Supplementary Information for “Sensing the shape of a cell with reaction-diffusion and energy minimization”

Amit R. Singh, Travis Leadbetter, and Brian A. Camley

1. Parameter Values

We tabulate the default parameter values used in our simulations. Any changes from the default values have been explicitly mentioned in the main text.

| Parameter | Default Value | Description |
|-----------|---------------|-------------|
| $k_0$     | 0.07 s$^{-1}$ | Basal activation rate |
| $\gamma$ | 5 s$^{-1}$    | Positive feedback activation rate |
| $K$       | 2 $\mu$m$^{-2}$ | Saturation parameter |
| $\delta$ | 3 s$^{-1}$    | Deactivation rate |
| $D$       | 0.5 $\mu$m$^2$s$^{-1}$ | Diffusion coefficient of the active $\rho$ |
| $M/S$     | 2.9 $\mu$m$^{-2}$ | Scaled total concentration (see Eq. 3 in the main text) |
| $S_{10}$  | 627 $\mu$m$^2$ | Total surface area of the smooth surface |
| $S_{10}$  | 664 $\mu$m$^2$ | Total surface area of the surface with 10% roughness |
| $S_{20}$  | 768 $\mu$m$^2$ | Total surface area of the surface with 20% roughness |
| $A_{10}$  | 0.09 $S_{10}$ | Surface area constraint for PM model on the smooth surface |
| $A_{20}$  | 0.09 $S_{20}$ | Surface area constraint for PM model on the 10% roughness surface |
| $A_{30}$  | 0.09 $S_{30}$ | Surface area constraint for PM model on the 20% roughness surface |
| $k_1$     | $10^3$ $\mu$m$^{-1}$ | Penalty coefficient (see Eq. 4 in the main text) |
| $\beta$  | $10^{-2}$ $\mu$m | Gradient-descent step size coefficient (see Eq. 5 in the main text) |
| $T_{sim}$ | 5,000 s    | Simulation time ($D = 0.5\mu$m$^2$/s simulations) |
| $N_{elem}$ | 80000 | Number of mesh elements |

Table S1. Default values of the parameters used in the simulations of the PM and the WP models.

2. Transient States in Perimeter Minimization

In Fig. 4c in the main text, we noted that perimeter minimization predicts some three “steady” state along the $x = 0$ line that have been marked as red dots in the plot. The initial positions of the centroid of the domains in all of these cases were along the $x = 0$ line which is a line of symmetry. If we apply a small random perturbation of 0.02 $\mu$m to each point ($x_i$, $y_i$) of the final states of these simulations, they attain steady states on one of the peak or valleys. Thus, we confirm that these are indeed transient states and are an artefact of the symmetry of the surface. Fig. S1 show the trajectory plot after perturbing the final states.

![Fig. S1. The steady states and trajectories obtained from the perimeter minimization simulations on the smooth surface after applying a small perturbation to the final states along $x = 0$ line shown in Fig. 4 in the main text.](image)
3. Details of the Numerical Methods

A. Finite Element Model for Wave-pinning on a 3D surface. Our approach uses some standard finite element tools: this section will be easier to follow with knowledge of weak forms of PDEs, discretization of the domain of PDEs into a linear triangular mesh, the shape functions of a linear triangular element, and Gaussian quadrature on linear triangles to evaluate integrals defined over the elements.

We want to solve the following system of equations

\[
\frac{\partial a}{\partial t} = D \nabla^2 a + f(a, b)
\]

\[
b = a - \frac{1}{S} \int a
\]

using the Finite Element Method. Here, for conciseness, we’ve written the membrane-bound concentration as \(a\) and the uniform cytosolic concentration as \(b\). \(a = \frac{\Delta a}{\Delta t}\). We’ve also implicitly assumed \(\omega = 1\) here, but this can be generalized to nonzero \(\omega\) by taking \(S \to \omega S\) in the final equations. The first step for setting up the finite-element problem is to derive the “weak form” of the PDEs. We can express the time derivative using finite-difference. Let \(a\) be the concentration at time-step \(n\) and \(\bar{a}\) be the value of \(a\) at time-step \(n - 1\). Let the time step size be \(\Delta t\).

\[
a - \bar{a} = \Delta D \nabla^2 a + \Delta f(a, b)
\]

We will linearize the non-linear reaction function \(f(a, b)\) using \(a^{k+1} = a^k + \delta a\) and \(b^{k+1} = b^k + \delta b\). Here \(\delta a\) is the correction to the value of \(a^k\) at the \(k\)th Newton iteration and similarly for \(\delta b\).

