesh of the dendritic segment we simulated N-methyl-D-aspartate Receptor (NMDAR) activation due to a Back Propagating Action Potential (BPAP) and Excitatory Postsynaptic Potential (EPSP) along the entire dendrite shown in Fig. 12 and Movie S4. Because the goal of this simulation was not to show biological accuracy, but rather to demonstrate that our approach is capable of producing biophysically relevant FEA simulations, we use a simplified version of the model found in Bell et al. [110].

We model a BPAP and EPSP which stimulates NMDAR opening and calcium ion influx into the

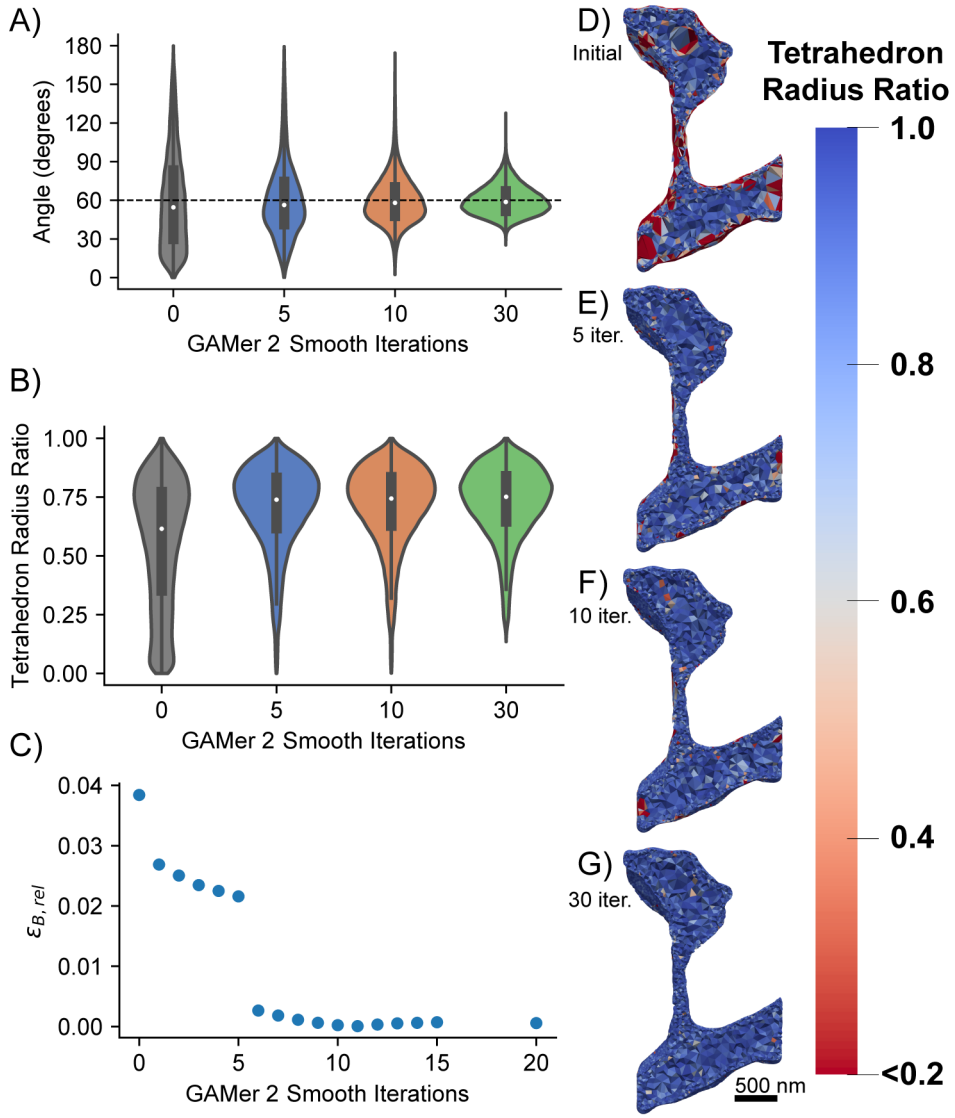

Fig 11. `GAMer 2` mesh conditioning reduces error in the simulated result. A) Distribution of angles on the surface mesh after 0, 5, 10, 30 smoothing operations. B) Distribution of tetrahedral radius-ratios*. The tetrahedron radius-ratio is defined as $\frac{3r_i}{r_o}$ where r_i is the radius of the inscribed sphere and r_o is the radius of the circumsphere (a value of 1 corresponds to an equilateral tetrahedron). C) Relative error (Eq. (16)) of B when solving Eqs. (8) to (10) for a single time step. (D-G) Tetrahedral radius-ratios after 0, 5, 10, 30 smoothing iterations. Generally, for simulation using the finite element method, most radius-ratios should be greater than $\frac{1}{3}$ [90].
*Artifacts which prevented tetrahedralization, e.g. intersecting faces, were removed and the holes were remeshed at each step.

cell. The reaction-diffusion of u, corresponding to calcium ion concentration, is described as follows,

$$\frac{\partial u}{\partial t} = D\Delta u - \frac{u}{\tau} \quad \text{in} \quad \Omega, \tag{17}$$

where D is the diffusion coefficient of u, Δ is the Laplacian operator, τ is a characteristic decay time, and Ω is the volumetric domain. We define boundary conditions corresponding to the ionic

