GAMer 2: A system for 3D mesh processing of cellular electron micrographs

Christopher T. Lee[1,2,☯], 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, for the first time, enabled imaging of single cells in 3D at a nanometer length scale resolution. An uncharted frontier for *in silico* biology is the ability to simulate cellular processes using these observed geometries. Enabling such simulations will require a system for going from electron micrographs to 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 develop an end-to-end pipeline for this task by adapting and extending computer graphics mesh processing and smoothing algorithms. Our workflow makes use of our recently rewritten mesh processing software, `GAMer 2`, which implements several mesh conditioning algorithms and serves as a platform to connect different pipeline steps. We apply this pipeline to a series of electron micrographs of neuronal dendrite morphology explored at three different length scales and demonstrate that the resultant meshes are suitable for finite element simulations. Our pipeline, which consists of free and open-source community driven tools, is a step towards routine physical simulations of biological processes in realistic geometries. We posit that a new frontier at the intersection of computational technologies and single cell biology is now open. Innovations in algorithms to reconstruct and simulate cellular length scale phenomena based on emerging structural data will enable realistic physical models and advance discovery.

Author summary

3D imaging of cellular components and associated reconstruction methods have made great strides in the past decade, opening windows into the complex intraceullar 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. Here, we describe a pipeline that takes images from electron microscopy as input and produces smooth surface and volume meshes as output. These meshes are suitable for building high-quality finite element simulations of cellular processes modeled by ordinary and partial differential equations, bringing us closer to realizing the goal of generating high-resolution simulations of such phenomena in realistic geometries. We demonstrate the utility of this pipeline by meshing 3D reconstructions of dendritic spines, calculating

the curvatures of the different component membranes, and conducting finite-element simulations of reaction-diffusion equations using the generated meshes. The software tools employed in our pipeline are community driven, open source, and free. We believe that technologies such as those presented will 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 the 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. 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 electron direct detectors have enabled the practical applications of techniques such as volume electron microscopy. Advances in microscopy techniques in recent years have opened windows into cells, giving us insight into cellular organization with unprecedented detail. 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 *in silico* animation of various cellular processes and the generation of experimentally testable hypotheses. However, to the best of our knowledge, there is no current free and open-source system for going from EM images to high-quality 3D meshes, which are essential for developing reliable high-resolution finite element simulations of cellular processes.

Here, we introduce our recently redesigned software, `GAMer 2` (Geometry-preserving Adaptive MeshER), as a tool connecting and implementing methods for 3D meshing and reconstruction of cellular electron micrographs (Fig. 1). The input to this pipeline are stacks of high-resolution images (Fig. 1A) from which contours of different organelles and cellular components that are separated by membranes are segmented and identified (Fig. 1B). These steps are often done in experimental and

imaging labs. Superposition of these 2D contours based on the known z-separation gives us the first 3D contour data set (Fig. 1B). Traced contours are then constructed into a primitive 3D surface mesh using `IMOD` (Fig. 1C) [1]. The surface meshes of the external and internal membranes are then conditioned to improve element quality via `GAMer`; faces are also marked using `BlendGAMer`, a `GAMer` addon for `Blender`, which enables the definition of boundary conditions for the simulation (Fig. 1D) [2]. `TetGen` takes these high-quality surface meshes as inputs and generates an unstructured tetrahedral mesh (Fig. 1E) [3]. This mesh is now ready for finite element based physical simulation of different cellular processes (Fig. 1F). In this work, we describe and explore the use of `GAMer 2` to bridge the gap between image acquisition/segmentation (Fig. 1A, B) and modeling with Partial Differential Equations (PDEs) (Fig. 1E, F).

Image acquisition and segmentation

We briefly summarize sample preparation in this section for completeness. Sample preparation begins with either cell culture or 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 [4–7].

Once the images are captured, in order to construct an initial mesh model from the data, the boundaries of features must first be identified. Much of this relies on 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 [7]. The resulting segmentations from volume EM can be visualized as stacks of contours. 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 of mesh generation.

Meshing challenges

A variety of challenges for meshing and subsequent physical simulations can arise at each step. Even with perfect experimental execution, and despite the enhanced surface contrast from heavy metal stains, the membranes of cells and their internal organelles 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 [4]. 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 not mentioned. Most of our datasets have been manually segmented and corrected in software such as `IMOD` [1], `ilastik` [8], or `TrackEM2` [9]. `IMOD` and other tools such as `ContourTiler` in `VolRoverN` [10] have the capacity to perform contour-tiling operations to generate a preliminary surface mesh suitable for basic 3D visualization. These meshes, however, are often not directly suitable for physical simulations due to various mesh artifacts. Some of these include intersecting faces, non-manifold features, and high aspect ratio faces, as shown in Fig. 2. We note that although there exist advanced tetrahedral mesh generation tools, such as `TetWild` [11], which can generate Finite Element Analysis (FEA) compatible volume meshes automatically from these poor quality

initial surfaces, many mesh defects are the result of the limited resolving powers of EM (e.g., Fig. 2A1, A2) and require more careful curation. To resolve these problems, we perform the surface mesh conditioning steps described below.

Methods

Mesh Processing

Our main contribution in this work is a surface mesh processing library `GAMer 2`, which features algorithms as described by Yu *et al.* [12,13], by Gao *et al.* [14,15], and [16]. We have recently rewritten `GAMer` in C++ using the `CASC` data structure [17] as the underlying mesh representation. This rewrite also introduces run-time stability with improved error handing, additional mesh analysis such as curvature estimation, and `pybind11` [18] wrapper based Python API `PyGAMer`. The code is licensed under LGPL v2.1 and can be obtained from GitHub (`https://github.com/ctlee/gamer`) [19]. `GAMer` can be used as a stand-alone library, or alternatively the algorithms can be accessed through a `GAMer Blender` add-on called `BlendGAMer`. `Blender` not only provides a customizable mesh visualization environment, but also tools such as sculpt mode, which allows users to flexibly manipulate the geometry [2]. We briefly review the concepts behind the mesh processing algorithms from Yu et al. [12, 13].

Local Structure Tensor

The mesh processing operations in `GAMer` are designed to preserve the local geometry; algorithms seek to respect the geometric ground truth observed in the micrographs. We use a Local Structure Tensor (LST) to account for the local geometry [20–22]. 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 [23]. 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 elements simulations are extremely sensitive to the quality of the mesh. Poor quality meshes can lead to unbounded error, numerical instability, long times to solution, and non-convergence. Generally, triangulations with high aspect ratios produce larger errors compared with equilateral elements [24].

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 to meshes embedded in 3D of the angle weighted smoothing scheme described by Zhou and Shimada for meshes embedded in 2D [25]. In essence, this algorithm applies local torsion springs to the 1-ring neighborhood of a vertex of interest to balance the angles.

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 [26–29] 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. These images can produce meshes that often contain hundreds of thousands to millions of faces and can cause global optimization based algorithms to fail. Therefore, our approach is a local operation, making it particularly suitable for cellular images.

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 not only improves triangle aspect ratios, but does so 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 [30]. A comparison of the angle-weighted smoothing algorithm with and without LST correction is shown in Fig. 4.

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 [31, 32], including topology preserving Euler operators, other algorithms such as vertex clustering which may not guarantee topological invariance [33], and remeshing [34]. 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 elements 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 [35], we avoid this problem by employing a vertex removal algorithm. First, vertices to be decimated are selected based upon some 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 vertices to remove. First, to selectively decimate vertices in low or high curvature regions, again information from the LST can be used. By comparing the magnitudes of the eigenvalues of the LST we can select for regions with different geometries. 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,

$$\frac{\lambda_2}{\lambda_1} < R_1, \tag{4}$$

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,

$$\frac{\max_{i=1}^{N_1} d(\mathbf{x}, \mathbf{v}_i)}{\overline{D}} < R_2, \tag{5}$$

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`, however the vertex removal decimation scheme can be employed with any other selection criteria.

Feature preserving anisotropic normal-based smoothing

To remove additional bumpiness from the mesh, we use a normal-based smoothing approach [36, 37], 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,

$$\bar{\mathbf{x}} = \frac{1}{\sum_{i=1}^{N_1} a_i} \sum_{i=1}^{N_1} a_i R(\mathbf{x}; e_i, \theta_i). \tag{6}$$

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 [37, 38] to compute the mean neighbor normals,

$$\bar{\mathbf{n}}_i = \frac{1}{\sum_{j=1}^{3} e^{K(\mathbf{n}_i \cdot \mathbf{n}_{ij})}} \sum_{j=1}^{3} e^{K(\mathbf{n}_i \cdot \mathbf{n}_{ij})} \mathbf{n}_{ij}, \tag{7}$$

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.

