Jellyfish shape as a mechanical balance - Readme

Mengsha Gong, Minakshi Ashok, Ariane Helou, Lea Goentoro
Correspondence: goentoro@caltech.edu, mgong@caltech.edu

This document summarizes the equations the codes solve. The equations are discretized using the Forward Euler Method. Go to **example_usage.m** to start using the codes.

Mesh Initialisation

offset_mesh.m

The jellyfish mesh is initialized as a 10 mm circle with manually designated x, y coordinates with nodes spaced around 1 mm apart. Edges between nodes are also manually designated such that equilateral triangular elements are created when possible. A user-specified offset value slices the circle into two semicircles set apart by the offset value. The discussion that follows is specific to the mesh geometry initialized here.

Force balance equation

visco_offset_SLM_newmus.m

This code estimates force from stress, and then uses the net force to compute displacement. For each node i, this code calls:

1. **equilibrium_initial_KV.m** (optional) to initialise mesh strains and stresses.
2. **SLM_elastic.m** to estimate the magnitude of stress from the viscoelastic edges and compute direction of elastic force
3. **find_pressure.m** to estimate the magnitude of stress from pressure and compute direction of pressure force
4. **contraction5offset.m** to estimate the magnitude of stress on muscle nodes and compute direction of muscle contraction force
5. **remesh_SLM_newmus.m** to add or remove nodes that have strayed too far or too close.

Each code is described in detail in the next sections.

Next, for each node, the magnitudes of forces operating on the node were estimated by multiplying the stress estimates with the characteristic cross-sectional area, A_{scale},

$$F = \sigma A_{scale}$$

This way of estimating force from stress assumes that the cross-sectional area across edges does not vary very much. We used A_{scale} = 1 mm^2 as the edge lengths are distributed around 1 mm throughout the simulation, and the thickness of jellyfish tissue is in the order of 1 mm. Our thought process is that since material measurements in jellyfish are still sparse (and the available measurements vary by three orders of magnitude, see Table S1), we set out in this first model for order-of-magnitude estimates as opposed to exact fits. As more detailed measurements of jellyfish become available, we hope that this first model, which can already capture the overall dynamics and outcomes of shape reorganization, can act as a scaffold for adding more complexity.

For each node, the net magnitude and direction of the force were estimated by vectorially adding the forces. In computing the net force, we took into account that jellyfish do not stay contracted at all times, and apply muscle force only for the fraction of time the jellyfish is contracted (Equation S9),

$$\boldsymbol{F}_i = (\boldsymbol{F}_{m,i} + \boldsymbol{F}_{ve,i} + \boldsymbol{F}_{p,i})t_c + (\boldsymbol{F}_{ve,i} + \boldsymbol{F}_{p,i})t_r$$

where t_c is the fraction of the time during which the jellyfish contracts, and t_r is the fraction of time during which the jellyfish relaxes. The fraction of time the jellyfish contracted was estimated as follows:

- We measured from experiments that each muscle contraction lasts on average for 0.8 seconds (contraction duration, Figure S6d).
- Contraction rate is the parameter varied in the simulation.
- For a given contraction rate, the fraction of time the jellyfish spends contracted is,

$$t_R = \frac{1 - contraction\ rate * contraction\ duration}{contraction\ duration}$$

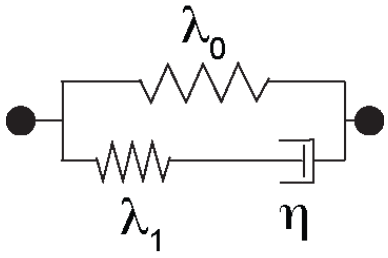
With the net force, the displacement of each node was computed using the force balance equation (Equation S2, and see derivation in the supplement):

$$\frac{d\boldsymbol{x}_i}{dt} = \frac{(\boldsymbol{F}_{m,i}+\boldsymbol{F}_{ve,i}+\boldsymbol{F}_{p,i})t_c+(\boldsymbol{F}_{ve,i}+\boldsymbol{F}_{p,i})t_r}{C} \qquad \text{(S2,9)}$$

Estimating stress from the viscoelastic response

SLM_elastic.m

The viscoelastic stress on node i is the vectorial sum of the viscoelastic stresses from the edges connecting to the node. The viscoelastic response of the edges is modeled as a Standard Linear Material (SLM), in which a spring acts in parallel to a Maxwell arm (which consists of a spring and dashpot in series), with the following implementations.