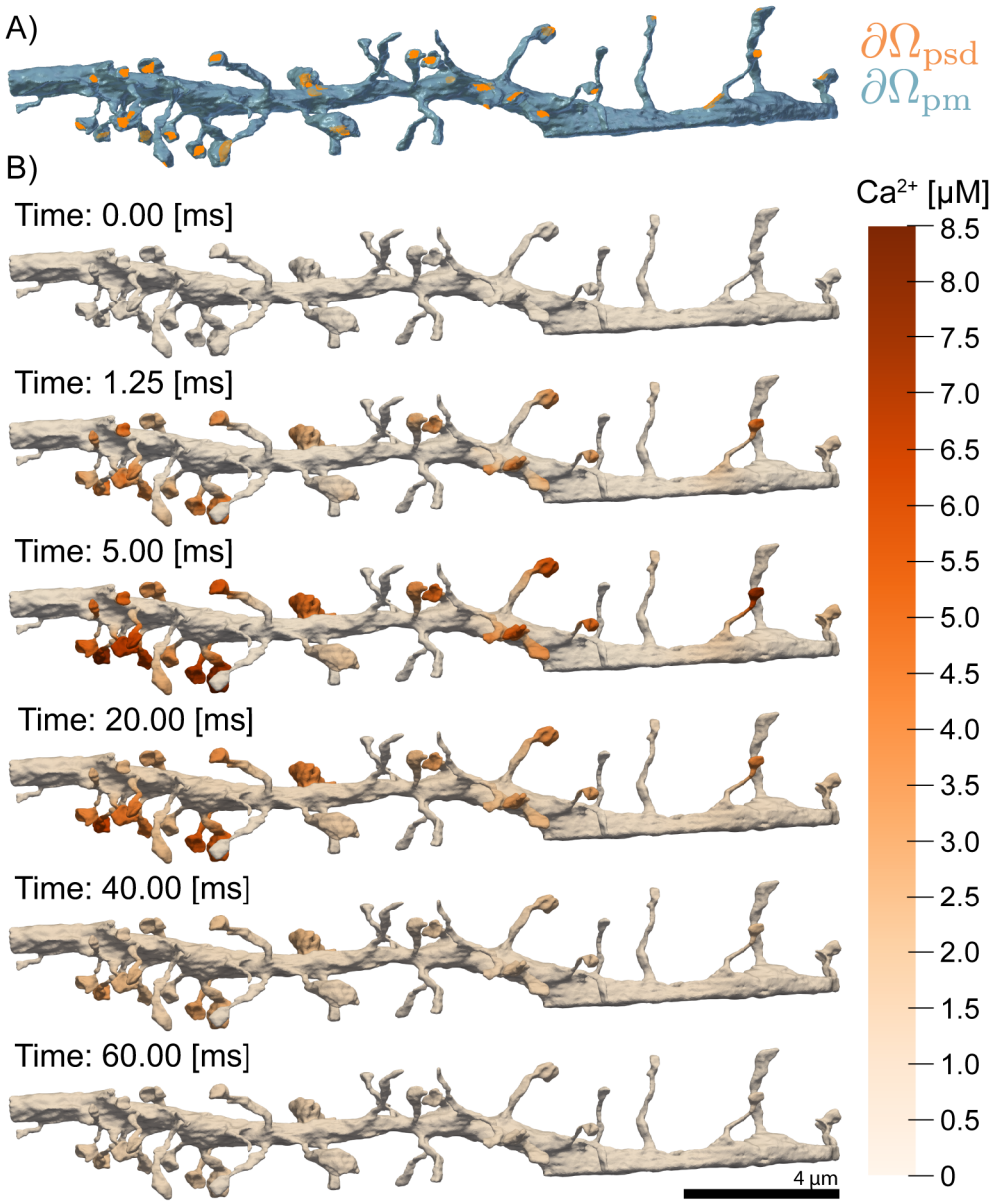

Fig 12. Time series of calcium dynamics from *N*-methyl-D-aspartate Receptor (NMDAR) opening in response to a prescribed membrane voltage trace in a full dendritic segment. Boundaries demarcating the Plasma Membrane (PM) and Postsynaptic Density (PSD) are shown in blue and orange respectively (top). Snapshots of calcium ion concentration throughout the domain are also shown for several time points. We apply a voltage corresponding to a back propagating action potential and excitatory postsynaptic potential. NMDAR localized at the PSD membrane, open in response to the voltage and calcium flows into the cell. Over time, the NMDAR close, and calcium is scavenged by calcium buffers.

flux through NMDARs, J_{NMDAR}, lining the post synaptic density, $\partial\Omega_{\text{psd}}$,

$$D(\mathbf{n} \cdot \nabla u) = J_{\text{NMDAR}}(t) \quad \text{on} \quad \partial\Omega_{\text{psd}}, \tag{18}$$

where $\mathbf{n}$ is the outwardly-oriented unit normal vector, and J_{NMDAR} is of the form,

$$J_{\text{NMDAR}} = G_{\text{NMDAR}}(t)B(V)(V(t) - V_{\text{rest}})\alpha. \tag{19}$$

$G_{\text{NMDAR}}(t)$ is a variable conductance which accounts for deactivation of the receptor, $B(V)$ is a term which accounts for Mg^{2+} inhibition, the voltage difference $V(t) - V_{\text{rest}}$ is prescribed to emulate a BPAP and EPSP, and α is a scaling term which groups factors such as probability of opening, receptor area density at the PSD, etc.

On the remainder of the plasma membrane which we denote as $\partial\Omega_{\text{pm}}$, we enforce the no-flux boundary condition,

$$D(\mathbf{n} \cdot \nabla u) = 0 \quad \text{on} \quad \partial\Omega_{\text{pm}}. \tag{20}$$

At time $t = 0$, we set the initial concentration of calcium ions to naught throughout the volume of the dendrite,

$$u(\mathbf{x}, t = 0) = 0 \quad \text{in} \quad \bar{\Omega}. \tag{21}$$

Where $\bar{\Omega}$ is the union of the volumetric and boundary domains,

$$\bar{\Omega} \equiv \Omega \cup \partial\Omega. \tag{22}$$

The surface of the geometry is composed of only post synaptic density and plasma membrane,

$$\partial\Omega \equiv \partial\Omega_{\text{psd}} \cup \partial\Omega_{\text{pm}}. \tag{23}$$

Finite element simulations of this model were solved using `FEniCS` [109].

In this simplified model, we assume that the back propagating potential stimulates the entirety of the dendritic branch simultaneously, leading to the opening of NMDARs localized to the PSD and an influx of calcium ions. Several representative snapshots of Ca^{2+} concentration over time, across the geometry, are shown in Fig. 12. The Ca^{2+} transient can be probed by monitoring the concentration at specific locations, shown in Fig. 13. As expected, we first observe that the calcium dynamics are spine size, spine shape and PSD-dependent. Probes 1 and 2 in Fig. 13 are in different spine heads and report differing Ca^{2+} transients. Furthermore, we observe that the narrow spine necks act as a diffusion barrier to calcium, preventing diffusing calcium ions from entering the dendritic shaft as illustrated by probe 3 in Fig. 13. This behavior of the spine neck as a diffusion barrier is consistent with other observations in the literature [111–114].

This example demonstrates that the meshes produced by `GAMer 2` through the workflow are directly compatible with finite element simulations and will allow for the generation of biophysically relevant hypotheses.

Discussion

The relationship between cellular shape and function is being uncovered as systems, structural biology, and physical simulations converge. Beyond traditional compartmentalization, plasma membrane curvature and cellular ultrastructure have been shown to affect the diffusion and localization of molecular species in cells [99, 115]. For example, fluorescence experiments have shown that the dendritic spine necks act as a diffusion barrier to calcium ions, preventing ions from entering the dendritic shaft [111]. Complementary to this and other experiments, various physical models solving reaction-diffusion equations in idealized geometries have been developed to further interrogate the structure-function relationships [51, 99, 110, 116–118].

