Supplemental Information

Materials and Methods

Image preprocessing

Before applying vector diffusion maps, images must be preprocessed so that the Euclidean distance between the image pixels is informative: we need images who are developmentally similar to have relatively small Euclidean distances, and images who are at disparate developmental time points to have relatively large distances. We must preprocess images to remove any experimental and imaging artifacts, so that most of the variation is relevant to the developmental dynamics.

The relevant image operations are listed below for our general purposes, along with the relevant MATLAB functions.

Intensity normalization Contrast-limited adaptive histogram equalization (using the `adapthisteq` function with an 8×8 tile grid, and a uniform distribution for the intensities with a clip limit of 0.01) is used to normalize the intensities of signals whose absolute intensity is not meaningful/informative.

Blur The `imfilter` function with a disc filter is used to blur signals whose small-scale structure is not informative.

Intensity scaling (multichannel images only) The `immultiply` function is used to scale signal intensities. This is important for multichannel images, as it determines the (relative) contributions of the various signals.

Mean-centering The Canny method (Canny, 1986) is used to detect the edges of the object in each image (using the `edge` function). The image is then translated so that the object (as determined by extremeties of the detected edges) is centered.

Size scaling For images whose relative size is unimportant to the developmental dynamics, the images are rescaled/dialated so that the object size (as determined by extremeties of the detected edges) is constant (we set this to be 80% of the total image).

Removing corners The image is cropped/filtered using a disc centered in the middle of the image and whose diameter is equal to the number of pixels; any pixels outside of this disc are set to 0 (no intensity). This removes any corner effects under rotations.

Drosophila gastrulation (live)

The original image resolution is 512×512 for the live *Drosophila* embryo images. All images were subsampled to 100×100 pixels for analysis, as this was found to be a sufficient resolution to retain all of the major developmental features within the data set. Images were normalized, and then blurred with a filter of radius 5% of the total image (5 pixels). Images were not mean-centered or rescaled, as the entire live imaging data set was approximately centered already.

Zebrafish epiboly

The original image resolution was 320×288 for the zebrafish images. A 16-pixel border was removed from the left and right sides to make the images square, and all images were then subsampled to 100×100 pixels for analysis, as this was found to be a sufficient resolution to retain all of the major developmental features within the data set. Images were not normalized or blurred. Images were mean-centered so that the embryo in each frame was (approximately) centered. Images were not rescaled for size, as changes in overall size are important.

Drosophila gastrulation (fixed)

The original image resolution was 1024×1024 for the fixed *Drosophila* images. All images were subsampled to 100×100 pixels for analysis, as this was found to be a sufficient resolution to retain all of the major developmental features within the data set. The nuclei channel was normalized, and all channels were blurred with a filter of radius 5% of the total image (5 pixels) to remove the effects of individual nuclei. The nuclei channel was scaled by half; because this signal occupies a larger fraction of the image relative to the other signals, its overall contribution is large, and so we downscaled it so that each signal would have more comparable weight in the algorithm. The images were mean-centered and scaled to have a constant size using the nuclei signal to detect the edges of the embryo within the frame.

Drosophila wing discs

The original image resolution was 1024×1024 for the wing disc z-stacks. Each of the original wing disc z-stacks contains 30–40 images. All images were subsampled to 100×100 pixels for analysis, as this was found to be a sufficient resolution to retain all of the major developmental

features within the data set. Each wing disc z-stack was reduced to a 21-image stack, consisting of the brightest image and the 10 images above and below. No channels were normalized or blurred in the images, and the channel intensities were kept at their imaging values. Images were mean-centered using the Wingless/Patched signal to detect the edges of the sample. Images were not rescaled for size, as changes in overall size are important.