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 and local receptor clusters can be arbitrarily distributed on the manifold, boundary definition is a non-trivial challenge. The `BlendGAMer` add-on supports the facile user-based definition of boundary markers on the surface [2]. Users can utilize any of the face selection methods 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 through `GAMer` into `TetGen` for tetrahedralization [3].

Mesh Generation Pipeline

Electron micrographs and segmentations from Wu *et al.* [39] were graciously shared by De Camilli and coworkers (Fig. 1A, B). We generated preliminary meshes of the geometry using the `imod2obj` utility included with `IMOD` [1] (Fig. 1C). The quality of the mesh was improved using algorithms described in §Mesh Processing and implemented in `GAMer 2` [19] (Fig. 1D). Some features, such as disconnections of the Endoplasmic Reticulum (ER), were manually reconnected using `Blender` mesh sculpting features. Additional discussion of the mesh artifacts and the curation process is described in the example applications. Boundaries were marked and the conditioned surface mesh was tetrahedralized using `TetGen` [3] (Fig. 1E).

Estimation of Membrane Curvatures

In addition to the generation of FEA compatible meshes of realistic cell geometries, this workflow can produce meshes 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.

Using the conditioned surface meshes, the curvature can be estimated using methods from discrete differential geometry. In `GAMer`, we have implemented the algorithms to compute curvatures as described by Meyer et al. [40], summarized as follows. The mean curvature normal is given by

$$\mathbf{K}(\mathbf{x}_i) = \frac{1}{\mathcal{A}_{\text{Mixed}}} \sum_{j \in N_1(i)} (\cot \alpha_{ij} + \cot \beta_{ij})(\mathbf{x}_i + \mathbf{x}_j), \tag{8}$$

where $\mathcal{A}_{\text{Mixed}}$ is the mixed discrete area around vertex $\mathbf{x}_i$, $\mathbf{x}_j$ is a vertex in the first ring of neighbors $N_1(i)$, α_{ij}, and β_{ij} are the angles opposite to the edge i, j. From the mean curvature normal we compute the signed mean curvature

$$\kappa_H(\mathbf{x}_i) = \frac{1}{2}\mathbf{K}(\mathbf{x}_i) \cdot \mathbf{n}_i, \tag{9}$$

where $\mathbf{n}_i$ is the unit surface normal at $\mathbf{x}_i$. To compute the discrete Gaussian curvature, we use the angle deficit formula

$$\kappa_G(\mathbf{x}_i) = \frac{1}{\mathcal{A}_{\text{Mixed}}} \left(2\pi - \sum_{j=1}^{\#f} \theta_j \right), \tag{10}$$

where θ_j is the angle of the jth face at vertex $\mathbf{x}_i$. The principal curvatures κ_1 and κ_2 are then computed from the mean (κ_H) and Gaussian (κ_G) curvatures

$$\kappa_1(\mathbf{x}_i) = \kappa_H(\mathbf{x}_i) + \sqrt{\Delta(\mathbf{x}_i)} \tag{11}$$

$$\kappa_2(\mathbf{x}_i) = \kappa_H(\mathbf{x}_i) - \sqrt{\Delta(\mathbf{x}_i)}, \tag{12}$$

where $\Delta(\mathbf{x}_i) = \kappa_H^2(\mathbf{x}_i) - \kappa_G(\mathbf{x}_i)$. In addition to estimating the curvatures, we can use the mesh models to interrogate the impacts of curvature on signaling.

Building High-Resolution Finite Element Simulations from High-Quality Tetrahedral Meshes

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 [41] on a dendritic spine. The spine was first geometrically modeled using the meshing pipeline described here, and the mesh was then used as the spatial domain for a detailed numerical simulation using the finite element method. Consider the reaction $\mathrm{A} + \mathrm{X} \underset{k_{\text{off}}}{\overset{k_{\text{on}}}{\rightleftharpoons}} \mathrm{B}$, 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 PDE,

$$\frac{\partial \mathrm{A}}{\partial t} = D_{\mathrm{A}} \Delta \mathrm{A} \quad \text{in} \quad \Omega, \tag{13}$$

two surface PDEs,

$$\frac{\partial \mathrm{X}}{\partial t} = D_{\mathrm{X}} \Delta_S \mathrm{X} - k_{\text{on}} \mathrm{A}|_{\partial\Omega} \mathrm{X} + k_{\text{off}} \mathrm{B} \quad \text{on} \quad \partial\Omega \tag{14}$$

$$\frac{\partial \mathrm{B}}{\partial t} = D_{\mathrm{B}} \Delta_S \mathrm{B} + k_{\text{on}} \mathrm{A}|_{\partial\Omega} \mathrm{X} - k_{\text{off}} \mathrm{B} \quad \text{on} \quad \partial\Omega, \tag{15}$$

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

$$D_{\mathrm{A}}(\mathbf{n}\cdot\nabla \mathrm{A}) = -k_{\mathrm{on}}\mathrm{A}\mathrm{X} + k_{\mathrm{off}}\mathrm{B} \quad \text{on} \quad \partial\Omega. \tag{16}$$

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. 9A). 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 $\mathrm{A}(t=0) = 1.0\,\mu\mathrm{M}$, $\mathrm{X}(t=0) = 1000\,\text{molecules}\,\mu\mathrm{m}^{-2}$, and $\mathrm{B}(t=0) = 0\,\text{molecules}\,\mu\mathrm{m}^{-2}$. 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:

$$\int_{\Omega} \frac{\mathrm{A}^{(n+1)} - \mathrm{A}^{(n)}}{\delta t} v_{\mathrm{A}} + D_{\mathrm{A}}\nabla \mathrm{A}^{(n+1)}\cdot\nabla v_{\mathrm{A}}\, d\Omega + \int_{\partial\Omega} k_{\mathrm{on}}\mathrm{A}^{(n+1)}\tilde{\mathrm{X}}v_{\mathrm{A}} - k_{\mathrm{off}}\tilde{\mathrm{B}}v_{\mathrm{A}}\, d\Gamma = 0, \tag{17}$$

$$\int_{\partial\Omega} \frac{\mathrm{X}^{(n+1)} - \mathrm{X}^{(n)}}{\delta t} v_{\mathrm{X}} + D_{\mathrm{X}}\nabla_S \mathrm{X}^{(n+1)}\cdot\nabla_S v_{\mathrm{X}} + k_{\mathrm{on}}\tilde{\mathrm{A}}\mathrm{X}^{(n+1)}v_{\mathrm{X}} - k_{\mathrm{off}}\mathrm{B}^{(n+1)}v_{\mathrm{X}}\, d\Gamma = 0, \tag{18}$$

$$\int_{\partial\Omega} \frac{\mathrm{B}^{(n+1)} - \mathrm{B}^{(n)}}{\delta t} v_{\mathrm{B}} + D_{\mathrm{B}}\nabla_S \mathrm{B}^{(n+1)}\cdot\nabla_S v_{\mathrm{B}} - k_{\mathrm{on}}\tilde{\mathrm{A}}\mathrm{X}^{(n+1)}v_{\mathrm{B}} + k_{\mathrm{off}}\mathrm{B}^{(n+1)}v_{\mathrm{B}}\, d\Gamma = 0. \tag{19}$$

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. (17) is solved to estimate $\mathrm{A}^{(n+1)}$, this estimate is then used in Eqs. (18) and (19) to obtain an estimate for $\mathrm{X}^{(n+1)}$ and $\mathrm{B}^{(n+1)}$ which are used again in Eq. (17) to further improve the estimate of $\mathrm{A}^{(n+1)}$. This cycle continues until a satisfactory convergence criterion is met.

Note that Eqs. (14) and (15) 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 PDE 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 (SFEM) have emerged [42–45] 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 (FEEC) [46] to provide a more general error analysis framework for surface finite element methods on n-surfaces, for static linear and nonlinear problems [47, 48], as well as for evolution problems on surfaces [49, 50]. 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` [51], and to use surface finite element modifications to `FEniCS` (described e.g. in [52]) for solving our geometric PDE Eqs. (14) and (15) above.