Following the standard equations of the SLM material, for each edge ij connecting nodes i and j, the magnitude of the viscoelastic stress is the sum of the stresses from the two springs,

$$\sigma_{ij} = \sigma_{ij}^0 + \sigma_{ij}^1$$

$$\sigma_{ij} = \lambda_0 \varepsilon_{ij} + \lambda_1 \varepsilon_{ij}^1$$

where the Hooke's law is used to compute the stress on the spring, and λ_0 and λ_1 are the elastic moduli of the springs. The strain of the elements is related to the total strain as follows,

Strain of spring 0: $\varepsilon_j^0 = \varepsilon_j$

Strain of spring 1: $\varepsilon_j^1 = \varepsilon_j - \varepsilon_j^v$

Strain of the viscous dashpot: $\frac{d\varepsilon_j^v}{dt} = \frac{\lambda_1}{\eta}\left(\varepsilon_j - \varepsilon_j^v\right)$

We tested implementing the viscoelastic equations in two ways. First, we made a simplification by assuming that during the time step Δt, the overall deformation ε_j changes very little, and estimated the strain in spring 1 using the Maxwell relaxation formula,

$$\varepsilon_j^1(t) \approx \varepsilon_j^1(t - \Delta t)\, exp(-\frac{\lambda_1}{\eta}\Delta t)$$

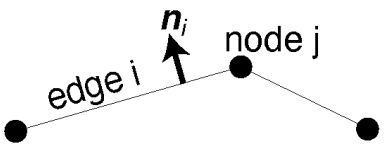
The advantage of using the simplification is that the effect of viscous dissipation is captured through an algebraic equation, as opposed to a differential equation. In addition, the simulation ran faster, which was helpful for scanning 6-7 orders of magnitude of parameter space for each of the material properties. The main figure in the Manuscript is plotted using this simplification.

Second, to assess this assumption, we directly solved the full equations. We verified that the constant-strain simplification, although it assumes more solidity in the system (5-10% increase in aspect ratios of the solutions), achieves the same overall shape solutions as the full solutions of the system.

The net magnitude of viscoelastic stress on node i was estimated by vectorially adding the stresses from the edges connected to the node. The direction of the viscoelastic force on an edge is taken as a unit vector parallel to the edge.

Estimating stress from pressure

find_f_pressure.m - to run the offset and butterfly simulations
find_f_pressure_initial.m - to run with **equilibrium_initial_KV.m** to estimate the initial condition.



This code estimates the magnitude of stress due to pressure. Pressure force on node j is the sum of pressure forces acting on the edges connected to the node. For each edge j, the magnitude of stress from pressure is estimated to scale with the change in pressure,

$$\Delta P = k_b\left(\frac{A(t)-A_0}{A_0}\right)$$

and the relative length of the edge,

$$\sigma_{p,i} = \frac{|l_i|}{l_{scale}}\Delta P = \frac{|l_i|}{l_{scale}} k_b\left(\frac{A(t)-A_0}{A_0}\right)$$

where k_b denotes bulk modulus (Pascal), A(t) the current area of the mesh, and A_0 the relaxed area of the mesh. This equation employs three approximations:

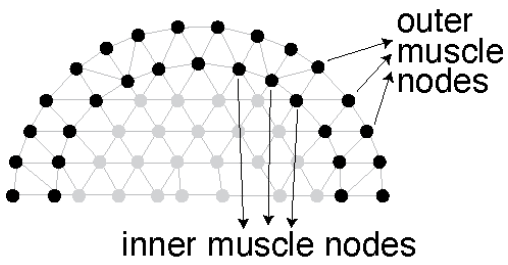
1. To compute the stress, change in area is used instead of change in volume assuming that thickness of jellyfish remains constant.
2. Pressure force is scaled with edge length because the magnitude of pressure force depends on the surface area on which it is acting, which since thickness is assumed to be constant, scales with edge length.
3. The relative length is used because of the two steps with which we estimate force. First, we estimated the stress scale (in this code). Then we estimate the force scale in SLM_viscoelastic.m by multiplying the stresses with characteristic cross-sectional area.

An alternative implementation would be to compute pressure force directly from each edge length in this code. Since, to begin with, (i) there are no available measurements in jellyfish to constrain the bulk modulus k_b; (ii) we are already simplifying the bulk pressure estimation by modeling the system as 2D; and (iii) to keep it consistent with how muscle and viscoelastic forces are estimated from the corresponding stresses, we chose the two-step implementation that was simplest for us.

The net magnitude of the stress on node i was estimated by vectorially adding the stresses from the edges connecting to the node. The direction of pressure force for each edge i was set as normal to the edge.