\[
a^{k+1} - \bar{a} = \Delta D \nabla^2 a^{k+1} + \Delta f(a^{k+1}, b^{k+1})
\]

\[
a^{k} + \delta a - \bar{a} = \Delta D \nabla^2 a^k + \Delta D \delta a + \Delta f(a^k, b^k)\delta a + \Delta f(a^k, b^k)\delta b
\]

where \(f_a\) and \(f_b\) are the derivatives of \(f\) with respect to \(a\) and \(b\) respectively. Collecting the terms in \(\delta a\) and \(\delta b\) on the LHS, we get

\[
\left(1 - \Delta f_a(a^k, b^k)\right)\delta a - \Delta D \nabla^2 \delta a - \Delta f_b(a^k, b^k)\delta b = \bar{a} - a^k + \Delta D \nabla^2 a^k + \Delta f(a^k, b^k)
\]

Multiply both sides by \(u\), the test function, and integrate by parts setting the boundary terms to zero due to the boundary conditions (which will be either no flux or periodic).

\[
\int_\Omega \left(1 - \Delta f_a(a^k, b^k)\right) u \delta a \, d\Omega + \int_\Omega \nabla u \cdot \nabla \delta a \, d\Omega - \int_\Omega f_b(a^k, b^k) u \delta b \, d\Omega =
\]

\[
\int_\Omega u (\bar{a} - a^k) \, d\Omega - \int_\Omega \nabla u \cdot \nabla a^k \, d\Omega + \int_\Omega u f(a^k, b^k) \, d\Omega
\]

We will simplify the constraint equation using these linearizations, \(a = a^{k+1} = a^k + \delta a\) and \(b = b^{k+1} = b^k + \delta b\):

\[
b^k + \delta b = a - \frac{1}{S} \int_\Omega (a^k + \delta a) \, d\Omega
\]

Collecting \(\delta a\) and \(\delta b\) on the LHS, we get

\[
\left(\frac{1}{S} \int_\Omega \delta a \, d\Omega\right) + \delta b = a - b^k - \frac{1}{S} \int_\Omega a^k \, d\Omega
\]

The integral equations 1 and 2 are the weak forms of the original PDEs. Our goal is to solve for \(\delta a\) and \(\delta b\). But this is an infinite-dimensional problem because \(\delta a\) varies continuously over the surface although \(\delta b\) is uniform. FEM converts the infinite-dimensional problem to a finite-dimensional problem i.e. a system of equations. It does this by discretizing the domain of the system of PDEs into a discrete mesh of simple geometric entities called as the “finite elements” like triangles, or rectangles (or tetrahedrons and hexahedrons for higher-dimensional domains). In our problem, the domain \(\Omega\) is a surface. The surface is discretized into a mesh of triangles \(\Omega_e\) where \(e = 1, 2, \ldots\). Fig. S2 shows a sample surface domain discretized into linear triangular elements. The next step in FEM is to discretize all the field variables in all the integrands in the above equations into their values at the “nodes” of the finite element. Fig. S2 shows the nodes of a sample finite element mesh along with one possible way to index the nodes globally and locally. The number and location of nodes depends on the type of finite element being used. A node of the finite element mesh may be shared between multiple finite elements e.g. in Fig. S2 global node 1 is the same as local node 1 for triangle (1) and local node 3 for triangle (2). It is useful to define a mapping between the global and local indices of the nodes. Let \(E\) be the total number of elements in the mesh, \(G\) be the total number of nodes in the finite element mesh, and let \(N\) denote the number of nodes per element. Let \(I\) denote the global index of a node and \(I^{(e)}\) denote its local index in the \(e^{th}\) element. We can define a mapping between the global and local nodes

\[
\Lambda^{(e)} : [I]_{i=1}^{E} \rightarrow [I^{(e)}]_{i=1}^{N}
\]

or equivalently

\[
i^{(e)} = \sum_{i=1}^{G} \Lambda^{(e)}_i I \quad e = 1, 2, \ldots, E
\]
Fig. S2. A sample surface $\Omega$ which is the domain of a PDE to be solved using FEM is shown with its boundaries as bold lines. It has been discretized into a mesh of non-overlapping linear triangles $\Omega_e$, where $e = 1, 2, \ldots, 8$ shown with thin lines. The circular dots are the nodes of the mesh that are coincident with the vertices of the triangles. The bold numbers show the global indices of the nodes. The italic numbers show the local indices of the nodes in a triangle. The number in parenthesis indexes the triangles.
We can verify that

\[ \Lambda_{ni}^{(e)} = \begin{cases} 1 & \text{if global node } I \text{ is the local node } i \text{ of element } e \\ 0 & \text{otherwise} \end{cases} \]  