Simulating reaction-diffusion in the volume

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 in Fig. 10, see also Movie S1 available online. Because the goal of this simulation was not to show biological accuracy, but rather to demonstrate that our pipeline is capable of producing biophysically relevant FEA simulations, we use a simplified version of the model found in Bell et al. [53].

We model a BPAP and EPSP which stimulates NMDAR opening and calcium ion influx into the cell. The reaction-diffusion of u, corresponding to calcium ion concentration, is described as follows,

$$\frac{\partial u}{\partial t} = D\Delta u - \frac{u}{\tau} \quad \text{in} \quad \Omega, \tag{20}$$

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

$$D(\mathbf{n} \cdot \nabla u) = J_{\text{NMDAR}}(t) \quad \text{on} \quad \partial\Omega_{\text{psd}}, \tag{21}$$

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

$$J_{\text{NMDAR}} = G_{\text{NMDAR}}(t)B(V)(V(t) - V_{\text{rest}})\alpha. \tag{22}$$

$G_{\text{NMDAR}}(t)$ 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 Postsynaptic Density (PSD), etc.

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

$$D(\mathbf{n} \cdot \nabla u) = 0 \quad \text{on} \quad \partial\Omega_{\text{pm}}. \tag{23}$$

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

$$u(\mathbf{x}, t = 0) = 0 \quad \text{in} \quad \bar{\Omega}. \tag{24}$$

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

$$\bar{\Omega} \equiv \Omega \cup \partial\Omega. \tag{25}$$

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

$$\partial\Omega \equiv \partial\Omega_{\text{psd}} \cup \partial\Omega_{\text{pm}}. \tag{26}$$

The model was solved using `FEniCS` [51] to run finite elements simulations.

Results

Table 1. Vertex and Element Counts for Meshed Geometries.

		Surface Mesh		Volume Mesh*	
		# Vertices	# Triangles	# Vertices	# Tetrahedra
Single Spine	Initial PM	4,695	9,330	–	–
	Conditioned PM	6,924	13,844	13,734	62,557
	Initial ER	6,546	19,654	–	–
	Conditioned ER	36,294	72,620	53,134	211,018
Two Spines	Initial PM	160,733	320,976	–	–
	Conditioned PM	18,027	36,050	28,989	122,082
Dendritic Segment	Initial PM	207,448	410,896	–	–
	Conditioned PM	126,336	252,668	194,848	798,626

*Non-manifold and other mesh artifacts prevent the tetrahedralization of initial meshes

We demonstrate the application of our pipeline to dendritic spine reconstructions of different sizes. All images of dendritic spines are from neurons taken from mouse cerebral cortex or nucleus accumbens from a recent publication using Focused-ion Beam Milling Scanning Electron Microscopy (FIB-SEM) [39]. In addition to their important role in synaptic and structural plasticity, these cellular structures demonstrate highly tortuous morphologies, high surface-to-volume ratios, and a geometric intricacy that serves as a good test-bed for our pipeline.

We have generated FEA compatible meshes of several geometries of increasing length scale: the ER of the single spine geometry which requires nanometer precision (Fig. 5E, F), the Plasma Membrane (PM) of the single spine which has a length scale of a couple of microns (Fig. 5D), the two spine geometry, a few microns (Fig. 5), and the dendrite with about 40 spines, with a length scale in the tens of microns (Fig. 5) [39]. We describe some of the challenges associated with the mesh conditioning process for each system and explore the estimation of membrane curvature on the mesh geometries (Figs. 6 and 7). Next, we evaluate the degree of improvement in finite element solution accuracy with respect to `GAMer` mesh conditioning algorithms. (Fig. 8). Using the resulting meshes, we perform two example simulations: 1) coupling surface and volume reaction-diffusion (Fig. 9), and 2) a simulated time-series of calcium influx due to NMDAR by a BPAP and EPSP in the full dendritic segment geometry (Figs. 10 and 11).

Meshing segments of increasing length scales

We have constructed a mesh of a single spine geometry based on EM images taken of the mouse cerebral cortex, shown in Fig. 5. This geometry contains two membranes: the PM and the ER

Fig. 5A. Due to differences in the morphology of the PM versus the ER, each component presents with it different meshing challenges. The PM contains 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 experimental choice of sample cutting planes may result in truncation of the image of the structure. We have remedied this truncation by triangulating the holes and smoothing out the local curvature.

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 [54]. Some ER morphology cannot even be resolved by the powers of EM [39].

For example, if the ER undergoes large geometric 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. In these cases, we manually curate the mesh in `Blender` to reconnect the broken ER segments. Fig. 2A1 and A2 illustrate this process in more detail. This spine also contains a specialized form of ER termed the spine apparatus, Fig. 5A, inset, which consists of seven folded cisternae. This highly organized structure bears geometrical similarities to a parking garage structure and helicoidal geometries [55–58]. The geometric detail of the spine apparatus is preserved by the conditioning process in our pipeline.

In Fig. 5, we also show the distribution of the 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 [24]. Prior to conditioning, the angle distribution is spread out and contains many large and small angles. After processing using `GAMer`, the angles of the mesh are improved, as indicated by the peaked distribution around 60 degrees. Fig. 2C shows a closeup of high aspect ratio triangles before and after processing with `GAMer`.

It is worth noting that even though this mesh of PM of the single spine is the simplest geometry we examine, there are many intersecting faces in the initial mesh. This is a common artifact which appears in most initial meshes but is easily and automatically resolved by `GAMer`'s vertex smoothing algorithm as shown in Fig. 2B.

Although the ER structure is significantly more complex, the angles 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. 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. 5A about 48% more triangles were needed compared to the ER mesh in Fig. 5A, inset, which required 270% more triangles, both relative to the initial mesh.

The pipeline 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. 5B 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` 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.

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. 5C. As in the one and two spine cases, our system robustly handles artifacts such as poor quality triangles and intersecting faces; Fig. 5C 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 pipeline's capability to produce well-conditioned meshes. Fig. 5C 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 `GAMer` for use denoting a boundary condition.

Estimating Membrane Curvatures in Realistic Geometries

One of the advantages of using EM-generated images of membrane structures in cells is that we can now bridge the gap between membrane mechanics and curvature studies and the 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.

Here, we use the meshes generated using our pipeline and calculate the curvature of the geometries using methods from discrete differential geometry. Shown in Figs. 6 and 7 are the principal curvatures κ_1 and κ_2 respectively. Looking at the first principal curvature, Fig. 6, corresponding to the maximum curvature of the local region, we first observe that for all geometries, this value is primarily positive. This signifies that the principal curvature turns in the same direction as the surface's outward facing normal. We also observe that tubular regions such as the neck of the spine have near uniform curvature (Fig. 6B). Finally, the regions of high bending such as the folds of the spine apparatus in the spine head (Fig. 6A, B, left panels) are highly curved and are connected by sheets with low curvature. The ER tubules along the spine neck have a near constant 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. 6C).

The second principal curvature, which corresponds to the minimum curvature of the local region, shows a different behavior Fig. 7. First, we found that the distribution of κ_2 spans both positive and negative values, centered around zero for both the plasma membrane and the endoplasmic reticulum. 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`, we are able to quantify the curvatures along the plasma membrane and the internal organelle membranes using tools from discrete differential geometry.

Simulations of Surface Diffusion

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. 9A. Shown in Fig. 9B 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. 9D). Fig. 9E

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` can be used for systems biology in realistic geometries.

Mesh convergence analysis

To illustrate the effects of `GAMer 2` mesh conditioning on FEA results, 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.,

$$\varepsilon_{L_2} = \left(\int_{\Omega} (u' - u)^2 \, d\Omega \right)^{\frac{1}{2}}. \tag{27}$$

This is a standard procedure that is used to measure h-refinement convergence rates; however between iterations of `GAMer` 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`, we used an error metric based on the relative difference in total molecules,

$$\varepsilon_{\text{rel}} = \left| \frac{\int_{\Omega'} u' \, d\Omega' - \int_{\Omega} u \, d\Omega}{\int_{\Omega} u \, d\Omega} \right|. \tag{28}$$

Fig. 8 (A-D) shows intermediary steps of the `GAMer` refinement process. For each mesh, a given number of smoothing iterations was performed; any remaining artifacts that would prevent tetrahedralization such as intersecting faces were removed and the resultant holes were re-triangulated. The surface meshes were all tetrahedralized using `TetGen` with the same parameters. The first time step of a reaction-diffusion partial differential equation with a constant Neumann boundary condition was solved and the resulting solutions were compared using Eq. (28), where Fig. 8E was assumed to be the ground-truth. Fig. 8F shows that the relative error consistently decreased as a result of further mesh conditioning in `GAMer`. The mesh with no smoothing operations applied had almost five times more molecules than the ground-truth mesh after just one time step. This analysis highlights not only the importance of using a high quality mesh in FEA but also that `GAMer` can generate such high quality meshes.