Estimating stress from muscle contraction

Contraction5offset.m – for offset simulations
Contraction5.m – for butterfly simulations



These codes estimate the magnitude of stresses due to muscle contraction and the direction of contraction.
In the beginning of the simulation, muscle nodes are defined based on the pattern we see in the staining experiments (as indicated by the green-shaded region in all the drawings in the main figures). For instance, for an intact medusa, muscle nodes are the outermost 2 rings of nodes. The edges that connect muscle nodes are correspondingly defined as muscle edges.

Once the simulation runs, at every time step, with each new location for the nodes, this code characterizes the new geometry of the muscle to estimate the magnitude and direction of the contraction force. The first half of the code goes through the outer muscle nodes, and then the same steps are applied to the inner muscle nodes.

Magnitude of stress from muscle contraction. To estimate the magnitude of muscle contraction force on node i, first we assume that the stress from muscle contraction has the following scale,

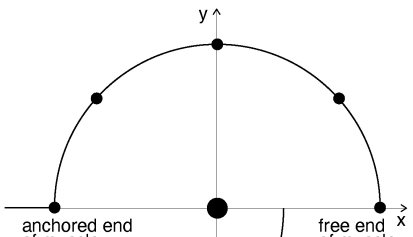
$$\sigma_{m,\,scale} \sim \lambda\, \varepsilon_{scale}$$

where λ is the elastic modulus (Pascal) and ε_{scale} is the characteristic strain in the system (unitless). We estimate ε_{scale} from the experiments, 15-20% (Figure S6b). For simplicity, we use as λ the sum of the elastic moduli of the springs ($\lambda_0 + \lambda_1$). The magnitude of muscle force on node i is then estimated using the following equation,

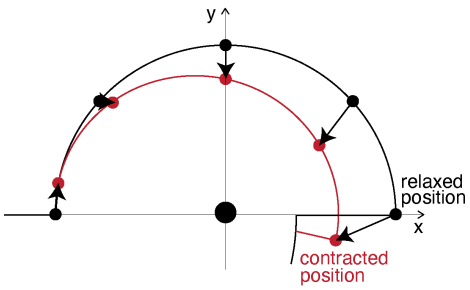
$$\sigma_{m,i} = k_i\, \sigma_{m,\,scale}$$

where k_i is a proportionality constant (unitless) that depends on how node i contracts, described next.

Estimating the contraction position of muscle node i. To compute the contracted position for a muscle node i, we follow the geometry of contraction observed in the experiments. The geometry of contraction is



simulated based on empirical measurements because, except for the simplest case of an intact, round medusa, the contraction shows asymmetry in strain that is not readily deducible from intuition alone. For instance, consider the offset graft, where in each half, one end of the muscle is anchored to the bulk tissue and another end is free.



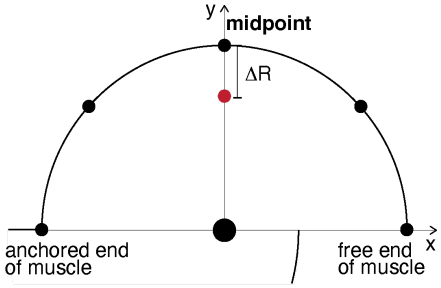
When the muscle contracts, the free end of the muscle contracts more than the anchored end. Our physical intuition for this asymmetry is as follows. It is not that total muscle force varies. However, muscle at the free end can produce a greater contraction *in the 2D plane we are considering* because there is less bulk tissue resisting it. By contrast, in the anchored end, as muscle contraction is pushing against the bulk tissue, a lot of the muscle force is lost to the direction perpendicular to the plane we are modeling, hence the smaller strain in the plane we are modeling.

In what follows, we use the offset muscle geometry as an example to illustrate how the code works, but the code considers 4 possible geometries relevant to the study:
Geometry 1: When one end of the muscle in anchored (muscle pieces in offset grafts)
Geometry 2: When both muscle ends are anchored (the middle muscle piece in butterfly graft).
Geometry 3: When no muscle ends are anchored (for half medusa, not used in graft simulations)
Geometry 4: When the muscle is a full ring (for intact medusa, not used in graft simulations)

The contracted position of each muscle node *i* is estimated as follows.
1. First, the code locates all muscle nodes, and defines **muscle piece** as a region in the mesh with continuous muscle nodes. For instance, an offset graft has two muscle pieces.