Clearly, \( \Lambda^{(e)} \) is a matrix of size \( N \times G \). As an example, in reference to Fig. S2, the global node index vector is \( I = [1, 2, 3, 4, 5, 6, 7, 8, 9]^T \). The local node index vector for triangle (5) is \( I = [5, 4, 8]^T \) that gives the global node indices for nodes 1, 2, and 3 of triangle (5). The mapping matrix \( \Lambda^{(5)} \) as per equations 3, and 4 is

\[
\Lambda^{(5)} = \begin{bmatrix}
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0
\end{bmatrix}.
\]

We can verify that

\[
\begin{bmatrix} 5 \\ 4 \\ 8 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{bmatrix} \]

As we noted above that in FEM the field variables are discretized to their nodal values. To obtain the value of the field variables at any position other than the nodes, interpolation is used. The interpolation functions are spatially varying and their exact form depends on the choice of the finite element. There is an interpolation function associated with each node of each element. These are called as the local interpolation functions. Let \( \phi_i^{(e)} \) denote the local interpolation function associated with the \( i \)th node of the \( e \)th element. These functions satisfy some useful properties like

\[
\phi_i^{(e)}(x, y) = 0 \quad \text{if } (x, y) \not\in \Omega_e
\]

\[
\phi_i^{(e)}(x^{(e)}, y^{(e)}) = \delta_{ij}
\]

\[
D^p \phi_i^{(e)}(x^{(e)}, y^{(e)}) = \delta_{ij}
\]

where \( \delta_{ij} \) is the Kronecker-delta which is 0 when \( i \neq j \) and 1 when \( i = j \). \( (x^{(e)}, y^{(e)}) \) are the coordinates of the \( j \)th node of the element \( e \). \( D^p \) indicates the \( p \)th order derivative with respect to \( x^{(e)} \) and \( y^{(e)} \) for each \( \phi_i^{(e)}(x, y) \). For example, a linear triangular finite element (that we have used in our simulations) with element index \( e \) has three nodes that coincide with its vertices as shown in Fig. S2. If the vertices are located at \((x_1, y_1), (x_2, y_2), \) and \((x_3, y_3)\), the corresponding local interpolation functions are

\[
\phi_1^{(e)}(x, y) = \frac{x(y_2 - y_3) + x_2y_3 - x_3y_2 - y(x_2 - x_3)}{x_1y_2 - x_1y_3 - x_2y_1 + x_2y_3 + x_3y_1 - x_3y_2}
\]

\[
\phi_2^{(e)}(x, y) = \frac{x(y_1 - y_3) - x_1y_3 + x_3y_1 + y(x_1 - x_3)}{x_1y_2 - x_1y_3 - x_2y_1 + x_2y_3 + x_3y_1 - x_3y_2}
\]

\[
\phi_3^{(e)}(x, y) = \frac{x(y_1 - y_2) + x_1y_2 - x_2y_1 + y(x_1 - x_2)}{x_1y_2 - x_1y_3 - x_2y_1 + x_2y_3 + x_3y_1 - x_3y_2}
\]

The local interpolation functions can be “assembled” together using the index mapping shown in equation 3 into global interpolation functions such that there is one interpolation function associated with each global node of the finite element mesh. The global interpolation function associated with the \( I \)th node is

\[
\psi_I(x, y) = \sum_{e=1}^{E} \sum_{i=1}^{N} \Lambda_{ni}^{(e)} \phi_i^{(e)}(x, y).
\]