Simulation of reaction-diffusion equations in dendrites

Next, we simulated a simplified model of calcium influx to demonstrate that the meshes generated using `GAMer` can be used for analysis of complex biological models. 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. As a result of this stimulus, calcium ions enter into the dendritic spine heads through the open NMDA receptors localized at the post synaptic density. Several representative snapshots of Ca^{2+} concentration over time, across the geometry, are shown in Fig. 10. The Ca^{2+} transient can be probed by monitoring the concentration at specific locations, shown in Fig. 11. As expected, we first observe that the calcium dynamics in spines are spine size, spine shape and PSD-dependent. Probes 1 and 2 in Fig. 11 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. 11. This behavior of the spine neck as a diffusion barrier is consistent with other observations in the literature [59–62].

This example demonstrates that the meshes produced by `GAMer` 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 [41, 63]. 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 [59]. 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 [41, 53, 64–67].

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` 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` 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 high quality (Fig. 1E) and we show that they can be used directly for computation of membrane curvatures (Figs. 6 and 7) and in finite elements simulations of reaction-diffusion systems (Figs. 10 and 11).

Bundled with `GAMer` 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` mesh conditioning operations. `Blender` also enables the painting of boundaries using its many mesh selection tools. Beyond the algorithms in `GAMer`, `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 problems induced by biology and wet experiments along with physical simulations, we anticipate that, to realize 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 which are currently manually executed such as reconnecting disconnected ER tubules.

We believe that our approach, coupled with advances in localization of various membrane

proteins [68], can bring us closer to the goal of *in silico* biology with realistic geometries. To accelerate this goal of routine cell modeling, 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 our pipeline 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 by 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 can be identified and patched.

Conclusion

In this study, we have described a computational pipeline that uses contours from electron microscopy images as input to generate surface and volume meshes that are compatible with finite element simulations for reaction-diffusion processes. Using this pipeline, 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.

Acknowledgments

We would like to thank Prof. Pietro De Camilli and coworkers for sharing their datasets from Wu et al. [39]. 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 addons. 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. 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] 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.

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

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

[4] 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.

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

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

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

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

[16] 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.

[17] Lee CT, Moody JB, Amaro RE, McCammon JA, Holst M. The Implementation of the Colored Abstract Simplicial Complex and its Application to Mesh Generation. arXiv. 2018;abs/1807.01417.

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

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

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

[21] 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.

[22] 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.

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

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

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

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

[27] 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.

[28] 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.

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

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

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

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

[33] 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`.

[34] 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`.

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

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

[37] 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.

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

[39] 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.

[40] 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.

[41] 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.

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

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

[44] 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.

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

[46] 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.

[47] 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.

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

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

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

[51] 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.

[52] 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.

[53] Bell M, Bartol T, Sejnowski T, Rangamani P. Dendritic spine geometry and spine apparatus organization govern the spatiotemporal dynamics of calcium. Journal of General Physiology. In Press;doi:10.1101/386367.

[54] 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.

[55] 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.

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

[57] 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.

[58] 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.

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

[60] 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.

[61] 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.

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

[63] 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.

[64] 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.

[65] Cugno A, Bartol TM, Sejnowski TJ, Iyengar R, Rangamani P. Geometric Principles of Second Messenger Dynamics in Dendritic Spines. bioRxiv. 2018;doi:10.1101/444489.

[66] 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. bioRxiv. 2019;doi:10.1101/521740.

[67] Ohadi D, Rangamani P. Geometric control of frequency modulation of cAMP oscillations due to Ca^{2+}-bursts in dendritic spines. bioRxiv. 2019;doi:10.1101/520643.

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

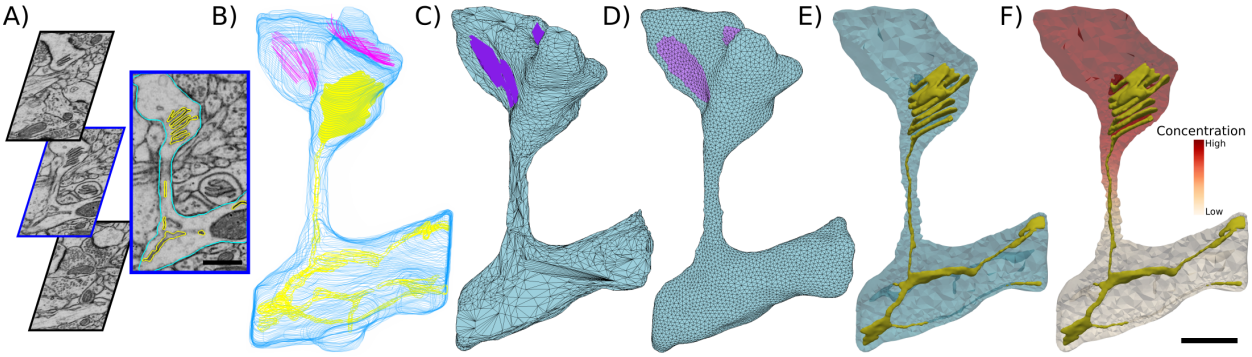

Fig 1. Pipeline from electron microscopy data to a reaction-diffusion finite element simulation on a well-conditioned unstructured tetrahedral mesh. A) Contours of segmented data overlaid on raw slices of electron microscopy data, B) Stacked contours from all slices of segmented data, C) Primitive initial 3D mesh reconstructed by existing `IMOD` software, D) Surface mesh after processing with our system; note the significantly higher quality of the mesh. E) Unstructured tetrahedral mesh suitable for finite element simulation obtained with `TetGen` software, F) Reaction-diffusion model simulated using `FEniCS` software. Scale bar: 500 nm.

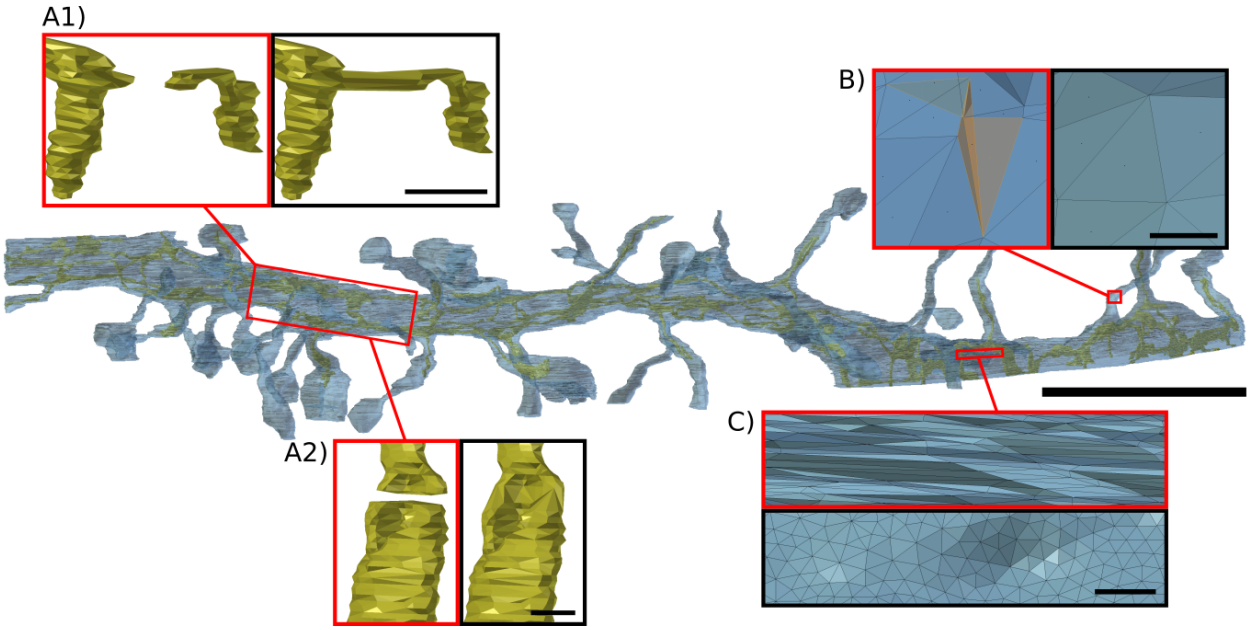

Fig 2. Initial surface mesh model of a dendritic segment with subcellular organelles imaged by FIB-SEM, courtesy of Wu et al. [39], contains many mesh artifacts and is not compatible with physical modeling. The blue surface represents the plasma membrane (PM), while yellow represents the membrane of the endoplasmic reticulum (ER). (A1/A2) Left, due to the fine ultrastructure of the process, portions of the ER are not resolved by the microscope and become erroneously disconnected as separate meshes. Right, biological realism is manually restored by bridging the two ER segments. Scale bars: A1) 100 nm, A2) 20 nm. (B) Left, intersecting faces prevent generation of a tetrahedral mesh. Right, `GAMer` smoothing algorithms automatically untangle knots of intersecting faces. Scale bar: 20 nm. (C) Top, high aspect ratio triangles. Bottom, well conditioned surface mesh. Scale bar: 100 nm. Scale bar for full geometry: 4 μm.