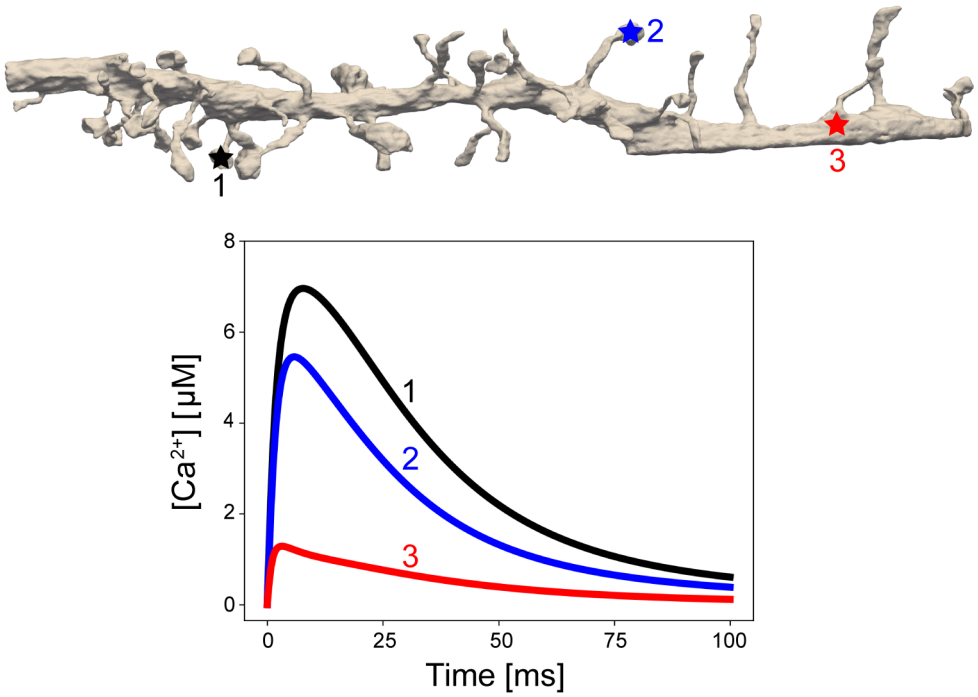

Fig 13. Representative traces of Ca^{2+} concentration over time at three positions. Spine and PSD morphology affect the calcium ion dynamics. For traces 1 and 2, variations in the PSD area and spine head volume lead to different peak calcium ion concentrations. At point 3, the calcium ion concentration values are diminished due to both calcium buffering in the cytosol and the spine neck behaving as a diffusion barrier.

An important next step will be to expand the spatial realism of these models to incorporate realistic geometries as informed by volume imaging modalities. Our tool `GAMer 2` serves as an important step towards filling the need for community driven tools to generate meshes from realistic biological scenes. We have demonstrated the utility of the mesh conditioning algorithms implemented in `GAMer 2` for a variety of systems across several length scales and upwards of hundreds of thousands of triangles. The volume meshes that result from our tools are of high quality (Fig. 7) and we show that they can be used for estimating membrane curvatures (Figs. 8 and 9) and in finite element simulations of reaction-diffusion systems (Figs. 10, 12 and 13).

Bundled with `GAMer 2` we include the `BlendGAMer` add-on which exposes our mesh conditioning algorithms to the `Blender` environment. `Blender` acts as a user interface that provides visual feedback on the effects of `GAMer 2` mesh conditioning operations. `Blender` also enables the painting of boundaries using its many mesh selection tools. Beyond the algorithms in `GAMer 2`, `Blender` also provides an environment for manual curation of mesh artifacts.

Current meshing methods are limited by the need for human biological insight. Experimental setups for volume electron microscopy are arduous and often messy. Microscopists take great care to optimize the experimental conditions, however small variations can lead to sample contamination, tears, precipitation of stain, or other problems. Many of these issues will manifest as artifacts on the micrographs, which makes it challenging to evaluate the ground truth. Automated segmentation algorithms using computer vision and machine learning approaches can fail as a result of these artifacts, and biologists will default to the time-tested, reliable but error-prone mode of manually tracing boundaries.

This is a unique opportunity for biological mesh generation to differentiate from other meshing tools employed in other engineering disciplines. To account for the challenges inherent to biology and wet experiments along with physical simulations, the realization of an automated mesh generation pipeline will require the development of specialized algorithms which tightly couple information across the workflow. As additional annotated datasets become available, machine learning models can be trained to perform tasks that are currently manually executed, such as reconnecting disconnected ER tubules.

The approach and tools presented here, coupled with advances in localization of various membrane proteins [119], brings us closer to the goal of *in silico* biology within realistic geometries. Towards the goal of making 3D cell modeling more routine, experimentalists can contribute by sharing segmented datasets from their work along with biological questions of interest. In exchange, modelers can generate testable predictions and measurements inaccessible to current experimental methods. Specifically, we anticipate that models enabled using `GAMer 2` will be of significant interest to two broad communities in computational biology: membrane biophysicists, focused on the analysis and simulation of membrane shapes, curvature generation, and membrane-protein interactions, and systems biologists, focused on understanding how cell shape and internal organization can impact signal transduction and the dynamics of second messenger microdomains. Through this interdisciplinary exchange, any gaps in our current meshing workflows will be identified and patched.

Conclusion

In this study, we present our mesh generation code `GAMer 2` and described several applications going from contours of electron micrographs as input to generate surface and volume meshes that are compatible with finite element simulations for reaction-diffusion processes. Using the resulting meshes, we have demonstrated the spatio-temporal dynamics of calcium influx in multiple spines along a dendrite. Future efforts will focus on the development of biologically relevant models and generation of experimentally testable hypotheses.