Finally, we are ready to discretize the field variables as interpolations of nodal values over the entire mesh using the global interpolation functions of equation 7. We will use capital letters to denote the nodal values e.g. \( F_{\omega} \) will denote the value of \( f_I(x, y) \) on the \( I \)th global node.
We will use the following substitutions:

\[ f_a (a(x,y), b) = \sum_{i=1}^{G} F_{ai}(x,y) \]
\[ f_b (a(x,y), b) = \sum_{m=1}^{G} F_{bm}(x,y) \]
\[ \delta a(x,y) = \sum_{i=1}^{G} \delta A_i \psi_i(x,y) \]
\[ \nabla \delta a(x,y) = \sum_{i=1}^{G} \delta A_i \nabla \psi_i(x,y) \]
\[ \bar{a}(x,y) = \sum_{n=1}^{G} \bar{A}_n \psi_n(x,y) \]
\[ a^k(x,y) = \sum_{n=1}^{G} A^k_n \psi_n(x,y) \]
\[ \nabla a^k(x,y) = \sum_{p=1}^{G} A^k_p \nabla \psi_p(x,y) \]
\[ f(a^k(x,y), b^k) = \sum_{q=1}^{G} F_{qk}(x,y) \]

The nodal values \( \delta A_i, i = 1, 2, \ldots, G \) are the unknowns that we have to solve for. Therefore, we need \( G \) equations to solve for the \( G \) unknowns. We generate these equations by setting \( u = \psi_j \) where \( j = 1, 2, \ldots, G \) in equation 1. Using the above substitutions and setting \( u = \psi_j \), equations 1, and 2 can be written as

\[
\int_\Omega \left( 1 - t \sum_i F_{ai} \psi_i(x,y) \right) \psi_j(x,y) \sum_i \delta A_i \psi_i(x,y) d\Omega + tD \int_\Omega \nabla \psi_j(x,y) \cdot \sum_i \delta A_i \nabla \psi_i(x,y) d\Omega - t \int_\Omega \psi_j(x,y) \sum_m F_{bm} \psi_m(x,y) d\Omega \delta b d\Omega \]

\[ = \int_\Omega \psi_j(x,y) \left( \sum_n (\bar{A}_n - A^k_n) \psi_n(x,y) \right) d\Omega - tD \int_\Omega \nabla \psi_j(x,y) \cdot \sum_p A^k_p \nabla \psi_p(x,y) d\Omega + t \int_\Omega \psi_j(x,y) \sum_q F_{qk} \psi_q(x,y) d\Omega \]

\[
\frac{1}{S} \int_\Omega \sum_i \delta A_i \psi_i(x,y) d\Omega + \delta b = \alpha - b^k - \frac{1}{S} \int_\Omega \sum_m A^k_m \psi_m(x,y) d\Omega \]

We can rearrange the above equations as

\[
\sum_i \delta A_i \left( \int_\Omega \left( 1 - t \sum_i F_{ai} \psi_i(x,y) \right) \psi_j(x,y) \psi_j(x,y) d\Omega + tD \nabla \psi_j(x,y) \cdot \nabla \psi_j(x,y) d\Omega \right) - \delta b \left( t \int_\Omega \psi_j(x,y) \sum_m F_{bm} \psi_m(x,y) d\Omega \right) = \sum_n (\bar{A}_n - A^k_n) \int_\Omega \psi_n(x,y) \psi_j(x,y) d\Omega - \sum_p A^k_p \left( tD \int_\Omega \nabla \psi_p(x,y) \cdot \nabla \psi_j(x,y) d\Omega \right) + t \int_\Omega \psi_j(x,y) \sum_q F_{qk} \psi_q(x,y) d\Omega \]

\[
\sum_i \delta A_i \left( \frac{1}{S} \int_\Omega \psi_i(x,y) d\Omega \right) + \delta b = \alpha - b^k - \frac{1}{S} \int_\Omega \sum_m A^k_m \psi_m(x,y) d\Omega \]

We can write the above equations as

\[
\sum_i P_{ij} \delta A_i + Q_j \delta b = R_j \]
\[
\sum_i S_i \delta A_i + \delta b = T \]
where

\[ P_{ij} = \int_{\Omega} \left( 1 - t \sum_{l} F_{il} \phi_i(x,y) \right) \phi_j(x,y) \psi_j(x,y) \, d\Omega + tD \int_{\Omega} \nabla \phi_i(x,y) \cdot \nabla \psi_j(x,y) \, d\Omega \]

\[ Q_j = -t \int_{\Omega} \psi_j(x,y) \sum_{m} F_{jm} \psi_m(x,y) \, d\Omega \]

\[ R_j = \sum_{l} (A^l_j - A^l_k) \int_{\Omega} \phi_l(x,y) \psi_j(x,y) \, d\Omega - \sum_{m} A^k_m \left( tD \int_{\Omega} \nabla \psi_m(x,y) \cdot \nabla \psi_j(x,y) \, d\Omega \right) + t \int_{\Omega} \phi_l(x,y) \sum_{q} F_{q} \psi_q(x,y) \, d\Omega \]

\[ S_j = \frac{1}{S} \int_{\Omega} \psi_j(x,y) \, d\Omega \]

\[ T = \alpha - b_1 - \frac{1}{S} \sum_{m} A^k_m \psi_m(x,y) \, d\Omega \]