Algorithms

We demonstrate the algorithms for registration and temporal ordering using a synthetic data set. The relatively simple dynamics of this data set allows us to easily visualize and illustrate the main features of the different algorithms. Motivated by the geometry of our *Drosophila* embryo images, we construct a sequence of concentration profiles defined on a ring, and rotate each ring randomly around its center; an example is shown in Fig. S2*A*. Rotation of the ring corresponds to shifting (with periodic boundary conditions) the one-dimensional concentration profile shown at the bottom of Fig. S2*A* (the symmetry group is $SO(2)$, the group of all two-dimensional proper rotations). Each concentration profile is a noisy Gaussian (shown in Fig. S2*B*), and the Gaussians increase in intensity as a function of "time". We discretize the profiles into 100 points, so our numerical data will be 100-dimensional vectors (the corresponding symmetry group for the discretized profiles is $\mathbb{Z}_{100}$, the group of integers modulo 100). Fig. S2*C* shows the entire data set; the concentration profiles have been stacked in an array, so that each row corresponds to a single profile. Because the profiles are unregistered and unordered, the underlying dynamics (a Gaussian whose amplitude grows in time) are not readily apparent.

Angular synchronization (Singer, 2011)

Let $x_1, \ldots, x_m$ denote the signals that we wish to align with respect to rotations; each signal is a function defined on the unit circle (on the plane). First assume that each signal x_i is a *noisy* rotated copy of the underlying signal x_{true} (which we are *not* given), such that

$$x_i = f(x_{true}, \theta_i) + \xi_i \tag{1}$$

where the function $f(x_{true}, \theta_i)$ rotates the signal x_{true} by θ_i degrees, and ξ_i is a (typically Gaussian) noise term. Our goal is to recover $\theta_1, \ldots, \theta_m$. Up to noise,

$$x_i \approx f(x_j, \theta_i - \theta_j); \tag{2}$$

note that (2) does not require knowledge of x_{true}. We can obtain an *estimate* of $\theta_i - \theta_j$ by computing the rotation that optimally aligns x_j to x_i, i.e.,

$$\theta_i - \theta_j \approx \theta_{ij} = \arg\min_{\theta} \|x_i - f(x_j, \theta)\|^2. \tag{3}$$

Practically, the signals are discretized in a n-long vector (the local intensity at n equidistant points around the circle); rotating the function by an angle θ then corresponds to cyclically shifting the elements of x_i by $\frac{\theta_i}{2\pi} n$ (rounded to the nearest integer to obtain a valid shift). For the one-dimensional discretized profiles shown in Fig. S2, we exhaustively search over all $n = 100$ possible shifts of the signals to obtain the optimal angles in (3). Alternatively, for continuous signals, an optimization algorithm can be used (Ahuja et al., 2007).

Rather than work with the angles θ_{ij} directly, it is more convenient to consider the rotation matrices,

$$R(\theta_{ij}) = \begin{bmatrix} \cos(\theta_{ij}) & -\sin(\theta_{ij}) \\ \sin(\theta_{ij}) & \cos(\theta_{ij}) \end{bmatrix}, \tag{4}$$

which we can think of as operating on the points of the unit circle (on the plane) on which our signal is defined. Successive rotations correspond to multiplication of the corresponding rotation matrices: $R(\alpha_1 + \alpha_2) = R(\alpha_1)R(\alpha_2)$. Due to the orthogonality of rotation matrices, $R(-\alpha) = R(\alpha)^T$.

Let d denote the dimension of the rotation matrices we are considering (for planar rotations, $R(\theta_{ij}) \in \mathbb{R}^{2\times 2}$ and $d = 2$). We construct the matrix $H \in \mathbb{R}^{md \times md}$, where H is an $m \times m$ matrix of $d \times d$ blocks, with the i, j^{th} block of H, H_{ij}, defined as

$$H_{ij} = R(\theta_{ij}). \tag{5}$$

Under our assumption that $\theta_{ij} \approx \theta_i - \theta_j$, $H_{ij} \approx R(\theta_i)R(\theta_j)^T$ and

$$H \approx \begin{bmatrix} R(\theta_1) \\ R(\theta_2) \\ \vdots \\ R(\theta_m) \end{bmatrix} \begin{bmatrix} R(\theta_1)^T R(\theta_2)^T \ldots R(\theta_m)^T \end{bmatrix}. \tag{6}$$