Supporting information

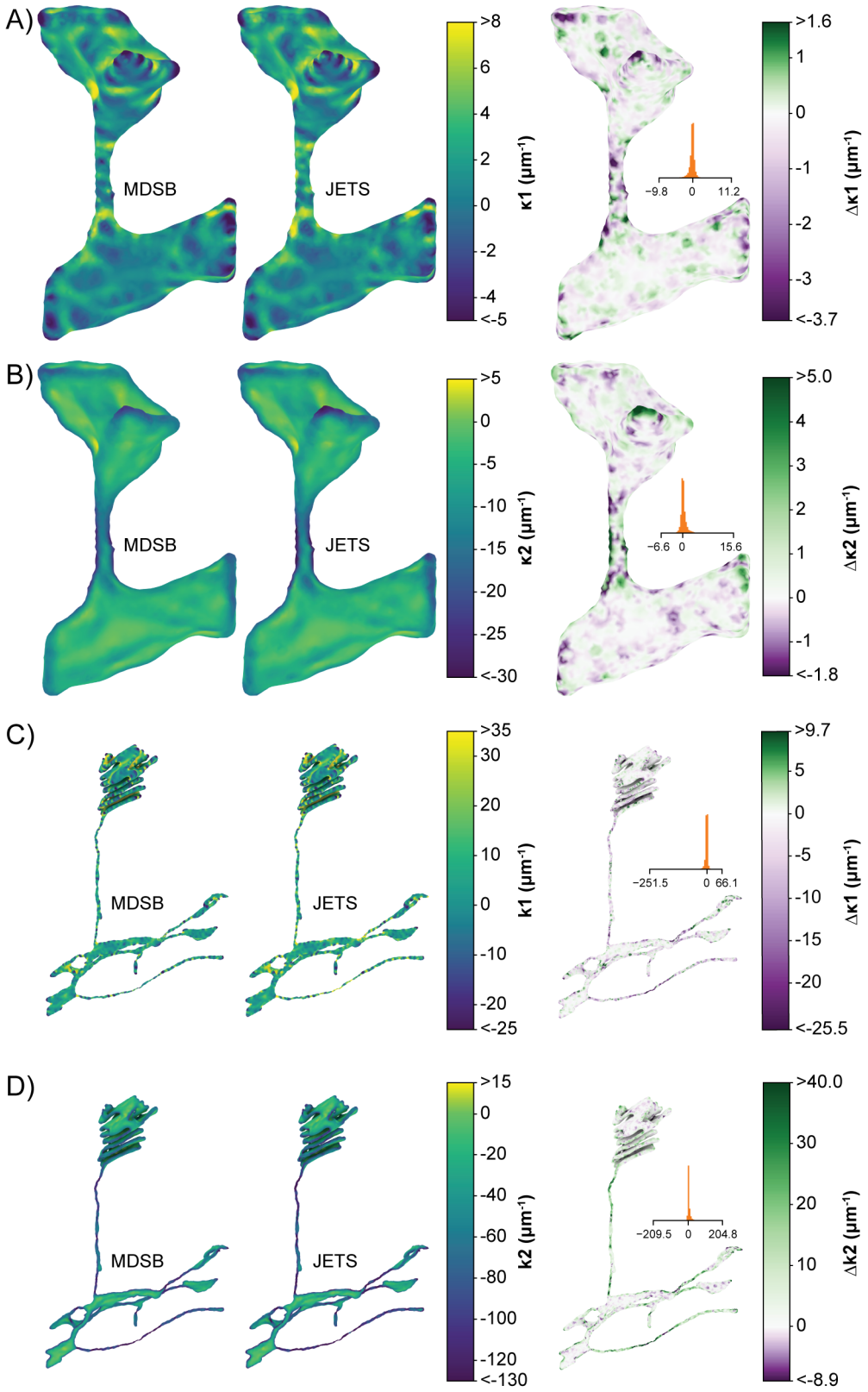
A)
MDSB
JETS
>8
6
4
2
0
-2
-4
<-5
κ1 (μm⁻¹)
−9.8
0
11.2
>1.6
1
0
-1
-2
-3
<-3.7
Δκ1 (μm⁻¹)
B)
MDSB
JETS
>5
0
-5
-10
-15
-20
-25
<-30
κ2 (μm⁻¹)
−6.6
0
15.6
>5.0
4
3
2
1
0
-1
<-1.8
Δκ2 (μm⁻¹)
C)
MDSB
JETS
>35
30
20
10
0
-10
-20
<-25
κ1 (μm⁻¹)
−251.5
0
66.1
>9.7
5
0
-5
-10
-15
-20
<-25.5
Δκ1 (μm⁻¹)
D)
MDSB
JETS
>15
0
-20
-40
-60
-80
-100
-120
<-130
κ2 (μm⁻¹)
−209.5
0
204.8
>40.0
30
20
10
0
<-8.9
Δκ2 (μm⁻¹)

Fig. S1 Comparison of Meyer-Desbrun-Schröder-Barr (MDSB) and Cazal-Pouget (JETS) curvature estimates for the single spine geometry. A) First principal curvature of the plasma membrane. B) Second principal curvature of the plasma membrane. C) First principal curvature of the endoplasmic reticulum. D) Second principal curvature of the endoplasmic reticulum. Shown on the left column are the estimated curvature values. On the right column the difference ($\Delta = \text{MDSB} - \text{JETS}$) between estimates is shown. The difference values are truncated at the 1st and 99th percentiles to improve color range. The distribution and full range of differences are plotted in the inset. Curvature estimates from both algorithms are qualitatively similar. Differences in quantitative value are the greatest at high curvature regions.

Movie S2 Animation proofing the initial mesh against the segmented micrographs. The PM (blue) is rendered as a wireframe. The ER (yellow) is displayed as a solid surface. Purple patches in the segmentation correspond to regions where the PSD is localized. It is using these labeled patches that the boundary marking is generated.

Movie S3 Animation proofing the `GAMer 2` conditioned mesh against the segmented micrographs. The PM (blue) is rendered as a wireframe. The ER (yellow) is displayed as a solid surface. Purple patches in the segmentation correspond to regions where the PSD is localized.

Movie S4 Trajectory of calcium ion signaling in the full dendrite geometry.

Dataset S5 Zip of `Blender` files for all meshes before and after `GAMer 2` processing. All files were generated using `Blender` version 2.80.

Acknowledgments

We would like to thank Prof. Pietro De Camilli and coworkers for sharing their datasets from Wu et al. [7]. We also thank Dr. Matthias Haberl, Mr. Evan Campbell, Profs. Brenda Bloodgood and Mark Ellisman for helpful discussion and suggestions. CTL especially thanks Dr. John B. Moody, and Mr. Mason V. Holst for discussions on GAMer 2 code development along with Dr. Tom Bartol for additional help with using Blender and the design of Blender add-ons. We thank Ms. Miriam Bell, Ms. Kiersten Scott, Ms. Jennifer Fromm, and Dr. Donya Ohadi for critical comments and suggestions for improving this manuscript.

CTL, REA, JAM, and MJH are supported in part by the National Institutes of Health under grant number P41-GM103426. CTL, and JAM are also supported by the NIH under RO1-GM31749. CTL also acknowledges support from the NIH Molecular Biophysics Training Grant T32-GM008326 and a Hartwell Foundation Postdoctoral Fellowship. MJH was supported in part by the National Science Foundation under awards DMS-CM1620366 and DMS-FRG1262982. PR was supported by the Air Force Office of Scientific Research (AFOSR) Multidisciplinary University Research Initiative (MURI) FA9550-18-1-0051 and JGL was supported by a fellowship from the UCSD Center for Transscale Structural Biology and Biophysics/Virtual Molecular Cell Consortium.

References

[1] Sydor AM, Czymmek KJ, Puchner EM, Mennella V. Super-Resolution Microscopy: From Single Molecules to Supramolecular Assemblies. Trends Cell Biol. 2015;25(12):730–748.

[2] Denk W, Horstmann H. Serial Block-Face Scanning Electron Microscopy to Reconstruct Three-Dimensional Tissue Nanostructure. PLOS Biology. 2004;2(11):e329. doi:10.1371/journal.pbio.0020329.