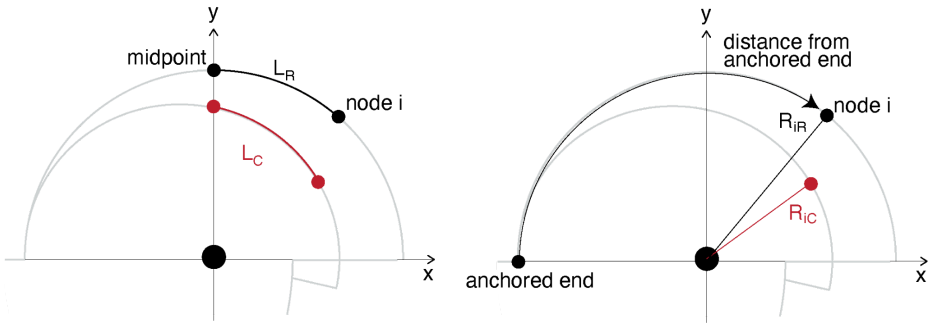
2. For each muscle piece, the code characterizes
- The length of the muscle piece (the sum of the edge lengths)
- The location of the end of the muscle piece
- Whether the end is anchored (connected to more than 3 edges) or free.
- The center of the muscle piece. The code circumscribes the two end-points and midpoint of the muscle piece by a circle. The center of the circle is defined as the center of the muscle piece toward which the muscle band nodes contract. The radius of this circle will be used to compute the radial distance of contraction.

3. With the geometry of the muscle piece approximated, the code computes the contracted position as follows.
- The code starts at the midpoint of the muscle, which is approximated to contract towards the center of the muscle piece (which is computed in step 2).



The contracted position of the midpoint is computed using the equation: displacement is computed as, $\Delta R=0.15$ x number of nodes from the anchored end, with a maximum value of $\Delta R=1.55$ mm.

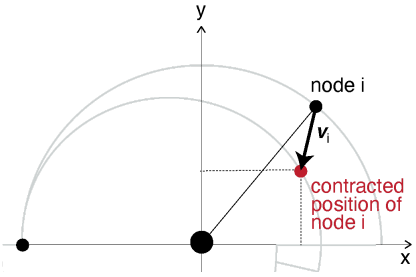
- Using the contracted position of the midpoint, the code computes the contracted position of other muscle nodes in the muscle piece, using the following rules:



$\frac{L_c - L_R}{L_R} = 0.25$, which implies $L_C = 0.75\, L_R$

$\Delta R = R_R - R_C = 0.15$ x number of nodes from the anchored end with a maximum value of $\Delta R = 1.55$ mm.

4. With the coordinates of contraction, the code computes the magnitude of muscle stress, by computing vector $\boldsymbol{v}_i$ as a vector going from the relaxed to the contracted position:



The magnitude of muscle stress is defined as proportional to the length of vector $\boldsymbol{v}_i$,

$$\sigma_{m,i} = k_i\, \sigma_{m,\,scale} = \frac{|v_i|}{l_{scale}}\, \sigma_{m,\,scale}$$

and use as characteristic length scale l_{scale} the initial edge size (1 mm). Essentially, we are making an approximation that the magnitude of muscle stress on node i varies from the characteristic stress scale in proportion to how much that node contracts relative to other nodes. The stresses in the inner muscle nodes are set as half of those in the outer muscle nodes,

$$\text{Outer muscle nodes } \sigma_{m,i} = \frac{|v_i|}{l_{scale}}\, \sigma_{m,\,scale}$$

$$\text{Inner muscle nodes } \sigma_{m,i} = 0.5\, \frac{|v_i|}{l_{scale}}\, \sigma_{m,\,scale}$$

We designed the muscle contraction force to be applied in this graded manner for stability. The muscle force is only applied on the nodes where the muscle is located and the effect of this compression is allowed to propagate into the bulk tissue through viscoelastic response in the edges. A large difference in force between one node and a neighboring node (e.g., at the boundary between muscle nodes and non-muscle nodes) can cause instability in the simulation.

5. The direction of the muscle force on node i is a unit vector parallel to the vector $\boldsymbol{v}_i$.

How the initial condition for the simulation was estimated

equilibrium_initial_KV.m

This code estimates the initial condition to use in the simulation. We implemented initial stress because we observe in the experiments that uncut jellyfish are under overall compression: if we make a cut on the surface, the cut sides will immediately splay apart.

We estimated the initial stresses ad hoc, by simulating the intact jellyfish for a very long time (1200 time steps, Δt=100 min; equivalent in real time of 80 days) until the system reaches an equilibrium. To facilitate this long run, we simplified the material model and simply simulated a spring moving in a viscous damper. To implement some initial stresses, we define the relaxed area as 0.1% higher than the mesh area, A_0 = 1.001 * total mesh area. The resulting stresses for each node was used as the initial condition in the offset and butterfly simulations.