It follows directly from (6) that the top block eigenvector of H contains our best estimates of $R(\theta_1), R(\theta_2), \ldots, R(\theta_m)$. Let $\phi_1, \phi_2, \ldots, \phi_{md}$ denote the eigenvectors of H ordered so that $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_{md}|$, where λ_i is the eigenvalue corresponding to ϕ_i. Then,

$$\hat{R} = \begin{bmatrix} \hat{R}_1 \\ \hat{R}_2 \\ \vdots \\ \hat{R}_m \end{bmatrix} = \begin{bmatrix} | & | & & | \\ \phi_1 & \phi_2 & \ldots & \phi_d \\ | & | & & | \end{bmatrix}, \tag{7}$$

where $\hat{R}_i \in \mathbb{R}^{d\times d}$ is (nearly) the estimate for $R(\theta_i)$. To obtain our estimate of $R(\theta_i)$, denoted $R_{i,est}$, we project $\hat{R}_i$ onto the closest orthogonal matrix,

$$R_{i,est} = U_i V_i^T, \tag{8}$$

where U_i and V_i are the left and right singular vectors, respectively, of $\hat{R}_i$. We adjust the sign of ϕ_1 so that $det(R_{i,est}) = +1$, ensuring proper rotations (note that

systematically incorporating improper rotations is also possible (Goemans and Williamson, 1995; Bandeira et al., 2013)). We estimate θ_i by inverting (4), and register the signals by rotating signal i by $-\theta_i$. We note that, in our actual computations, the pairwise rotations θ_{ij} are computed in a discrete setting, then the overall synchronization is performed in the continuum context to obtain θ_i, and the results are rounded to give the closest discrete shift.

Importantly, this formulation also considers *higher-order* consistency information. For example, given our pairwise estimates R_{ij}, we know that relationships of the form

$$R(\theta_{ik})R(\theta_{kj}) \approx R(\theta_i)R(\theta_k)^T R(\theta_k)R(\theta_j)^T = R(\theta_i)R(\theta_j)^T \tag{9}$$

should also hold. Note that

$$(H^2)_{ij} = \sum_k R(\theta_{ik})R(\theta_{kj}); \tag{10}$$

therefore, *all* information of the form in (9) is contained in the matrix H^2 (and higher order consistency information in its higher powers). Because H and H^2 have the same eigenvectors, our problem formulation accounts for not only pairwise alignment information, but also these higher-order considerations.

Diffusion maps (Coifman et al., 2005)

Given m data points $x_1, \ldots, x_m$ (typically vectors in a high-dimensional vector space), we want to find a coordinate transformation $y(x)$ that preserves local geometry: points that are "close" in the original space should also be "close" in the coordinates y. The first step is to construct the matrix $W \in \mathbb{R}^{m\times m}$, where W_{ij} is large if points x_i and x_j are "close." We use a diffusion kernel,

$$W_{ij} = \exp\left(-\frac{d^2(x_i, x_j)}{\epsilon^2}\right), \tag{11}$$

where $d(x_i, x_j)$ is a pairwise distance between x_i and x_j (often the Euclidean distance), and ϵ is a characteristic scale. Points less than ϵ apart are thus considered "close" and points farther than ϵ apart are considered "far away". ϵ can be chosen using several techniques (see, for example (Coifman et al., 2008; Rohrdanz et al., 2011)); here, we take ϵ to be 1/4 of the median of the pairwise distances for the two-dimensional images, and 1/2 of the median of the pairwise distances for the three-dimensional z-stacks.

To find the coordinate y, we want solve the following optimization problem (Belkin and Niyogi, 2003)

$$\arg\min_y \sum_{ij} W_{ij}(y(x_i) - y(x_j))^2. \tag{12}$$

We first compute the diagonal matrix D, where $D_{ii} = \sum_{j=1}^m W_{ij}$, and the matrix A, where

$$A = D^{-1}W. \tag{13}$$

We calculate the eigenvectors $\phi_1, \phi_2, \ldots, \phi_m$, ordered such that $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_m|$. Because the matrix A is similar to the symmetric matrix $D^{-1/2}WD^{-1/2}$, A is guaranteed to have real eigenvalues and real, orthogonal eigenvectors. Because the matrix A is row-stochastic, $\lambda_1 = 1$ and ϕ_1 is a constant vector; this is a trivial solution to (12). The next eigenvector, ϕ_2, is the (non-trivial) solution to (12), so that $\phi_2(j)$, the j^{th} entry of ϕ_2, gives the "new" coordinate for data point x_j (i.e., $\phi_2(j) = y(x_j)$). In our application, we have assumed that this *single* direction of variability, parameterized by ϕ_2, is one-to-one with time. Ordering the data by $\phi_2(j)$ will then, effectively, order them in time. The procedure generalizes when the data lie on higher-dimensional manifolds (not just curves) in data space, where leading eigenvectors can give subsequent embedding coordinates for the data.