[3] Harris KM, Perry E, Bourne J, Feinberg M, Ostroff L, Hurlburt J. Uniform Serial Sectioning for Transmission Electron Microscopy. Journal of Neuroscience. 2006;26(47):12101–12103. doi:10.1523/JNEUROSCI.3994-06.2006.

[4] Knott G, Marchman H, Wall D, Lich B. Serial Section Scanning Electron Microscopy of Adult Brain Tissue Using Focused Ion Beam Milling. Journal of Neuroscience. 2008;28(12):2959–2964. doi:10.1523/JNEUROSCI.3189-07.2008.

[5] Knott G, Genoud C. Is EM dead? Journal of Cell Science. 2013;126(20):4545–4552. doi:10.1242/jcs.124123.

[6] Briggman KL, Bock DD. Volume electron microscopy for neuronal circuit reconstruction. Current Opinion in Neurobiology. 2012;22(1):154–161. doi:10.1016/j.conb.2011.10.022.

[7] Wu Y, Whiteus C, Xu CS, Hayworth KJ, Weinberg RJ, Hess HF, et al. Contacts Between the Endoplasmic Reticulum and Other Membranes in Neurons. Proc Natl Acad Sci USA. 2017;114(24):E4859–E4867. doi:10.1073/pnas.1701078114.

[8] Kubota Y, Sohn J, Kawaguchi Y. Large Volume Electron Microscopy and Neural Microcircuit Analysis. Frontiers in Neural Circuits. 2018;12:98. doi:10.3389/fncir.2018.00098.

[9] Kasthuri N, Hayworth KJ, Berger DR, Schalek RL, Conchello JA, Knowles-Barley S, et al. Saturated Reconstruction of a Volume of Neocortex. Cell. 2015;162(3):648–661. doi:10.1016/j.cell.2015.06.054.

[10] Calì C, Wawrzyniak M, Becker C, Maco B, Cantoni M, Jorstad A, et al. The Effects of Aging on Neuropil Structure in Mouse Somatosensory Cortex—A 3D Electron Microscopy Analysis of Layer 1. PLOS ONE. 2018;13(7):e0198131. doi:10.1371/journal.pone.0198131.

[11] Motta A, Berning M, Boergens KM, Staffler B, Beining M, Loomba S, et al. Dense Connectomic Reconstruction in Layer 4 of the Somatosensory Cortex. Science. 2019;doi:10.1126/science.aay3134.

[12] Calì C, Agus M, Kare K, Boges DJ, Lehväslaiho H, Hadwiger M, et al. 3D Cellular Reconstruction of Cortical Glia and Parenchymal Morphometric Analysis from Serial Block-Face Electron Microscopy of Juvenile Rat. Progress in Neurobiology. 2019; p. 101696. doi:10.1016/j.pneurobio.2019.101696.

[13] Zheng Z, Lauritzen JS, Perlman E, Robinson CG, Nichols M, Milkie D, et al. A Complete Electron Microscopy Volume of the Brain of Adult Drosophila Melanogaster. Cell. 2018;174(3):730–743.e22. doi:10.1016/j.cell.2018.06.019.

[14] Rognes ME, Ham DA, Cotter CJ, McRae ATT. Automating the solution of PDEs on the sphere and other manifolds in FEniCS 1.2. Geoscientific Model Development. 2013;6(6):2099–2119.

[15] Resasco DC, Gao F, Morgan F, Novak IL, Schaff JC, Slepchenko BM. Virtual Cell: Computational Tools for Modeling in Cell Biology. Wiley Interdisciplinary Reviews Systems Biology and Medicine. 2012 Mar-Apr;4(2):129–140. doi:10.1002/wsbm.165.

[16] Kerr RA, Bartol TM, Kaminsky B, Dittrich M, Chang JCJ, Baden SB, et al. Fast Monte Carlo Simulation Methods for Biological Reaction-Diffusion Systems in Solution and on Surfaces. SIAM Journal on Scientific Computing. 2008;30(6):3126–3149. doi:10.1137/070692017.

[17] Stiles JR, Bartol TM. Monte Carlo Methods for Simulating Realistic Synaptic Microphysiology Using MCell. In: De Schutter E, editor. Computational Neuroscience: Realistic Modeling for Experimentalists; 2001. p. 87–127.

[18] Stiles JR, Van Helden D, Bartol TM, Salpeter EE, Salpeter MM. Miniature Endplate Current Rise Times Less than 100 Microseconds from Improved Dual Recordings Can Be Modeled with Passive Acetylcholine Diffusion from a Synaptic Vesicle. Proceedings of the National Academy of Sciences of the United States of America. 1996;93(12):5747–5752. doi:10.1073/pnas.93.12.5747.

[19] Andrews SS, Bray D. Stochastic Simulation of Chemical Reactions with Spatial Resolution and Single Molecule Detail. Physical Biology. 2004;1(3):137–151. doi:10.1088/1478-3967/1/3/001.

[20] Hepburn I, Chen W, Wils S, De Schutter E. STEPS: Efficient Simulation of Stochastic Reaction–Diffusion Models in Realistic Morphologies. BMC Systems Biology. 2012;6(1):36. doi:10.1186/1752-0509-6-36.

[21] Hepburn I, Chen W, De Schutter E. Accurate Reaction-Diffusion Operator Splitting on Tetrahedral Meshes for Parallel Stochastic Molecular Simulations. The Journal of Chemical Physics. 2016;145(5):054118. doi:10.1063/1.4960034.

[22] Chen W, De Schutter E. Parallel STEPS: Large Scale Stochastic Spatial Reaction-Diffusion Simulation with High Performance Computers. Frontiers in Neuroinformatics. 2017;11. doi:10.3389/fninf.2017.00013.

[23] Drawert B, Engblom S, Hellander A. URDME: A Modular Framework for Stochastic Simulation of Reaction-Transport Processes in Complex Geometries. BMC systems biology. 2012;6:76. doi:10.1186/1752-0509-6-76.

[24] Roberts E, Stone JE, Luthey-Schulten Z. Lattice Microbes: High-Performance Stochastic Simulation Method for the Reaction-Diffusion Master Equation. Journal of Computational Chemistry. 2013;34(3):245–255. doi:10.1002/jcc.23130.

[25] Hattne J, Fange D, Elf J. Stochastic Reaction-Diffusion Simulation with MesoRD. Bioinformatics. 2005;21(12):2923–2924. doi:10.1093/bioinformatics/bti431.

[26] Oliveira RF, Terrin A, Benedetto GD, Cannon RC, Koh W, Kim M, et al. The Role of Type 4 Phosphodiesterases in Generating Microdomains of cAMP: Large Scale Stochastic Simulations. PLOS ONE. 2010;5(7):e11725. doi:10.1371/journal.pone.0011725.

