PERSISTENCE IMAGES AND CROCKER PLOTS AS TOPOLOGICAL FEATURE VECTORS

August 10, 2018

Lori Ziegelmeier

TRIPODS Summer Bootcamp: Topology and Machine Learning
ICERM

INTRODUCTION

MOTIVATION:

Goal:

Develop representations of persistent homology that ‘vectorize’ topological information in a way that is accessible to machine learning techniques.

PERSISTENT HOMOLOGY AND MACHINE LEARNING

TOPOLOGICAL DATA ANALYSIS

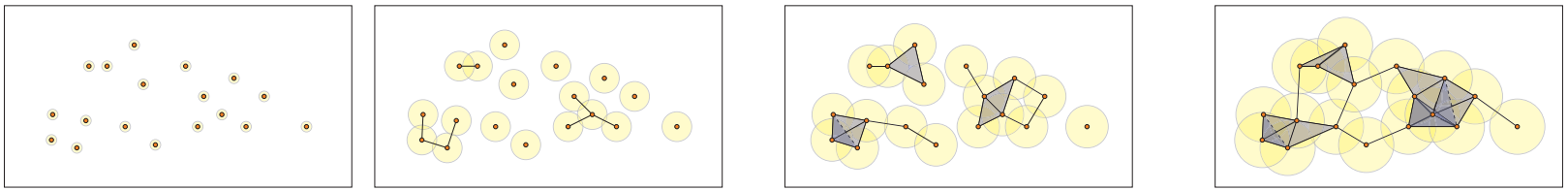
1. Envision data as a point cloud

2. Create connections between proximate points
    - build simplicial complex

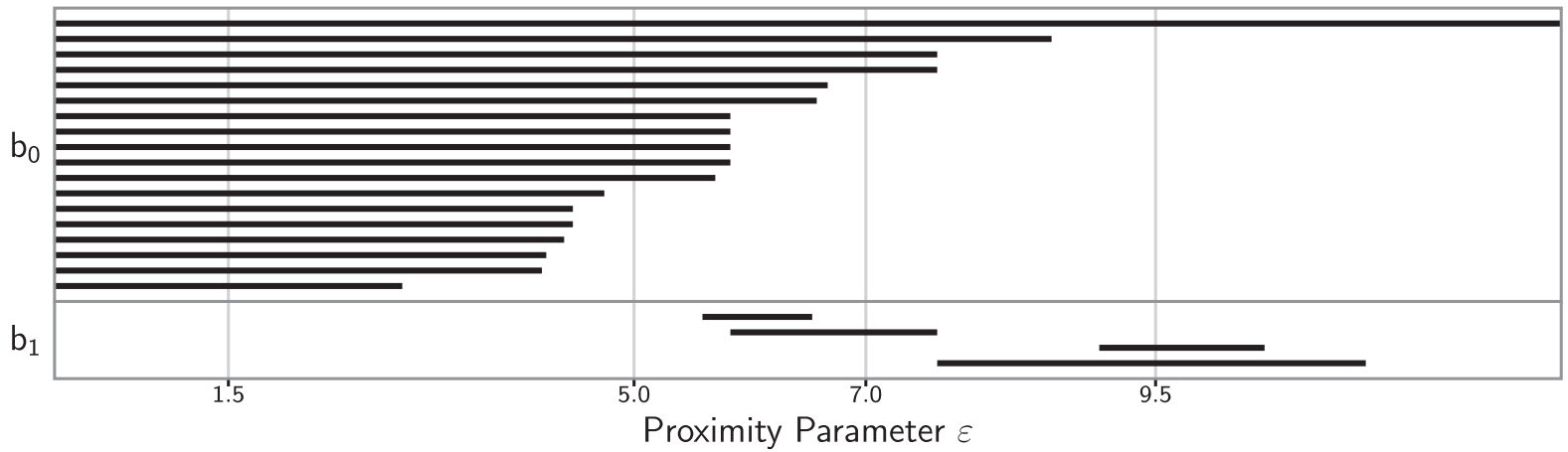
3. Determine topological structure of complex
    - compute homology