Vector diffusion maps(Singer and Wu, 2012)

In vector diffusion maps, given data points $x_1, \ldots, x_m$, one first constructs the matrix $S \in \mathbb{R}^{md\times md}$, with the i, j^{th} block of S, S_{ij}, defined as

$$S_{ij} = A_{ij}H_{ij} \tag{14}$$

where $A_{ij} \in \mathbb{R}$ (defined in (13)) pertains to the diffusion kernel between data points, and $H_{ij} \in \mathbb{R}^{d\times d}$ (defined in (5)) pertains to the pairwise alignment between data points. It is important to note that distance $d(x_i, x_j)$ used in the diffusion kernel in (11) is the distance between data points *after* after pairwise alignment, i.e., the minimum distance between all possible shifts of the two data points (which is obtained in (17)). In the language of symmetry groups, this distance is a metric between the orbits induced by the relevant symmetry group.

One then computes the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_{md}$ and eigenvectors $\phi_1, \phi_2, \ldots, \phi_{md}$ of S, ordered such that $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_{md}|$. These eigenvectors contain information about *both* the optimal rotations (the "synchronization" component) and the variation of the data *after* the spatial symmetries have been factored out (in our case, their temporal variation). Assuming that the data (after symmetries have been factored out) are relatively closely clustered, it is reasonable to expect, as in angular synchronization, that the top (block) eigenvector of S contains approximations of the optimal rotations, which can be computed in the same way from (8). We then expect subsequent eigenvectors to contain information about the main direction(s) of data variability modulo the geometric symmetries.

In general, the embedding coordinates are given by

$$\psi_{k,l}(i) = \langle \phi_k(i), \phi_l(i) \rangle, \tag{15}$$

where $\phi_k(i) \in \mathbb{R}^d$ denotes the i^{th} block of ϕ_k, If we assume that the rotations and the dynamics are uncoupled and therefore separable, then the eigenvectors of S have the following structure: each block eigenvector contains estimates of the optimal rotations (up to a constant rotation)

multiplied by the corresponding embedding coordinate (a scalar) As the first diffusion maps coordinate is constant over the data, the first block eigenvector contains only the optimal rotations. The second block eigenvector (eigenvectors $d+1$ through $2d$) contains the optimal rotations, each multiplied by their second diffusion maps coordinate. We can therefore recover this diffusion maps coordinate by taking inner products of the columns of the second block eigenvector with columns of the first block eigenvector. The j^{th} embedding coordinate will be given by $\psi_{k,l}$, where $jd+1 < k \leq (j+1)d$ and $1 \leq l \leq d$, and we select k, l such that the coordinate $\psi_{k,l}$ has the largest variability, i.e., the j^{th} coordinate is $\psi_{k,l}$, where k, l is the solution to

$$\max_{\substack{jd+1 \leq k \leq (j+1)d \\ 1 \leq l \leq d}} \sum_i \psi_{k,l}(i)^2. \tag{16}$$

Registering images

To register sets of images, the first step is to compute the optimal alignments between pairs of images. Practically, we have square images discretized as pixels (rather than continuous functions on the plane). For each image pair I_i and I_j we compute

$$\theta_{ij} = \underset{0^\circ \leq \theta < 360^\circ}{\arg\min} \|g(I_j, \theta) - I_i\|^2. \tag{17}$$