[27] Gut G, Herrmann MD, Pelkmans L. Multiplexed Protein Maps Link Subcellular Organization to Cellular States. Science. 2018;361(6401). doi:10.1126/science.aar7042.

[28] Thul PJ, Åkesson L, Wiking M, Mahdessian D, Geladaki A, Blal HA, et al. A Subcellular Map of the Human Proteome. Science. 2017;356(6340). doi:10.1126/science.aal3321.

[29] Peddie CJ, Collinson LM. Exploring the Third Dimension: Volume Electron Microscopy Comes of Age. Micron. 2014;61:9–19. doi:10.1016/J.MICRON.2014.01.009.

[30] Titze B, Genoud C. Volume Scanning Electron Microscopy for Imaging Biological Ultrastructure. Biol Cell. 2016;108(11):307–323. doi:10.1111/boc.201600024.

[31] Borrett S, Hughes L. Reporting Methods for Processing and Analysis of Data from Serial Block Face Scanning Electron Microscopy. J Microsc. 2016;263(1):3–9. doi:10.1111/jmi.12377.

[32] Tsai WT, Hassan A, Sarkar P, Correa J, Metlagel Z, Jorgens DM, et al. from Voxels to Knowledge: A Practical Guide to the Segmentation of Complex Electron Microscopy 3D-Data. J Vis Exp. 2014;(90). doi:10.3791/51673.

[33] Vasan R, Rowan MP, Lee CT, Johnson GR, Rangamani P, Holst M. Applications and Challenges of Machine Learning to Enable Realistic Cellular Simulations. arXiv:191105218 [physics]. 2019;.

[34] Kremer JR, Mastronarde DN, McIntosh JR. Computer Visualization of Three-Dimensional Image Data Using IMOD. J Struct Biol. 1996;116(1):71–76. doi:10.1006/JSBI.1996.0013.

[35] Sommer C, Strähle C, Köthe U, Hamprecht FA. ilastik: Interactive Learning and Segmentation Toolkit. In: Eighth IEEE International Symposium on Biomedical Imaging (ISBI 2011).Proceedings; 2011. p. 230–233.

[36] Cardona A, Saalfeld S, Schindelin J, Arganda-Carreras I, Preibisch S, Longair M, et al. TrakEM2 Software for Neural Circuit Reconstruction. PLOS ONE. 2012;7(6):1–8. doi:10.1371/journal.pone.0038011.

[37] Edwards J, Daniel E, Kinney J, Bartol T, Sejnowski T, Johnston D, et al. VolRoverN: enhancing surface and volumetric reconstruction for realistic dynamical simulation of cellular and subcellular function. Neuroinformatics. 2014;12(2):277–289.

[38] Boissonnat JD, Geiger B. Three-Dimensional Reconstruction of Complex Shapes Based on the Delaunay Triangulation. In: Biomedical Image Processing and Biomedical Visualization. vol. 1905. International Society for Optics and Photonics; 1993. p. 964–975.

[39] Bermano A, Vaxman A, Gotsman C. Online Reconstruction of 3D Objects from Arbitrary Cross-Sections. ACM Trans Graph. 2011;30(5):113:1–113:11. doi:10.1145/2019627.2019632.

[40] Hu Y, Zhou Q, Gao X, Jacobson A, Zorin D, Panozzo D. Tetrahedral Meshing in the Wild. ACM Trans Graph. 2018;37(4):60:1–60:14. doi:10.1145/3197517.3201353.

[41] Schöberl J. An advancing front 2D/3D-mesh generator based on abstract rules. Comput Vis Sci. 1997;doi:10.1007/s007910050004.

[42] Cignoni P, Callieri M, Corsini M, Dellepiane M, Ganovelli F, Ranzuglia G. MeshLab: an Open-Source Mesh Processing Tool. In: Scarano V, De Chiara R, Erra U, editors. Eurographics Ital. Chapter Conf. The Eurographics Association; 2008.

[43] Geuzaine C, Remacle JF. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. Int J Numer Methods Eng. 2009;79(11):1309–1331. doi:10.1002/nme.2579.

[44] CGAL, Computational Geometry Algorithms Library;. Available from: `http://www.cgal.org`.

[45] Project TC. CGAL User and Reference Manual. 5th ed. CGAL Editorial Board; 2019.

[46] Alliez P, Jamin C, Rineau L, Tayeb S, Tournois J, Yvinec M. 3D Mesh Generation. In: CGAL User and Reference Manual. 5.0 ed. CGAL Editorial Board; 2019.Available from: `https://doc.cgal.org/5.0/Manual/packages.html#PkgMesh3`.

[47] Sheng M, Kim E. The postsynaptic organization of synapses. Cold Spring Harb Perspect Biol. 2011;3(12).

[48] Caré BR, Soula HA. Receptor clustering affects signal transduction at the membrane level in the reaction-limited regime. Phys Rev E Stat Nonlin Soft Matter Phys. 2013;87(1):012720.

[49] Bray D, Levin MD, Morton-Firth CJ. Receptor clustering as a cellular mechanism to control sensitivity. Nature. 1998;393(6680):85–88.

[50] Arendt KL, Royo M, Fernández-Monreal M, Knafo S, Petrok CN, Martens JR, et al. PIP3 controls synaptic function by maintaining AMPA receptor clustering at the postsynaptic membrane. Nat Neurosci. 2010;13(1):36–44.

[51] Ohadi D, Rangamani P. Geometric Control of Frequency Modulation of cAMP Oscillations due to Calcium in Dendritic Spines. Biophys J. 2019;117(10):1981–1994.

[52] Yu Z, Holst MJ, Cheng Y, McCammon JA. Feature-Preserving Adaptive Mesh Generation for Molecular Shape Modeling and Simulation. J Mol Graph Model. 2008;26(8):1370–1380. doi:10.1016/j.jmgm.2008.01.007.

[53] Yu Z, Holst MJ, Andrew McCammon J. High-Fidelity Geometric Modeling for Biomedical Applications. Finite Elem Anal Des. 2008;44(11):715–723. doi:10.1016/j.finel.2008.03.004.

[54] Gao Z, Yu Z, Holst M. Quality Tetrahedral Mesh Smoothing via Boundary-Optimized Delaunay Triangulation. Computer Aided Geometric Design. 2012;29(9):707–721.

[55] Gao Z, Yu Z, Holst M. Feature-Preserving Surface Mesh Smoothing via Suboptimal Delaunay Triangulation. Graphical Models. 2013;75(1):23–38.

[56] Chen L, Holst M. Efficient Mesh Optimization Schemes Based on Optimal Delaunay Triangulations. Comp Meth in Appl Mech Engr. 2011;200(9–12):967–984.

[57] Lee CT, Moody JB, Amaro RE, McCammon JA, Holst MJ. The Implementation of the Colored Abstract Simplicial Complex and its Application to Mesh Generation. ACM Trans Math Softw. 2019;45(3).