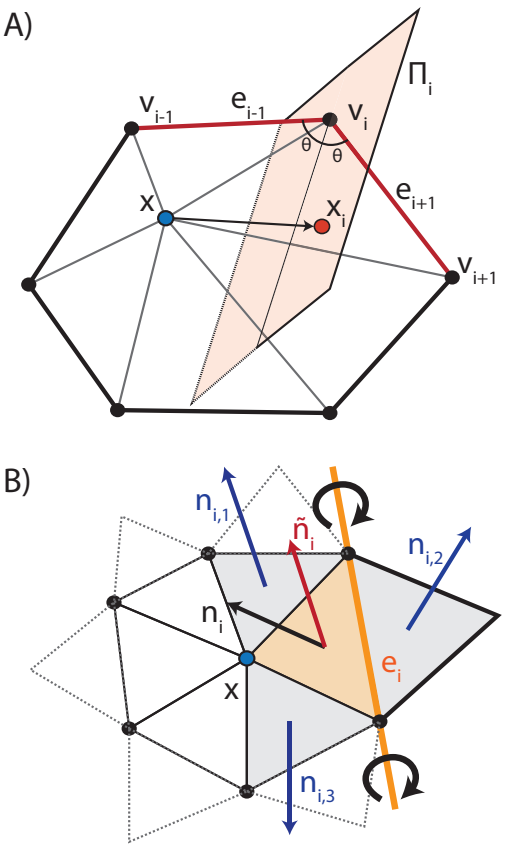

Fig 3. Schematic illustrating `GAMer` mesh conditioning algorithms. A) angle-based surface mesh conditioning, and B) anisotropic normal smoothing algorithms which are previously described by Yu et al. [12, 13] and implemented in `GAMer`.

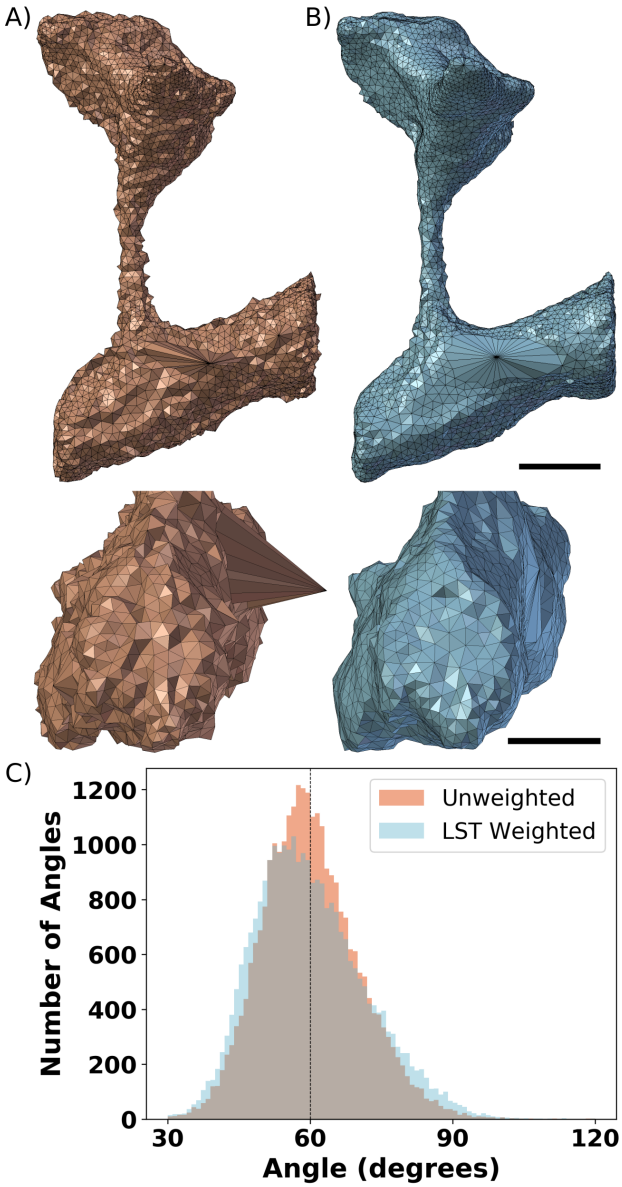

Fig 4. Comparison of 50 iterations of angle weighted smoothing algorithm. A) without and B) with Local Structure Tensor (LST) based correction. The LST helps to preserve the geometric structure albeit with slight degradation to the mesh angles. LST is a simple metric to capture local geometric information which can be used to constrain conditioning operations. C) Mesh angles are improved in both the LST weighted and unweighted meshes. Scale bar top row: 500 nm, bottom row: 250 nm.