where $g(I_j, \theta)$ is image I_j rotated around the center of the square by θ degrees. The norm, $\|\cdot\|$, is the Euclidean norm between the pixel intensities of the channels. The domain of the image (a square) is not invariant to our rotations; however, the pixels near the corners of the square are preprocessed to have zero intensity, and so the norm can be meaningfully computed as long as the main image does not "move out of" the original square. Image rotation is performed with the `imwarp` function in MATLAB, using linear interpolation to estimate the pixel intensities after rotation. The missing pixels in the corners of the rotated image are taken to have zero intensity. The solution to (17) is not easily computed, as the objective function will most likely be nonconvex. Therefore, instead of using an optimization procedure, we discretize the search space and exhaustively search to find the solution (for the results presened, we use 10° discretization steps). Although computationally demanding, this "embarrassingly parallelizable" direct enumeration approach is not prohibitive here. Once we have computed θ_{ij} for all image pairs, we can proceed with the vector diffusion maps procedure. The rotation matrices returned from vector diffusion maps can be used to calculate the angle of rotation for each image (by inverting (4)), and the function g as described previously is then used to rotate the images.

The eigenvalue spectrum

We can use the eigenvalues from (vector) diffusion maps to help deduce the dimensionality of the data. In diffusion maps, the largest eigenvalue will always be 1 and correspond to the trivial (constant) eigenvector, and $|\lambda_k|$ gives a measure of the importance of coordinate ϕ_k. We therefore expect to see a "spectral gap" in the eigenvalues which separates the meaningful coordinates from those corresponding to noise. However, some embedding coordinates which appear meaningful according to the eigenvalues may be harmonics of previous coordinates (Ferguson et al., 2010), and one must visually check for correlations among potential embedding coordinates before deducing the true dimensionality of the data.

In vector diffusion maps, the importance of each coordinate is measured by the product of the corresponding eigenvalues (i.e., the importance of $\psi_{k,l}$ is given by $|\lambda_k \lambda_l|$). We again expect to see a "spectral gap" in these eigenvalue products between those corresponding to meaningful coordinates (modulo higher harmonics) and those corresponding to noise.

Smooth trajectories from registered and ordered images

Once we have registered and ordered the images, we can smooth the resulting trajectory to obtain a "stereotypic" developmental trajectory. Let $I_1, \ldots, I_m$ denote the set of *registered and ordered* images (so I_j is the j^{th} image in the ordered set). We define the average image at time τ, denoted $\overline{I}_\tau$ (where $1 \leq \tau \leq m$), as a (Gaussian) weighted average of the images,

$$\overline{I}_\tau = \frac{\sum_j \exp\left(-\frac{|j-\tau|^2}{\sigma^2}\right) I_i}{\sum_j \exp\left(-\frac{|j-\tau|^2}{\sigma^2}\right)} \tag{18}$$

where σ is the scale of the Gaussian filter. For the images in Fig. 5 and 6, we take $\sigma = 2$. See (Kemelmacher-Shlizerman et al., 2011) for a more detailed discussion.

References

Ahuja, S., Kevrekidis, I. G. and Rowley, C. W. (2007). Template-based stabilization of relative equilibria in systems with continuous symmetry. *Journal of Nonlinear Science* **17**, 109–143.

Bandeira, A. S., Singer, A. and Spielman, D. A. (2013). A Cheeger inequality for the graph connection Laplacian. *SIAM Journal on Matrix Analysis and Applications* **34**, 1611–1630.

Belkin, M. and Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation* **15**, 1373–1396.

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* pp. 679–698.

Coifman, R. R., Lafon, S., Lee, A. B., Maggioni, M., Nadler, B., Warner, F. and Zucker, S. W. (2005). Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proc. Natl. Acad. Sci. U.S.A.* **102**, 7426–7431.

Coifman, R. R., Shkolnisky, Y., Sigworth, F. J. and Singer, A. (2008). Graph Laplacian tomography from unknown random projections. *IEEE Trans. Image Process.* **17**, 1891–1899.

Ferguson, A. L., Panagiotopoulos, A. Z., Debenedetti, P. G. and Kevrekidis, I. G. (2010). Systematic determination of order parameters for chain dynamics using diffusion maps. *Proc. Natl. Acad. Sci. U.S.A.* **107**, 13597–13602.

Goemans, M. X. and Williamson, D. P. (1995). Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)* **42**, 1115–1145.

Kemelmacher-Shlizerman, I., Shechtman, E., Garg, R. and Seitz, S. M. (2011). Exploring photobios. In *ACM Trans. Graph.*, volume 30, p. 61.