4. Vary proximity parameter to assess different scales
    - calculate persistent homology

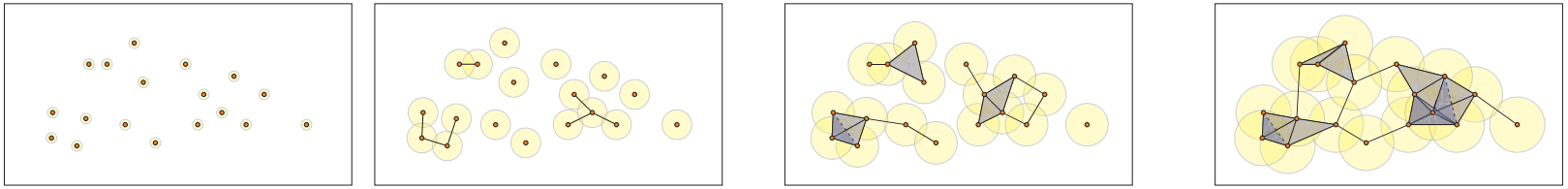
COMPUTE PERSISTENT HOMOLOGY



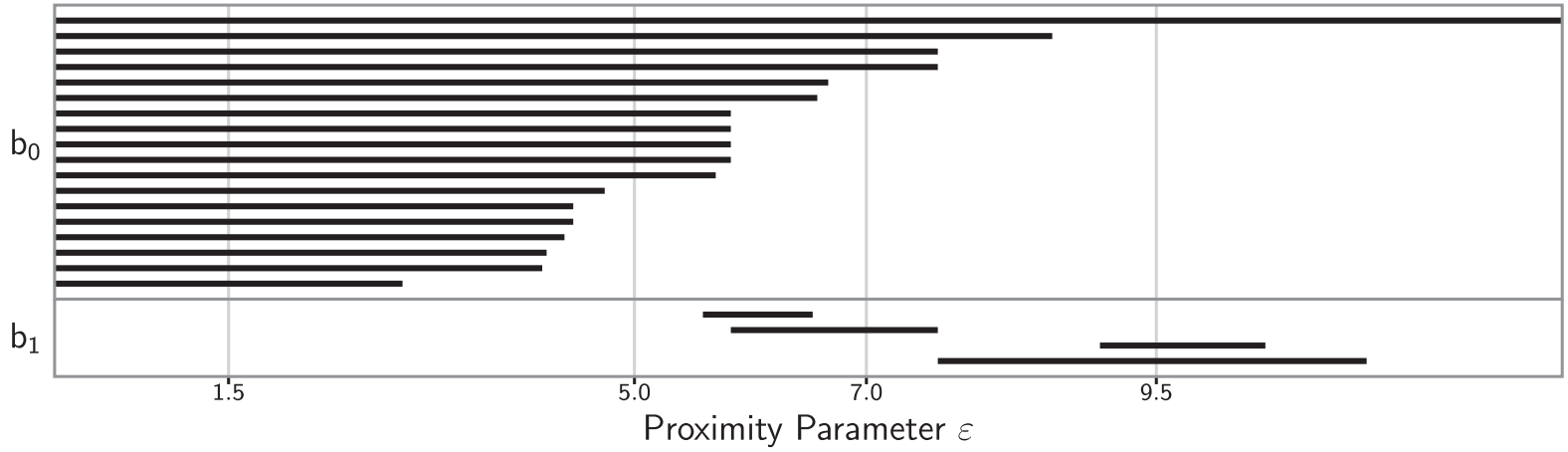
$\varepsilon = 1.5$ $\varepsilon = 5.0$ $\varepsilon = 7.0$ $\varepsilon = 9.5$

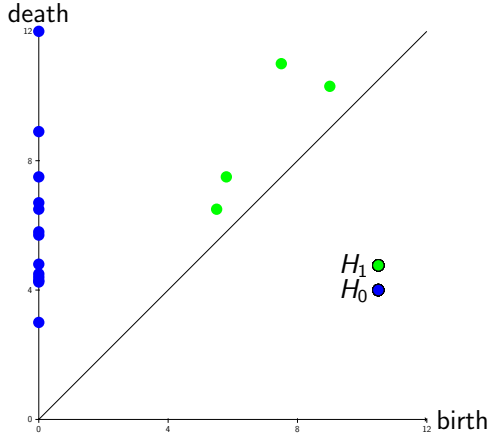


COMPUTE PERSISTENT HOMOLOGY



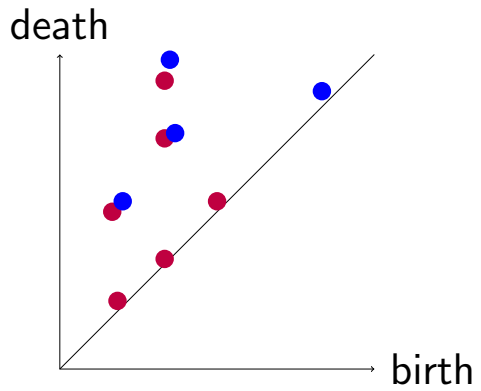
$\varepsilon = 1.5$ $\varepsilon = 5.0$ $\varepsilon = 7.0$ $\varepsilon = 9.5$





PERSISTENCE DIAGRAMS AS A METRIC SPACE

The space of Persistence Diagrams (PDs) can be endowed with a metric.