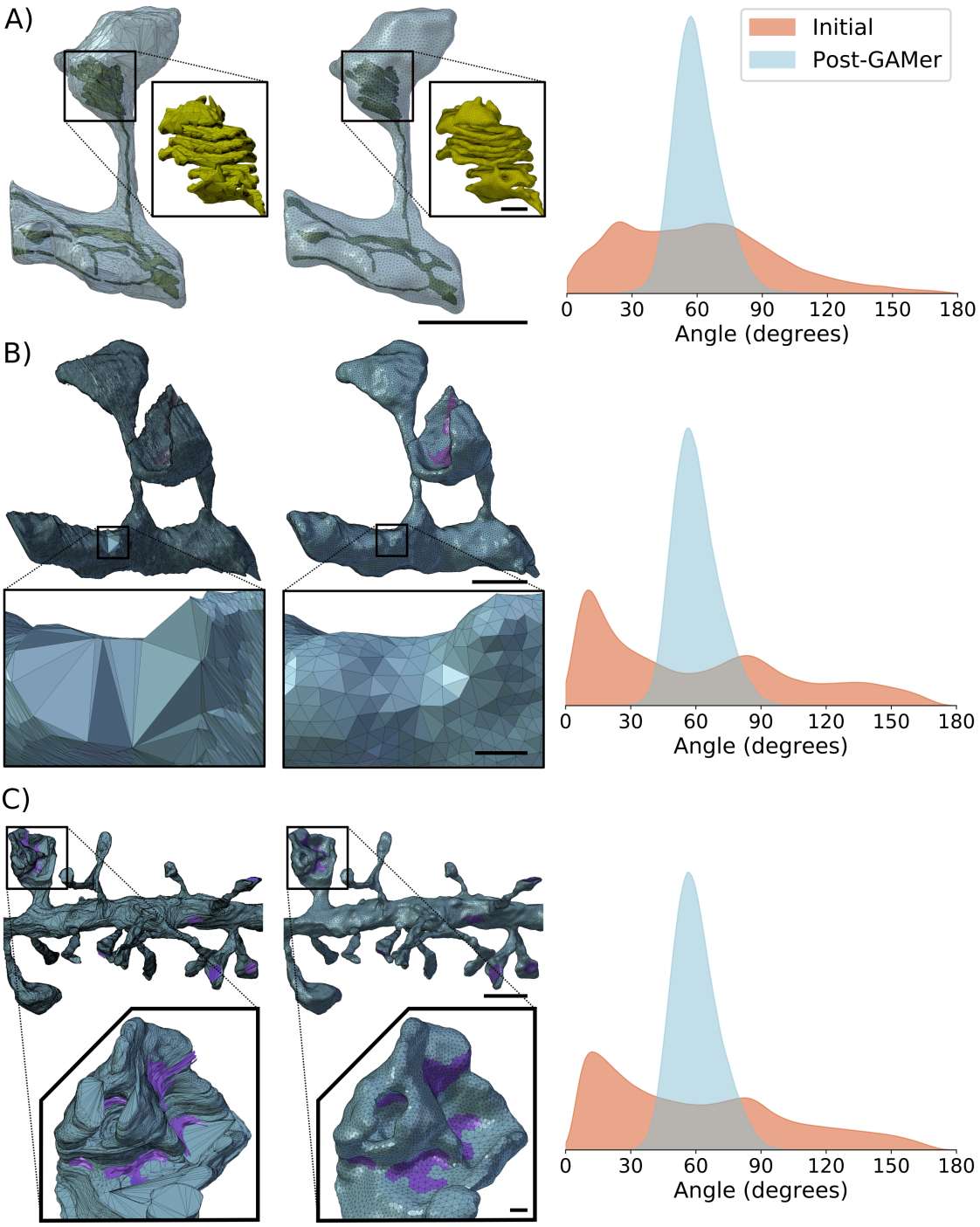

Fig 5. Quantification of mesh quality pre- and post-`GAMer` processing for several geometries. Data at varying spatial scales can be processed via the `GAMer` framework. Surface meshes of dendritic spine geometries before (left) are compared with their mesh after `GAMer` processing (middle). The shift 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. B) Surface mesh of PM of two dendritic spines. Faces marked as purple are PSD. Inlay: close-up of a region with a large variance in angle distribution before `GAMer` processing. C) Surface mesh of PM of a dendrite segment with many spines. Inlay: `GAMer` preserves the intricate details of a highly curved spine head with multiple regions of PSD. Scale bars: full geometries 1 μm, inlays: 100 nm.

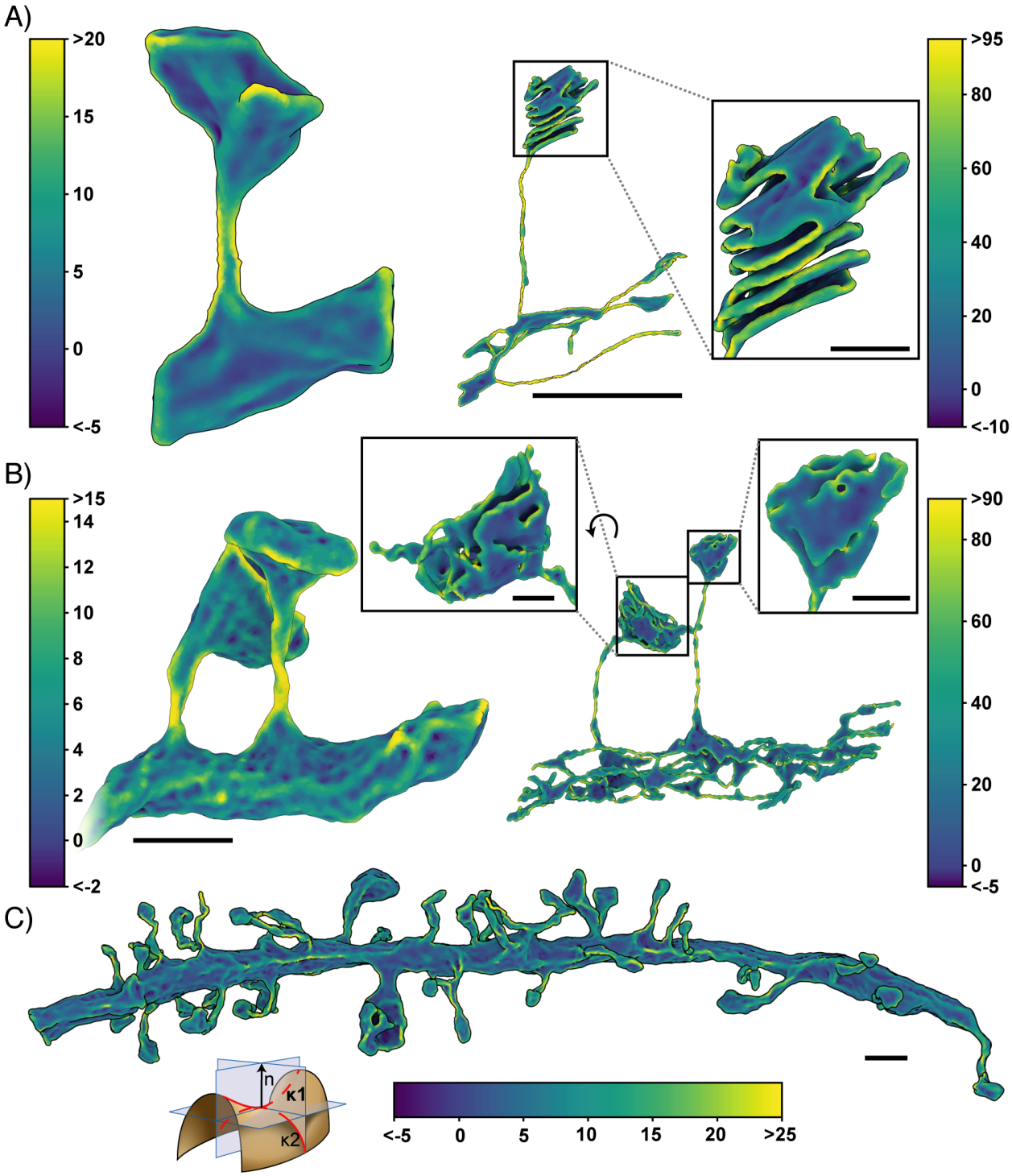

Fig 6. Estimated first principal curvatures of the spine geometries. The signed first principal curvature, corresponding to the maximum curvature at each mesh point, is estimated using `GAMer`. Color bars correspond to curvature values with units of μm^{-1}. Geometries are A) single spine model, left: plasma membrane, right: endoplasmic reticulum; B) two spine moddel, left: plasma membrane, right: endoplasmic reticulum; and C) plasma membrane of the dendritic branch model. Scale bars: full geometries 2 μm, inlays: 200 nm. Curvature schematic modified from Wikipedia, credited to Eric Gaba (CC BY-SA 3.0).