Equation 12 can also be written as a block matrix equation

\[
\begin{pmatrix}
[P] & [Q] \\
[S] & 1
\end{pmatrix}
\begin{pmatrix}
\delta A \\
\delta b
\end{pmatrix}
= \begin{pmatrix}
[R] \\
T
\end{pmatrix}
\]

All the integrals in the definition of the matrices above will be calculated using Gaussian Quadrature over triangles. In general, for solving the above system of equations on a 3D surface, the Laplacian operator in the original PDEs has to be the Laplace-Beltrami operator. But if we discretize the 3D surface using linear triangles we can use the regular Laplacian operator (1). We use FEniCS (2) for implementing the finite element method. FEniCS provides various solvers for solving the linearized system of equations that we obtained above. We use the default LU decomposition solver.

**B. Perimeter Minimization Model.** In our perimeter minimization model of cell polarity, the domain moves to locally minimize the cell’s perimeter, while keeping the area constant, i.e. minimizing Equation 4 in the main text. This energy $F$ is a function of the domain perimeter $L$ and its area $A$. To describe the domain dynamics we need to be able to compute the length of a closed curve embedded in the 3D surface, and the surface area enclosed by the curve. We will also need the derivatives of the length and the area with respect to the points parameterizing the domain shape. In the following text, we show how to compute these. We will derive these results using the first fundamental form from differential geometry.

Consider a surface $\Omega$ whose points are defined as

\[ \Omega = \{(x, y, h(x,y)) \mid (x,y) \in \mathbb{R}^2 \text{ and } h \in \mathbb{R}\} \]

We define a piecewise continuous closed curve $\gamma$ embedded in the surface $\Omega$. Let its projection in the $x$-$y$ plane be $\Gamma$. We will parameterize $\Gamma$ using polar-coordinates centered at a point $(a, b)$ in the interior of $\Gamma$ in the $x$-$y$ plane.

\[ \Gamma(\theta) = (a + r(\theta) \cos \theta, b + r(\theta) \sin \theta) \]

We choose $r(\theta)$ to be a $N$ segment piecewise linear function of $\theta$ such that the $k^{th}$ segment of $r(\theta)$ is

\[ r^{(k)}(\theta) = r_k + \frac{r_{k+1} - r_k}{\theta_{k+1} - \theta_k} (\theta - \theta_k) \quad \text{where } k = 1, 2, ..., N. \]

Since $\Gamma$ is a closed curve we have an effective periodic boundary condition, $r_{N+1} = r_1$ and $\theta_{N+1} = \theta_1$, i.e.

\[ r^{(N)}(\theta) = r_N + \frac{r_1 - r_N}{\theta_1 - \theta_N} (\theta - \theta_N). \]

In the above equations we have used

\[ r_k = r(\theta_k) \quad \text{where } \theta_k = \frac{2\pi(k-1)}{N}. \]

The values of $r^k$ are the unknown parameters of our optimization problem. The coordinates $(a, b)$ are calculated as the mean of the $x$ and $y$ coordinates calculated of from the values of $r^k$ and $\theta^k$ after every iteration of the gradient-descent algorithm.

A point $X$ on the surface $\Omega$ in the interior of $\gamma$ can be written as

\[ X(\theta) = [x(r(\theta), \theta), \ y(r(\theta), \theta), \ h(x,y)]^T \]

The coefficients of the first fundamental form can be obtained as

\[ E = \frac{\partial X}{\partial r} \cdot \frac{\partial X}{\partial r} \]

\[ F = \frac{\partial X}{\partial r} \cdot \frac{\partial X}{\partial \theta} \]

\[ G = \frac{\partial X}{\partial \theta} \cdot \frac{\partial X}{\partial \theta} \]
**B.1. Length.** Length of the $k^\text{th}$ segment of the curve $\gamma$ is

$$L^{(k)} = \int_{\theta_k}^{\theta_{k+1}} \sqrt{E r' + 2 F r' + G} \, d\theta$$

where $r' = \frac{\partial r^{(k)}}{\partial \theta} = \frac{r_{k+1} - r_k}{\theta_{k+1} - \theta_k}$. Let $f(r, r'; \theta) = \sqrt{E r' + 2 F r' + G}$ where $E$, $F$ and $G$ are functions of $r$. Then using 1-point Gaussian quadrature we can write

$$L^{(k)} \approx (\theta_{k+1} - \theta_k) f\left(\frac{\theta_{k+1} + \theta_k}{2}\right).$$