Definition

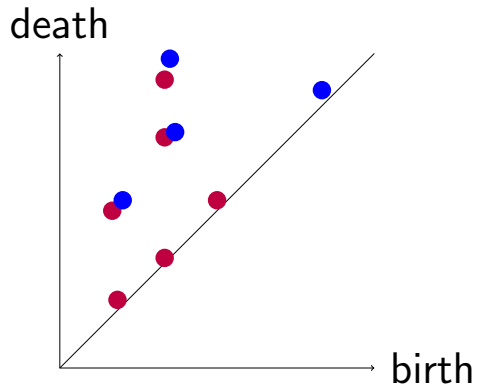
The p-Wasserstein distance between two PDs B and B' is given by

$$W_p(B, B') = \inf_{\gamma : B \to B'} \left(\sum_{u \in B} ||u - \gamma(u)||_\infty^p \right)^{1/p},$$

where $1 \leq p < \infty$ and γ ranges over bijections between B and B'.

PERSISTENCE DIAGRAMS AS A METRIC SPACE

The space of Persistence Diagrams (PDs) can be endowed with a metric.



Definition

The bottleneck distance between two PDs B and B' is given by

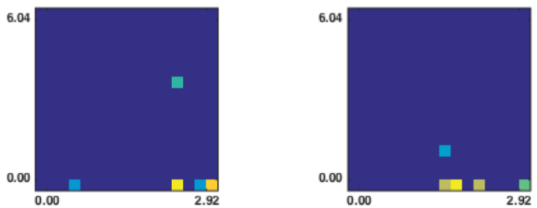
$$W_\infty(B, B') = \inf_{\gamma: B \to B'} \sup_{u \in B} ||u - \gamma(u)||_\infty,$$

where ranges over bijections between B and B'.

OTHER APPROACHES

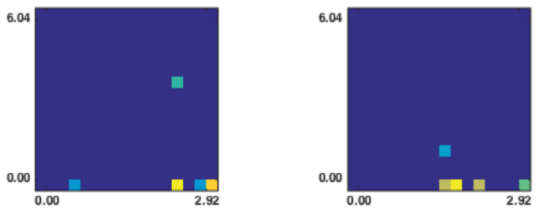
OTHER APPROACHES

- Rouse et al. (2015) create a vector representation by superimposing a grid over a PD and counting number of points in each bin.

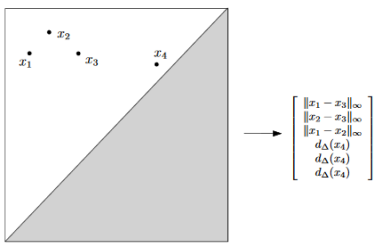


OTHER APPROACHES

- Rouse et al. (2015) create a vector representation by superimposing a grid over a PD and counting number of points in each bin.

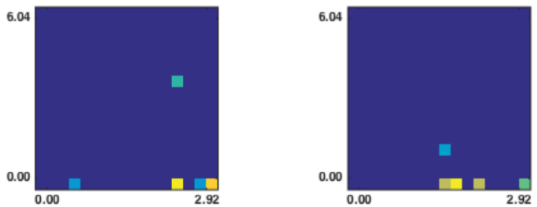


- Carriere et al. (2015) develop a stable vector representation by rearranging the entries of the distance matrix between points in a PD.

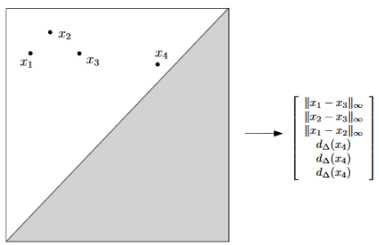


OTHER APPROACHES

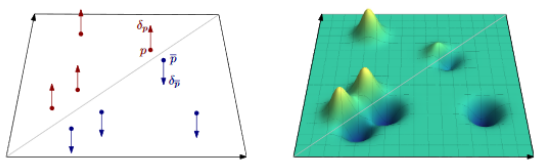
- Rouse et al. (2015) create a vector representation by superimposing a grid over a PD and counting number of points in each bin.