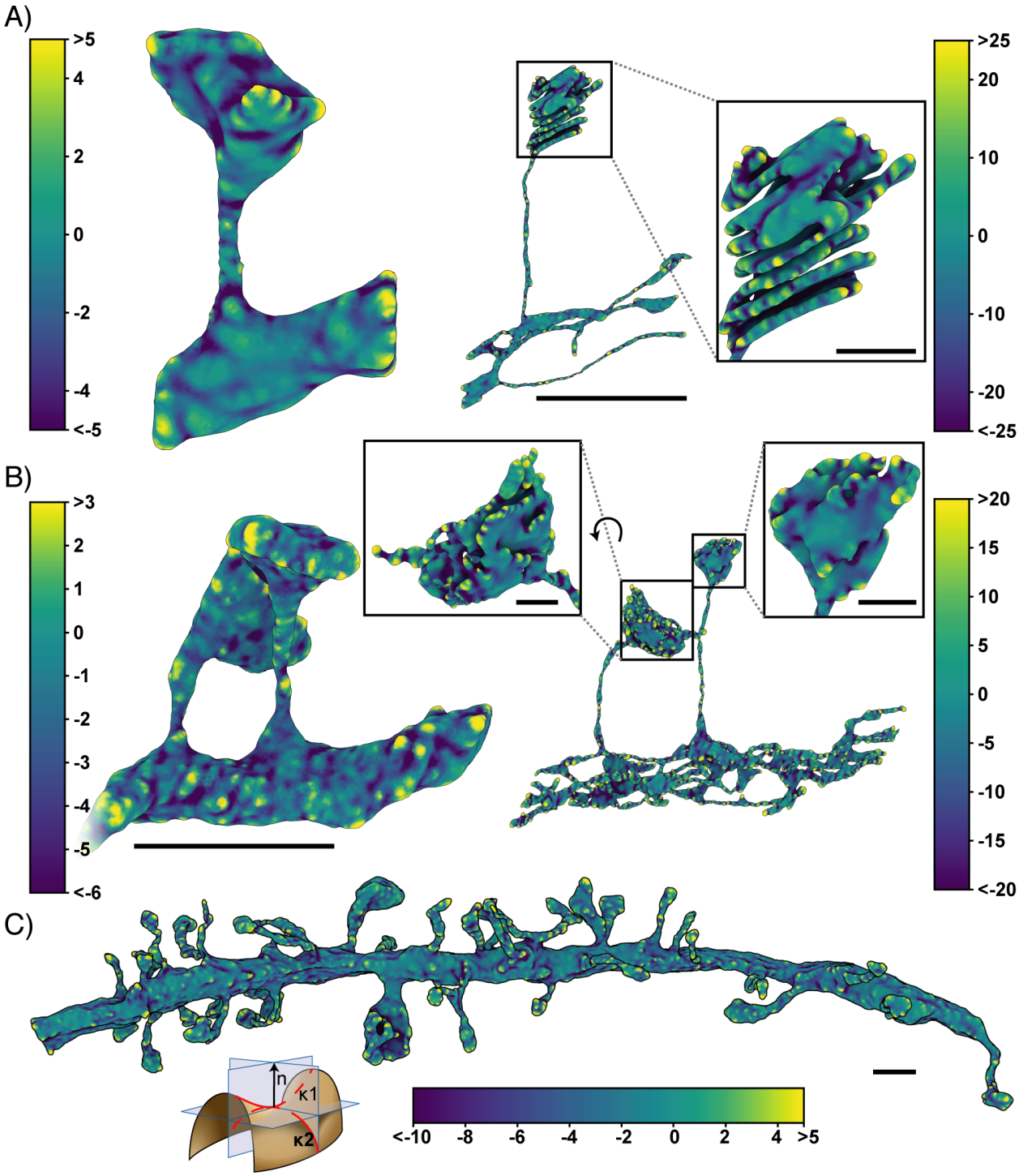

Fig 7. Estimated second principal curvatures of the spine geometries. The signed second principal curvature, corresponding to the minimum curvature at each mesh point, is estimated using `GAMer`. Color bars correspond to curvature values with units of μm^{-1}. Geometries are A) single spine model, left: 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 μm, inlays: 200 nm. Curvature schematic modified from Wikipedia, credited to Eric Gaba (CC BY-SA 3.0).

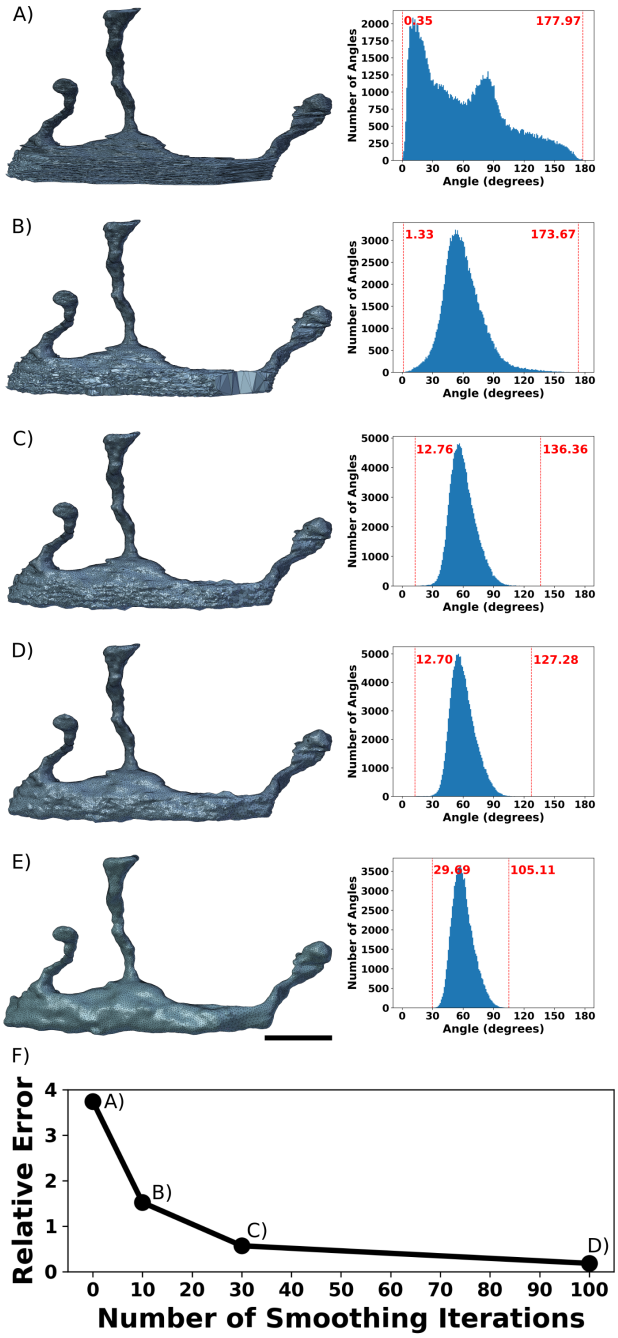

Fig 8. GAMer mesh conditioning reduces error in the result. (A-D) Mesh after 0, 10, 30, 100 smoothing operations (artifacts which prevented tetrahedralization, e.g. intersecting faces, were removed and the holes were re-meshed). E) shows the finalized mesh. F) Relative error was calculated using Eq. (28) using the final form of the mesh, (E), as the ground truth. Scale bar = $1\,\mu$m.

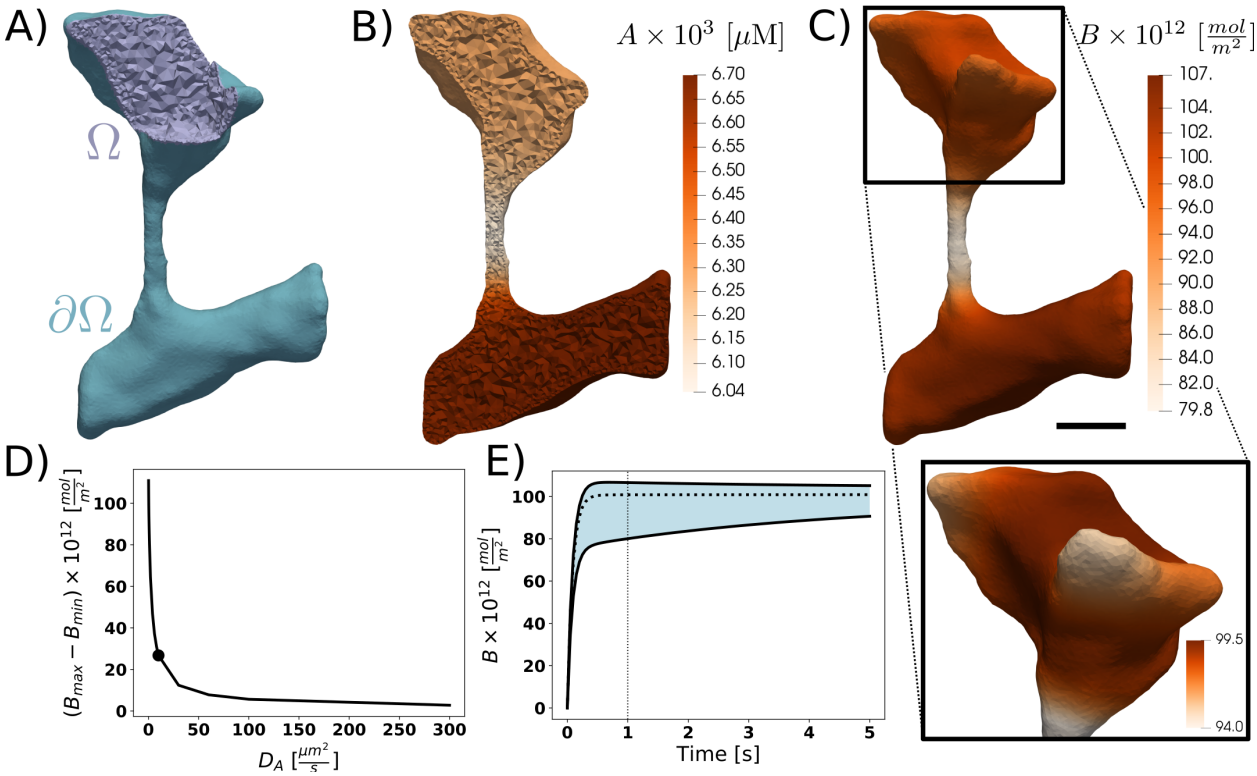

Fig 9. Simulations of coupled surface volume diffusion A) illustrates the domains for the volume and surface PDEs. B) and C) show the concentrations of species A and B, respectively, at $t = 1.0\,\mathrm{s}$ when D_A is set to $10\,\mu\mathrm{m}^2/\mathrm{s}$. D) Difference between maximum and minimum 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 highlighted. 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 $t = 1.0s$.