Nonetheless, the solution does not appear to be very sensitive to the precise magnitudes of the initial stresses on the nodes, as we obtained nearly identical solutions when the relaxed area was varied by ten-fold (to 1% greater) or when the simulation was run from zero initial stresses.

Remeshing

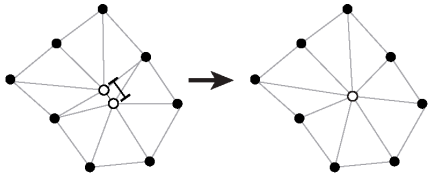
remesh_SLM_newmus.m

This code is called by **visco_offset_SLM_newmus.m** to check the mesh integrity and remesh nodes that have become too close or too far. Remeshing is performed every 5 hours, or if one of these two conditions are met:

1. If minimum edge length <0.35 mm.If there are nodes i and i+2 on the boundary that are <1 mm.

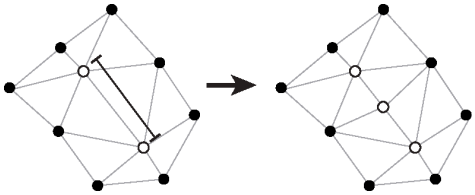
The initial distance between the nodes, that is, the edge length, is ~1 mm. As nodes displace, <u>the remeshing rules are:</u>

Rule 1. If the distance between two nodes <0.35 mm, replace the two nodes by a single node. The new node is created at the midpoint between the parent nodes and is connected to all neighbors of *either* parent node. Strain of edges is preserved when one edge is replaced by another. If a new edge is replacing two edges, the strain of the new edge is the average of the strain of the replaced edges.

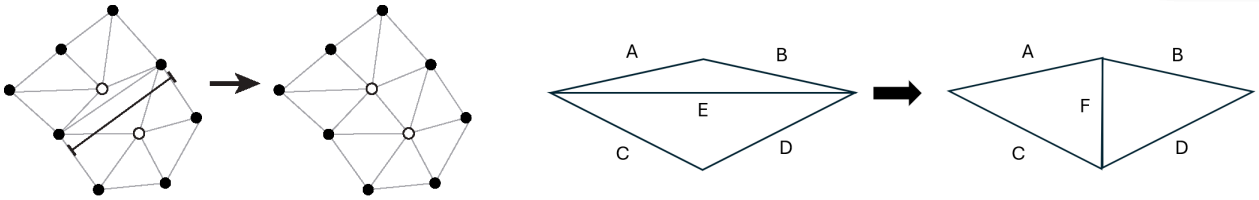


Rule 2. If the distance between two nodes >1.5 mm, create a new node at the midpoint. The new node is created at the midpoint between the parent nodes and is connected to neighbors of *both* parent nodes. The edges connecting the new node to the parent nodes inherit the strain of the

node that connected the parent nodes. The strain of edges connecting the new edge to other neighboring nodes is equal to the average of the strain of the edges connecting each parent to that neighboring node.

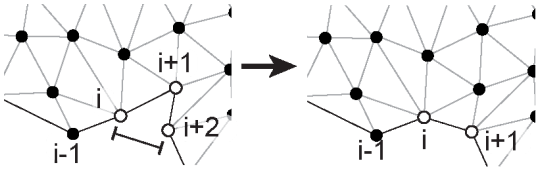


Rule 3. If a triangular element become too elongated (see figures below) such that the length of edge E becomes >90% of the sum of the lengths of edges A and B, and a neighboring element here composed of edges C, D, and E shares the long edge, edge E is eliminated and replaced by edge F.



The strain of the new edge F is the average of the strains of edges C and D. The strain of edge A is revised to be the mean of edges A and E, and the strain of edge B is revised to be the mean of edges B and E. If edge E was a muscle edge, A and B now inherit that muscle property.

Rule 4. When the distance between boundary nodes i and i+2 <1 mm, create a new boundary edge. The strain in the new boundary edge is the average of the strains of the two previous boundary edges. The previously boundary edges are reassigned as non-boundary.



Rule 5. Identities of the nodes and edges are inherited as follows,

1. If both parent nodes have the same identity, the child node inherits that identity.
2. If one of the parent nodes is boundary or muscle, the child node is boundary or muscle.
3. If a boundary or muscle edge is replaced by a new edge, that edge inherits boundary or muscle properties.
4. Length of muscle cannot decrease: If the length of an outer muscle band falls below 90% of its original length, additional boundary edges and nodes are assigned as muscle until the threshold is reached.