[58] Lee CT, Moody JB, Laighlin JG, Holst MJ. GAMer 2.0 Software;. Available from: `https://github.com/ctlee/gamer`.

[59] Jakob W, Rhinelander J, Moldovan D. pybind11 – Seamless operability between C++11 and Python; 2017.

[60] Beazley DM. Automated Scientific Software Scripting with SWIG. Future Gener Comput Syst. 2003;19(5):599–609. doi:10.1016/S0167-739X(02)00171-1.

[61] Blender Online Community. Blender - a Free and Open-Source 3D Computer Graphics Softare Toolset; 2018. Available from: `http://www.blender.org`.

[62] Knutsson H. Representing Local Structure Using Tensors. Comput Vis Lab LINKOPING Univ. 1989; p. 244–251.

[63] Haußecker H, Jähne B. A Tensor Approach for Local Structure Analysis in Multi-Dimensional Images. In: Girod G, Niemann H, Seidel HP, editors. 3D Image Analysis and Synthesis; 1996. p. 171–178.

[64] Fernández JJ, Li S. an Improved Algorithm for Anisotropic Nonlinear Diffusion for Denoising Cryo-Tomograms. J Struct Biol. 2003;144(1-2):152–161. doi:10.1016/j.jsb.2003.09.010.

[65] Weickert J. Anisotropic Diffusion in Image Processing. B. G. Teubner; 1998.

[66] Hughes TJR. The Finite Element Method: Linear Static and Dynamic Finite Element Analysis. Dover Civil and Mechanical Engineering. Dover Publications; 2000. Available from: `https://books.google.com/books?id=yarmSc7ULRsC`.

[67] Belytschko T, Liu WK, Moran B, Elkhodary K. Nonlinear Finite Elements for Continua and Structures. Wiley; 2014. Available from: `https://books.google.com/books?id=BQpfAQAAQBAJ`.

[68] Shewchuk JR. What Is a Good Linear Finite Element? - Interpolation, Conditioning, Anisotropy, and Quality Measures. Proc 11th Int Meshing Roundtable. 2002;94720:115–126.

[69] Zhou T, Shimada K. An Angle-Based Approach to Two-Dimensional Mesh Smoothing. Proc 9th Int Meshing Roundtable. 2000;.

[70] Taubin G. a Signal Processing Approach to Fair Surface Design. In: Proc. 22nd Annu. Conf. Comput. Graph. Interact. Tech. - SIGGRAPH '95; 1995.

[71] Desbrun M, Meyer M, Schröder P, Barr AH. Implicit Fairing of Irregular Meshes Using Diffusion and Curvature Flow. In: Proc. 26th Annu. Conf. Comput. Graph. Interact. Tech. - SIGGRAPH '99; 1999.

[72] Jones TR, Durand F, Desbrun M. Non-iterative, Feature-preserving Mesh Smoothing. ACM Trans Graph. 2003;22(3):943–949. doi:10.1145/882262.882367.

[73] Fleishman S, Drori I, Cohen-Or D. Bilateral Mesh Denoising. ACM Trans Graph. 2003;22(3):950–953. doi:10.1145/882262.882368.

[74] Shewchuk JR. Lecture Notes on Delaunay Mesh Generation; 1999.

[75] Chen CY, Cheng KY. A Sharpness Dependent Filter for Mesh Smoothing. Comput Aided Geom Des. 2005;doi:10.1016/j.cagd.2005.04.003.

[76] Yu Z, Bajaj C. A Segmentation-Free Approach for Skeletonization of Gray-Scale Images Via Anisotropic Vector Diffusion. In: Proc. 2004 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognition, 2004. CVPR 2004.. vol. 1. IEEE; 2004. p. 415–420.

[77] Perona P, Malik J. Scale-Space and Edge Detection Using Anisotropic Diffusion. IEEE Trans Pattern Anal Mach Intell. 1990;doi:10.1109/34.56205.

[78] Cignoni P, Montani C, Scopigno R. A Comparison of Mesh Simplification Algorithms. Computers & Graphics. 1997;22:37–54.

[79] Kobbelt L, Campagna S, peter Seidel H. A General Framework for Mesh Decimation. In: in Proceedings of Graphics Interface; 1998. p. 43–50.

[80] Garland M, Heckbert PS. Surface Simplification Using Quadric Error Metrics. In: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '97. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co.; 1997. p. 209–216. Available from: `https://doi.org/10.1145/258734.258849`.

[81] Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W. Mesh Optimization. In: Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '93. New York, NY, USA: ACM; 1993. p. 19–26. Available from: `http://doi.acm.org/10.1145/166117.166119`.

[82] Dey TK, Edelsbrunner H, Guha S, Nekhayev DV. Topology Preserving Edge Contraction. Publ Inst Math (Beograd) (NS. 1998;66:23–45.

[83] Si H. TetGen, a Delaunay-Based Quality Tetrahedral Mesh Generator. ACM Trans Math Softw. 2015;41(2):1–36. doi:10.1145/2629697.

[84] Nixon-Abell J, Obara CJ, Weigel AV, Li D, Legant WR, Xu CS, et al. Increased Spatiotemporal Resolution Reveals Highly Dynamic Dense Tubular Matrices in the Peripheral ER. Science (80-). 2016;doi:10.1126/science.aaf3928.

[85] Terasaki M, Shemesh T, Kasthuri N, Klemm RW, Schalek R, Hayworth KJ, et al. Stacked Endoplasmic Reticulum Sheets Are Connected by Helicoidal Membrane Motifs. Cell. 2013;154(2):285–296. doi:10.1016/j.cell.2013.06.031.

[86] Marshall WF. Differential Geometry Meets the Cell. Cell. 2013;154(2):265–266. doi:10.1016/j.cell.2013.06.032.

[87] Shemesh T, Klemm RW, Romano FB, Wang S, Vaughan J, Zhuang X, et al. A Model for the Generation and Interconversion of ER Morphologies. Proc Natl Acad Sci USA. 2014;111(49):E5243–E5251. doi:10.1073/pnas.1419997111.

[88] Shibata Y, Shemesh T, Prinz WA, Palazzo AF, Kozlov MM, Rapoport TA. Mechanisms Determining the Morphology of the Peripheral ER. Cell. 2010;143(5):774–788. doi:10.1016/j.cell.2010.11.007.

[89] Hu K, Yan DM, Bommes D, Alliez P, Benes B. Error-Bounded and Feature Preserving Surface Remeshing with Minimal Angle Improvement. IEEE Transactions on Visualization and Computer Graphics. 2017;23(12):2560–2573. doi:10.1109/TVCG.2016.2632720.

[90] Parthasarathy VN, Graichen CM, Hathaway AF. A comparison of tetrahedron quality measures. Finite Elem Anal Des. 1994;15(3):255–261.