- Carriere et al. (2015) develop a stable vector representation by rearranging the entries of the distance matrix between points in a PD.



- Reininghaus et al. (2015) produce a stable surface from a PD by taking sum of a positive Gaussian centered on each PD point together with a negative Gaussian centered on its reflection below the diagonal.

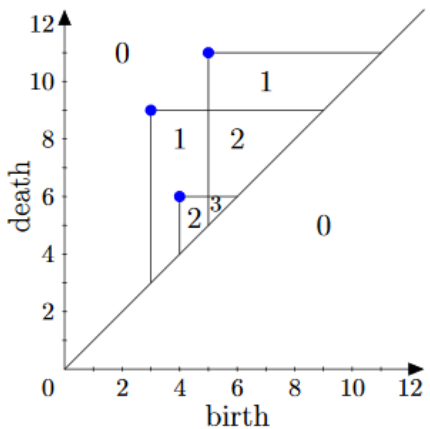


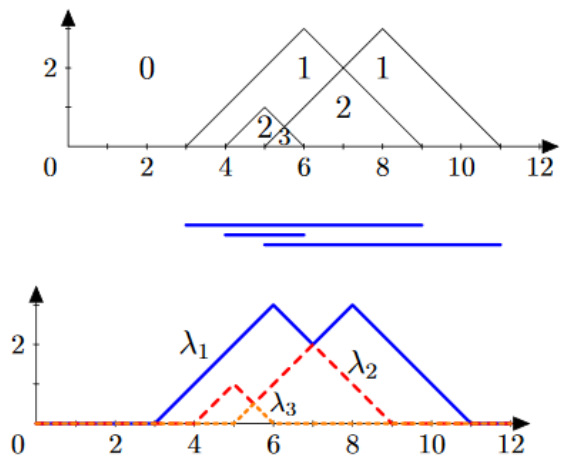
Images from corresponding manuscripts

PERSISTENCE LANDSCAPES

- Bubenik (2015) develops persistence landscape (PL), a stable functional representation of a PD that lies in a Banach space.
- A PL is a function $\lambda : \mathbb{N} \times \mathbb{R} \to [-\infty, \infty]$ (which can equivalently be thought of as a sequence of functions $\lambda_k : \mathbb{R} \to [-\infty, \infty]$)

$$\lambda_k(x) = k\text{-th largest value of } \min(x - b_i, d_i - x).$$





Images from Bubenik (2015)

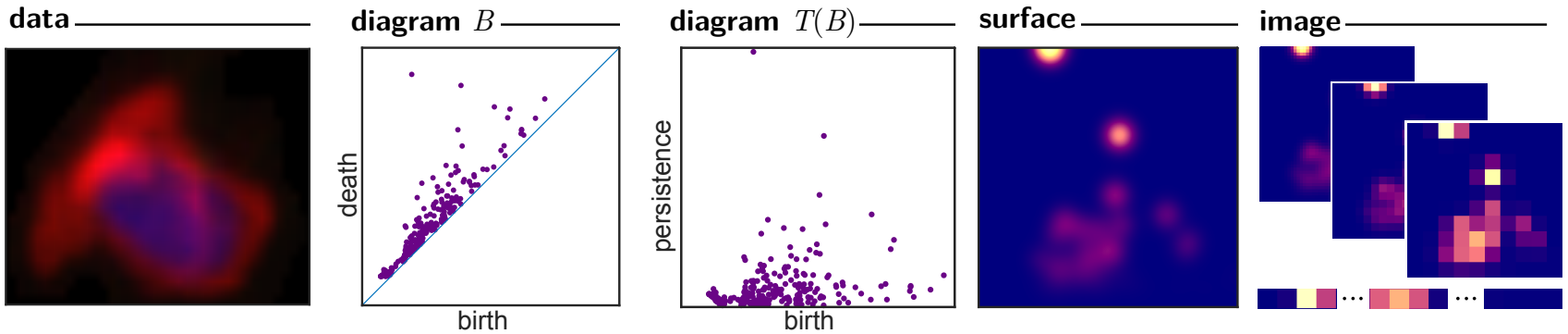
PROBLEM STATEMENT

How can we represent a persistence diagram so that:

1. the output of the representation is a vector in $\mathbb{R}^n$,
2. the representation is stable with respect to input noise,
3. the representation is efficient to compute,
4. the representation maintains an interpretable connection to the original PD, and
5. the representation allows one to adjust the relative importance of points in different regions of the PD?