Rohrdanz, M. A., Zheng, W., Maggioni, M. and Clementi, C. (2011). Determination of reaction coordinates via locally scaled diffusion map. *J. Chem. Phys.* **134**, 124116.

Shlens, J. (2005). A tutorial on principal component analysis. *Systems Neurobiology Laboratory, University of California at San Diego* .

Singer, A. (2011). Angular synchronization by eigenvectors and semidefinite programming. *Appl. Comput. Harmon. Anal.* **30**, 20–36.

Singer, A. and Wu, H.-T. (2012). Vector diffusion maps and the connection Laplacian. *Commun. Pure Appl. Math.* **65**, 1067–1144.

Outline of algorithm used to register and order images

1. Compute the alignments for each pair of images, as described in (17). Retain both the optimal pairwise rotations as well as the minimum distances obtained when pairs are optimally aligned.

2. Compute the corresponding rotation matrices $R(\theta_{ij})$ from (4).

3. Select ϵ for use in the diffusion maps kernel. Taking ϵ to be $25-50\%$ of the median of the pairwise distances often yields good results.

4. Compute the matrix S in (14). A_{ij} is computed from (11) and (13), using the pairwise distances from step 1, and $H_{ij} = R(\theta_{ij})$ are computed in step 2.

5. Compute the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_{2m}$ and eigenvectors $\phi_1, \phi_2, \ldots, \phi_{2m}$ of S, ordered such that $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_{2m}|$.

6. Estimate the optimal three-dimensional rotation for each image: stack the first two eigenvectors ϕ_1, ϕ_2 into a $2m \times 2$ matrix, and then divide this matrix into m 2×2 blocks, $\hat{R}_1, \ldots, \hat{R}_m$. Compute the estimated rotations for each of these m blocks as in (8). If most of the rotations are improper ($det(R) = -1$), flip the sign of one of the eigenvectors.

7. Multiply each estimated rotation $R_{i,est}$ by $R_{1,est}^T$ (so that $R_{1,est}$ will become the identity matrix).

8. For each image i, compute the optimal rotation to align the image by converting $R_{i,est}^T$ (note the transpose) to the corresponding angle using (4).

9. Compute the first embedding coordinate $\psi_{k,1}(i)$ as described in (15) and (16), where $3 \leq k \leq 4$.

10. To order the images, sort them by the values of this embedding coordinate $\psi_{k,l}(i)$.

Data set	VDM Rank Correlation	PCA Rank Correlation
Drosophila gastrulation (live)	0.9989	0.8137
Zebrafish epiboly	0.9955	0.6351
Drosophila gastrulation (fixed)	0.9716	0.8658
Drosophila wing discs	0.9436	0.9381

Table 1: Comparison between rank correlation coefficients when ordering using the first vector diffusion maps (VDM) embedding coordinate, and rank correlation coefficients when ordering by the first principal component analysis (PCA) (Shlens, 2005) projection coefficient. For ordering using PCA, we computed the first principal component of the registered images, and then ordered the images by the projection coefficients onto this first mode. The PCA ordering is always less accurate than the vector diffusion maps ordering. In the zebrafish data set, the ordering is much less accurate using PCA, as the dynamics of the morphing and spreading of the cell mass are highly nonlinear. The PCA and VDM orderings are comparable for the wing disc data; this is to be expected, as the tissue simply grows in time.

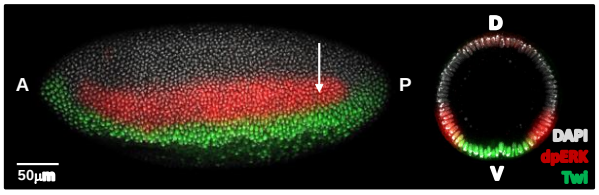

Fig. S1: (Left) A lateral view of a *Drosophila* embryo stained with DAPI (gray), dpERK (red), and Twi (green). The embryo is presented so that the anterior (A) side is to the left and the posterior (P) side is to the right. The arrow indicates the position where the cross-section of an embryo is imaged. (Right) A dorsoventral view of the cross-section of the *Drosophila* embryo. The dorsal (D) side is up and the ventral (V) side is down. Images were collected at the focal plane $\sim 18\%$ from the posterior pole of an embryo (arrow in the left image).