It should also be noted that, because we are choosing $r(\theta)$ to be piecewise linear, we have

$$r\left(\frac{\theta_{k+1} + \theta_k}{2}\right) = \frac{r_{k+1} + r_k}{2}.$$  \[13\]

Using chain-rule and product rule of differentiation,

$$\frac{\partial L^{(k)}}{\partial r_k} = (\theta_{k+1} - \theta_k) \left( \frac{\partial f}{\partial r} + \frac{\partial f}{\partial r'} \frac{\partial r'}{\partial r_k} \right).$$

Therefore,

$$\frac{\partial L^{(k)}}{\partial r_k} = \left( \frac{\theta_{k+1} - \theta_k}{2} \right) \left( \frac{\partial f}{\partial r} + \frac{\partial f}{\partial r'} \frac{\partial r'}{\partial r_k} \right).$$

Similarly, we can show that

$$\frac{\partial L^{(k)}}{\partial r_{k+1}} = \left( \frac{\theta_{k+1} - \theta_k}{2} \right) \left( \frac{\partial f}{\partial r} + \frac{\partial f}{\partial r'} \frac{\partial r'}{\partial r_{k+1}} \right).$$

**B.2. Area.** Area of the $k^\text{th}$ triangular slice of the region bounded by $\gamma$ can be obtained as

$$A^{(k)} = \int_{\gamma_k}^{\gamma_{k+1}} \sqrt{E G - F^2} \, dr \, d\theta$$

We will use 1-point Gaussian quadrature along $\theta$ which gives

$$A^{(k)} \approx (\theta_{k+1} - \theta_k) \int_0^{\frac{\Delta \theta_{k+1}}{2}} \sqrt{E G - F^2} \, dr$$

We will use a 5-point Gaussian quadrature along $r$. Let's use $g \equiv \sqrt{E G - F^2}$.

$$A^{(k)} \approx (\theta_{k+1} - \theta_k) \int_0^{\frac{\Delta \theta_{k+1}}{2}} \sqrt{E G - F^2} \, dr$$

where $w_i$ are the Gaussian quadrature weights and $\xi_i$ are the corresponding quadrature points. We can now write

$$\frac{\partial A^{(k)}}{\partial r_k} = \left( \frac{\theta_{k+1} - \theta_k}{4} \right) \sum_i w_i g_i + \left( \frac{\theta_{k+1} - \theta_k}{4} \right) v_i \left( \frac{r_{k+1} + r_k}{4} \right) \sum_i g_i \frac{\partial g_i}{\partial r_k}$$

We have to evaluate $g_i$ at $r = (1 + \xi_i)(r_{k+1} + r_k)/4$, therefore $\partial g_i/\partial r_k = (1 + \xi_i)/4$,

$$\frac{\partial A^{(k)}}{\partial r_k} = \left( \frac{\theta_{k+1} - \theta_k}{4} \right) \sum_i w_i g_i + \left( \frac{\theta_{k+1} - \theta_k}{4} \right) v_i \left( \frac{r_{k+1} + r_k}{4} \right) \sum_i (1 + \xi_i) w_i \frac{\partial g_i}{\partial r_k}$$

Similarly, it can be shown that

$$\frac{\partial A^{(k)}}{\partial r_{k+1}} = \frac{\partial A^{(k)}}{\partial r_k}$$

Together, these formulas let us compute the domain perimeter and its area by summing over the areas and lengths in each triangular slice $k$. Similarly, we can use the derivatives of the area for each slice in order to evolve the gradient descent, Eq. 5 in the main text.

**Supplementary Movie Captions**

**Movie S1.** Evolution of wave-pinning model on the sinusoidal surface showing the migration of a domain from its initial position to the final position at the domain peak. View is from the top.

**References**

1. ME Rognes, DA Ham, CJ Cotter, ATT McRae, Automating the solution of PDEs on the sphere and other manifolds in FEniCS 1.2. *Geosci. Model. Dev.* 6, 2099–2119 (2013).
2. M Alnæs, et al., The FEniCS project version 1.5. *Arch. Numer. Softw.* 3 (2015).