PERSISTENCE IMAGES

PERSISTENCE IMAGE PIPELINE



PERSISTENCE IMAGE (PI)

1. Transform the PD from birth-death coordinates to birth-persistence coordinates
   - via $T : \mathbb{R}^2 \longrightarrow \mathbb{R}^2$ the linear transformation $T(x, y) = (x, y - x)$

PERSISTENCE IMAGE (PI)

1. Transform the PD from birth-death coordinates to birth-persistence coordinates
   - via $T : \mathbb{R}^2 \to \mathbb{R}^2$ the linear transformation $T(x, y) = (x, y - x)$
2. For each point $u = (u_x, u_y)$ in PD $T(B)$, center a probability distribution ϕ_u
   - In our applications, $\phi_u = g_u$ a normalized Gaussian distribution:

$$g_u(x, y) = \frac{1}{2\pi\sigma^2} e^{-[(x-u_x)^2+(y-u_y)^2]/2\sigma^2}$$

PERSISTENCE IMAGE (PI)

1. Transform the PD from birth-death coordinates to birth-persistence coordinates
   - via $T : \mathbb{R}^2 \to \mathbb{R}^2$ the linear transformation $T(x, y) = (x, y - x)$
2. For each point $u = (u_x, u_y)$ in PD $T(B)$, center a probability distribution ϕ_u
   - In our applications, $\phi_u = g_u$ a normalized Gaussian distribution:

     $$g_u(x, y) = \frac{1}{2\pi\sigma^2} e^{-[(x-u_x)^2+(y-u_y)^2]/2\sigma^2}$$
3. Produce a persistence surface as a weighted sum of these distributions
   - $\rho_B(x, y) = \sum_{u \in T(B)} f(u) g_u(x, y)$

PERSISTENCE IMAGE (PI)

1. Transform the PD from birth-death coordinates to birth-persistence coordinates
   - via $T : \mathbb{R}^2 \to \mathbb{R}^2$ the linear transformation $T(x, y) = (x, y - x)$
2. For each point $u = (u_x, u_y)$ in PD $T(B)$, center a probability distribution ϕ_u
   - In our applications, $\phi_u = g_u$ a normalized Gaussian distribution:

     $$g_u(x, y) = \frac{1}{2\pi\sigma^2} e^{-[(x-u_x)^2+(y-u_y)^2]/2\sigma^2}$$
3. Produce a persistence surface as a weighted sum of these distributions
   - $\rho_B(x, y) = \sum_{u \in T(B)} f(u) g_u(x, y)$
4. Overlay a grid onto the PD

PERSISTENCE IMAGE (PI)

1. Transform the PD from birth-death coordinates to birth-persistence coordinates
   - via $T : \mathbb{R}^2 \to \mathbb{R}^2$ the linear transformation $T(x, y) = (x, y - x)$
2. For each point $u = (u_x, u_y)$ in PD $T(B)$, center a probability distribution ϕ_u
   - In our applications, $\phi_u = g_u$ a normalized Gaussian distribution:

     $$g_u(x, y) = \frac{1}{2\pi\sigma^2} e^{-[(x-u_x)^2+(y-u_y)^2]/2\sigma^2}$$
3. Produce a persistence surface as a weighted sum of these distributions
   - $\rho_B(x, y) = \sum_{u \in T(B)} f(u) g_u(x, y)$
4. Overlay a grid onto the PD
5. The image value at pixel p, a square in the grid, is the integral of the persistence surface over that pixel
   - $I(\rho_B)_p = \iint_p \rho_B(x, y)\, dydx$

THE WEIGHTING FUNCTION

Define a weighting function $f : \mathbb{R}^2 \to \mathbb{R}$ that is

- zero along the horizontal axis,
- continuous, and
- piecewise differentiable.

THE WEIGHTING FUNCTION

Define a weighting function $f : \mathbb{R}^2 \to \mathbb{R}$ that is

- zero along the horizontal axis,
- continuous, and
- piecewise differentiable.

Convenient choice depends only on the vertical persistence coordinate y.

THE WEIGHTING FUNCTION

Define a weighting function $f : \mathbb{R}^2 \to \mathbb{R}$ that is

- zero along the horizontal axis,
- continuous, and
- piecewise differentiable.

Convenient choice depends only on the vertical persistence coordinate y.
For example:

$$f(x, y) = w_b(y) = \begin{cases} 0 & y \leq 0 \\ \frac{y}{b} & 0 < y < b \\ 1 & y \geq b \end{cases}$$

where b is the persistence value of the most persistent feature in all trials.