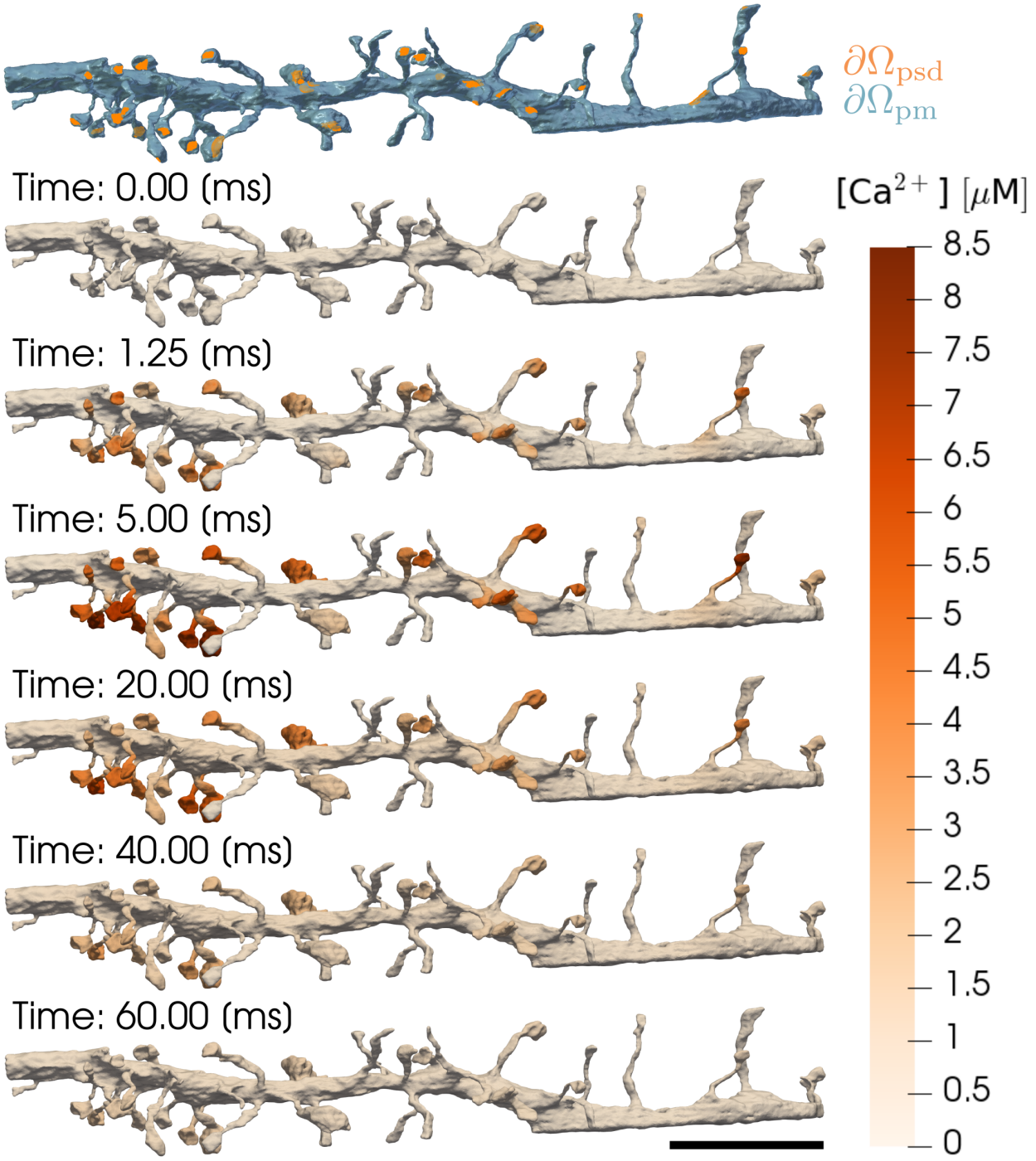

Fig 10. Time series of calcium dynamics from NMDA receptor opening in response to a prescribed membrane voltage trace in a full dendritic segment. Boundaries demarcating the plasma membrane (PM) and post synaptic density (PSD) are shown in blue and orange respectively (top). Snapshots of calcium ion 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. NMDA receptors localized at the PSD membrane, open in response to the voltage and calcium flows into the cell. Over time, the NMDA receptors close, and calcium is scavenged by calcium buffers. Scale bar: 4 μm.

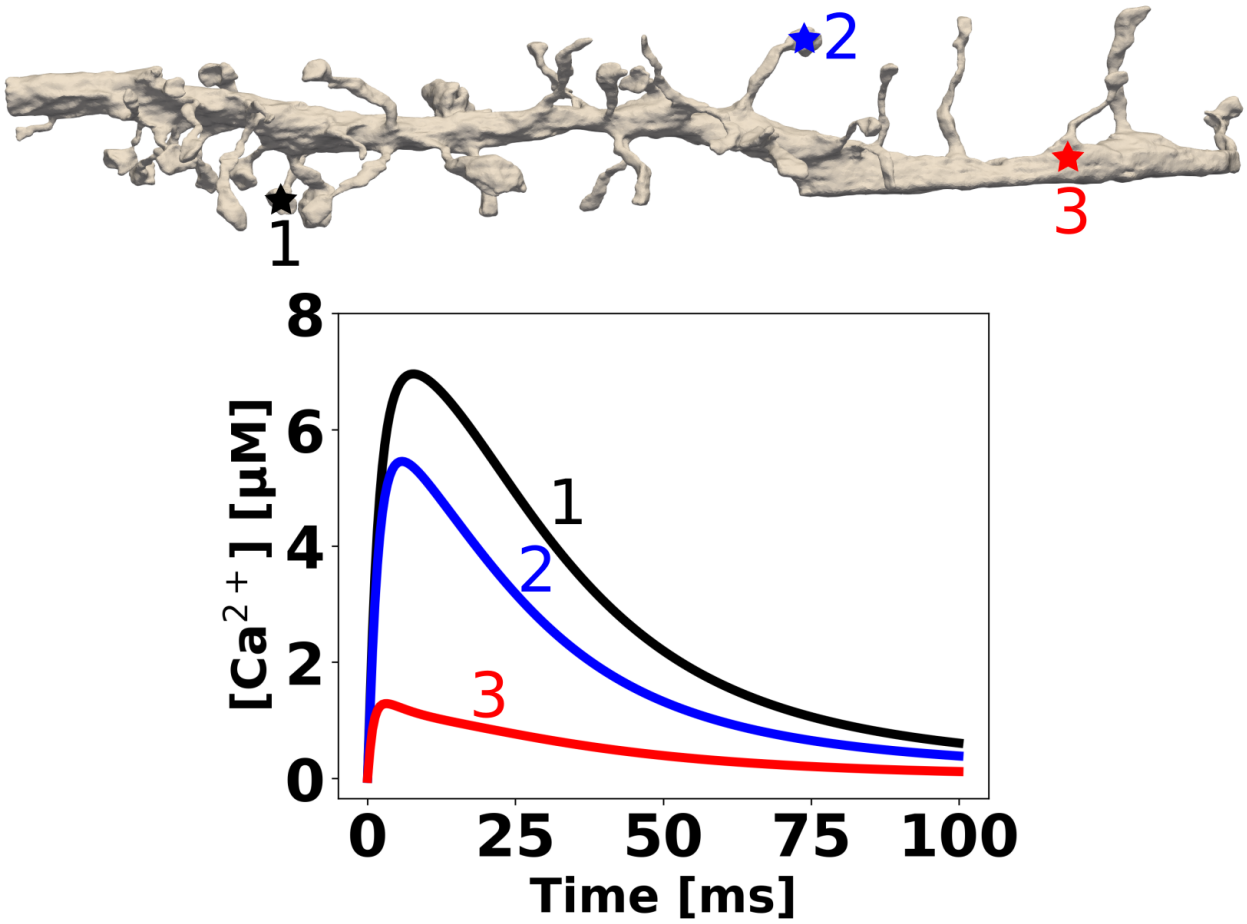

Fig 11. Representative traces of Ca^{2+} concentration over time at three positions. Spine and PSD morphology affect the calcium ion dynamics.