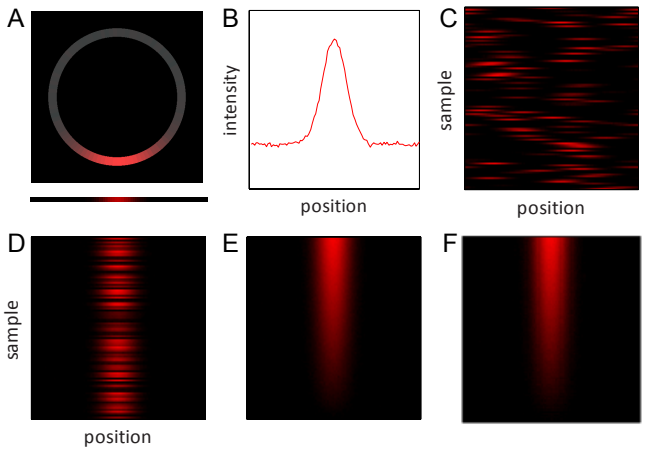

Fig. S2: Synthetic data set used to illustrate the data processing algorithms. *(A)* One-dimensional concentration profile on a ring (top), and the corresponding profile on a line (bottom). *(B)* Intensity corresponding to the profile in *A*. *(C)* An ensemble of concentration profiles, each of the form described in *A*. Each row in the array corresponds to a single profile. *(D)* The profiles in *C*, now registered using angular synchronization. *(E)* The profiles in *D*, now temporally ordered using diffusion maps. *(F)* The profiles in *C*, registered and temporally ordered in a single step using vector diffusion maps.

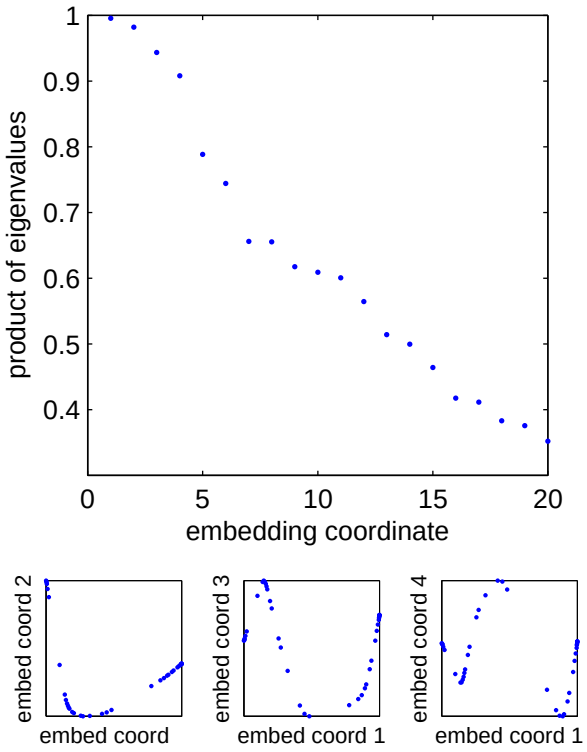

Fig. S3: Eigenvalue spectra for the *Drosophila* live imaging data set presented in Fig. 3. Note that there is a gap after the fourth eigenvalue product. Below are the second, third, and fourth embedding coordinate plotted versus the first embedding coordinate. Note that coordinates 2–4 are higher harmonics (and thus simple functions) of coordinate 1, and are therefore not informative about structure in the data set. We can conclude that the data set is effectively one-dimensional and can be parameterized/ordered by coordinate 1.