[91] Zhang Y, Bajaj C, Sohn BS. 3D Finite Element Meshing from Imaging Data. Computer Methods in Applied Mechanics and Engineering. 2005;194(48):5083–5106. doi:10.1016/j.cma.2004.11.026.

[92] Zhang Y, Bajaj C, Xu G. Surface Smoothing and Quality Improvement of Quadrilateral/Hexahedral Meshes with Geometric Flow. In: Hanks BW, editor. Proceedings of the 14th International Meshing Roundtable. Berlin, Heidelberg: Springer; 2005. p. 449–468.

[93] Delfinado CJA, Edelsbrunner H. An Incremental Algorithm for Betti Numbers of Simplicial Complexes on the 3-Sphere. Computer Aided Geometric Design. 1995;12(7):771–784. doi:10.1016/0167-8396(95)00016-Y.

[94] Cazals F, Pouget M. Estimating Differential Quantities Using Polynomial Fitting of Osculating Jets. Computer Aided Geometric Design. 2005;22(2):121–146. doi:10.1016/j.cagd.2004.09.004.

[95] Pouget M, Cazals F. Estimation of Local Differential Properties of Point-Sampled Surfaces. In: CGAL User and Reference Manual. 5.0 ed. CGAL Editorial Board; 2019.Available from: `https://doc.cgal.org/5.0/Manual/packages.html#PkgJetFitting3`.

[96] Meyer M, Desbrun M, Schröder P, Barr AH. Discrete differential-geometry operators for triangulated 2-manifolds. In: Hege HC, Polthier K, editors. Visualization and Mathematics III. Berlin, Heidelberg: Springer Berlin Heidelberg; 2003. p. 35–57.

[97] Borrelli V, Cazals F, Morvan JM. On the Angular Defect of Triangulations and the Pointwise Approximation of Curvatures. Computer Aided Geometric Design. 2003;20(6):319–341. doi:10.1016/S0167-8396(03)00077-3.

[98] Váša L, Vaněček P, Prantl M, Skorkovská V, Martínek P, Kolingerová I. Mesh Statistics for Robust Curvature Estimation. Computer Graphics Forum. 2016;35(5):271–280. doi:10.1111/cgf.12982.

[99] Rangamani P, Lipshtat A, Azeloglu EU, Calizo RC, Hu M, Ghassemi S, et al. Decoding Information in Cell Shape. Cell. 2013;154(6):1356–1369. doi:10.1016/j.cell.2013.08.026.

[100] Dziuk G. Finite elements for the Beltrami operator on arbitrary surfaces. In: Partial differential equations and calculus of variations. Springer; 1988. p. 142–155.

[101] Holst M. Adaptive numerical treatment of elliptic systems on manifolds. Adv Comput Math. 2001;15(1–4):139–191.

[102] Demlow A, Dziuk G. An adaptive finite element method for the Laplace-Beltrami operator on implicitly defined surfaces. SIAM J Numer Anal. 2007;45(1):421–442 (electronic). doi:10.1137/050642873.

[103] Demlow A. Higher-Order Finite Element Methods and Pointwise Error Estimates for Elliptic Problems on Surfaces. SIAM J Numer Anal. 2009;47:805–827.

[104] Arnold DN, Falk RS, Winther R. Finite element exterior calculus: from Hodge theory to numerical stability. Bull Amer Math Soc (NS). 2010;47(2):281–354. doi:10.1090/S0273-0979-10-01278-4.

[105] Holst M, Stern A. Geometric variational crimes: Hilbert complexes, finite element exterior calculus, and problems on hypersurfaces. Found Comput Math. 2012;12(3):263–293.

[106] Holst M, Stern A. Semilinear Mixed Problems on Hilbert Complexes and Their Numerical Approximation. Found Comput Math. 2012;12(3):363–387.

[107] Gillette A, Holst M, Zhu Y. Finite Element Exterior Calculus for Evolution Problems. Journal of Computational Mathematics. 2017;35(2):186–212.

[108] Holst M, Tiee C. Finite Element Exterior Calculus for Parabolic Evolution Problems on Riemannian Hypersurfaces. Journal of Computational Mathematics. 2018;36(6):792–832.

[109] Alnæs MS, Blechta J, Hake J, Johansson A, Kehlet B, Logg A, et al. The FEniCS Project Version 1.5. Archive of Numerical Software. 2015;3(100). doi:10.11588/ans.2015.100.20553.

[110] Bell M, Bartol T, Sejnowski T, Rangamani P. Dendritic spine geometry and spine apparatus organization govern the spatiotemporal dynamics of calcium. J Gen Physiol. 2019;151(8):1017–1034.

[111] Bloodgood BL. Neuronal Activity Regulates Diffusion Across the Neck of Dendritic Spines. Science (80-). 2005;310(5749):866–869. doi:10.1126/science.1114816.

[112] Bloodgood BL, Sabatini BL. Ca2+ Signaling in Dendritic Spines. Curr Opin Neurobiol. 2007;17(3):345–351. doi:10.1016/j.conb.2007.04.003.

[113] Bloodgood BL, Giessel AJ, Sabatini BL. Biphasic Synaptic Ca Influx Arising from Compartmentalized Electrical Signals in Dendritic Spines. PLoS Biol. 2009;7(9):e1000190. doi:10.1371/journal.pbio.1000190.

[114] Bloodgood BL, Sabatini BL, Van Dongen A. NMDa Receptor-Mediated Calcium Transients in Dendritic Spines. Boca Raton, FL, USA: CRC Press; 2009.

[115] Calizo RC, Ron A, Hu M, Bhattacharya S, Janssen WGM, Hone J, et al. Cell Shape Regulates Subcellular Organelle Location to Control Short-term Ca2+ Signal Dynamics in VSMC. bioRxiv. 2018;doi:10.1101/161950.

[116] Neves SR, Tsokas P, Sarkar A, Grace EA, Rangamani P, Taubenfeld SM, et al. Cell Shape and Negative Links in Regulatory Motifs Together Control Spatial Information Flow in Signaling Networks. Cell. 2008;doi:10.1016/j.cell.2008.04.025.

[117] Cugno A, Bartol TM, Sejnowski TJ, Iyengar R, Rangamani P. Geometric principles of second messenger dynamics in dendritic spines. Sci Rep. 2019;9(1):11676.

[118] Ohadi D, Schmitt DL, Calabrese B, Halpain S, Zhang J, Rangamani P. Computational Modeling Reveals Frequency Modulation of Calcium-cAMP/PKA Pathway in Dendritic Spines. Biophys J. 2019;117(10):1963–1980.

[119] Stone MB, Shelby SA, Veatch SL. Super-Resolution Microscopy: Shedding Light on the Cellular Plasma Membrane. Chem Rev. 2017;117(11):7457–7477.