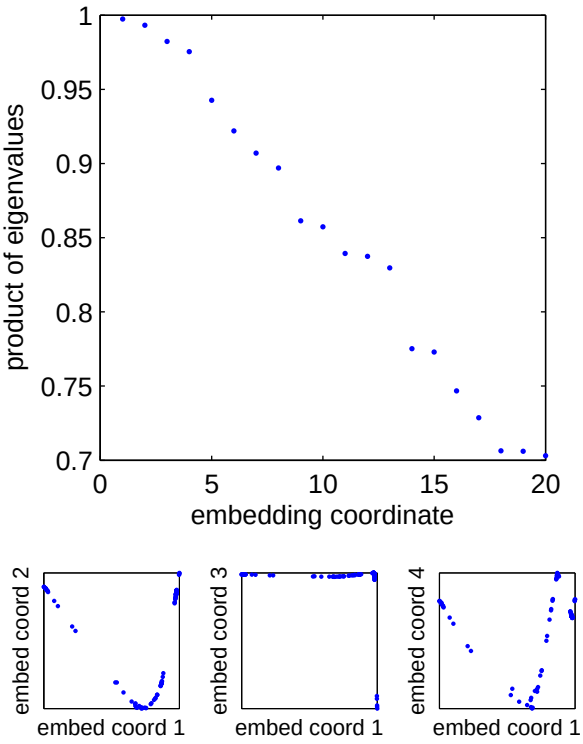

Fig. S4: Eigenvalue spectra for the zebrafish data set presented in Fig. 4. Note that there is a gap after the fourth eigenvalue product. Below are the second, third, and fourth embedding coordinate plotted versus the first embedding coordinate. Note that coordinates 2–4 are higher harmonics (and thus simple functions) of coordinate 1, and are therefore not informative about structure in the data set. It is not immediately obvious that coordinate 3 is a harmonic of coordinate 1; the distortion in the plot is due to density effects in the data (the developmental changes are slower towards the beginning of the trajectory, and so there is a higher density of images in this portion of the one-dimensional curve), We can conclude that the data set is effectively one-dimensional and can be parameterized/ordered by coordinate 1.

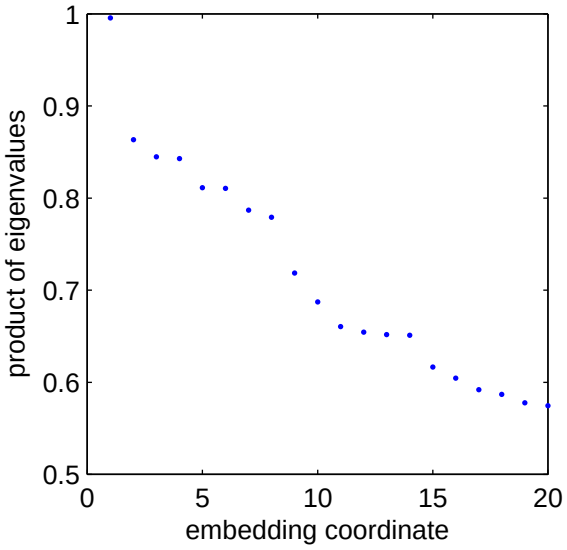

Fig. S5: Eigenvalue spectra for the fixed *Drosophila* images presented in Fig. 5. Note that there is a gap after the first eigenvalue product. We can conclude that the data set is effectively one-dimensional and can be parameterized/ordered by coordinate 1.

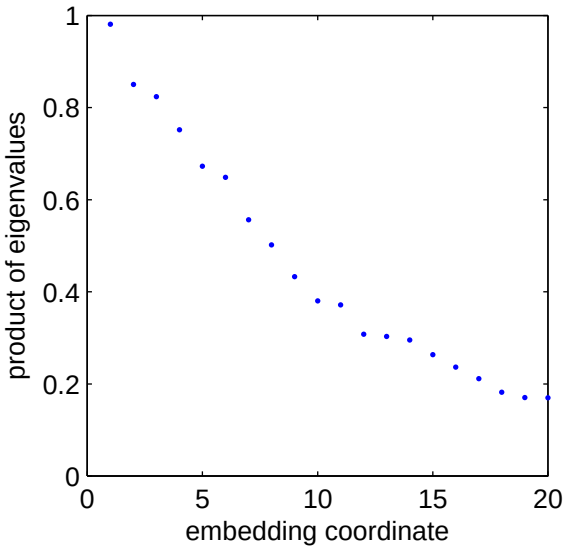

Fig. S6: Eigenvalue spectra for the wing disc data set presented in Fig. 6. Note that there is a gap after the first eigenvalue product. We can conclude that the data set is effectively one-dimensional and can be parameterized/ordered by coordinate 1.