EXAMPLES OF PERSISTENCE IMAGES (CIRCLE)

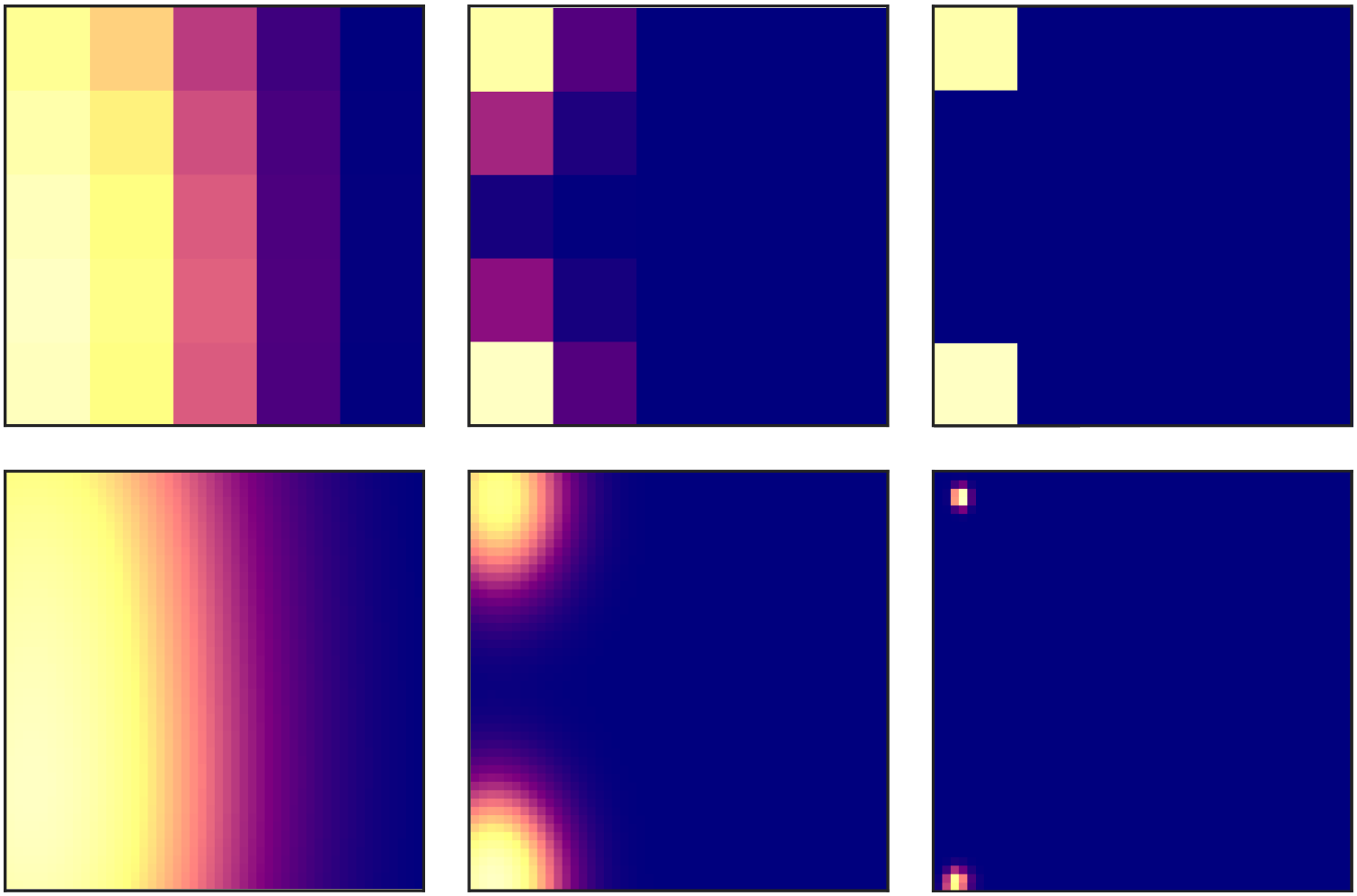


Figure: H_1 PIs of a noisy circle. The first row has resolution 5×5 while the second has 50×50. The columns have variance $\sigma = 0.2$, $\sigma = 0.01$, and $\sigma = 0.0001$, respectively.

EXAMPLES OF PERSISTENCE IMAGES (TORUS)

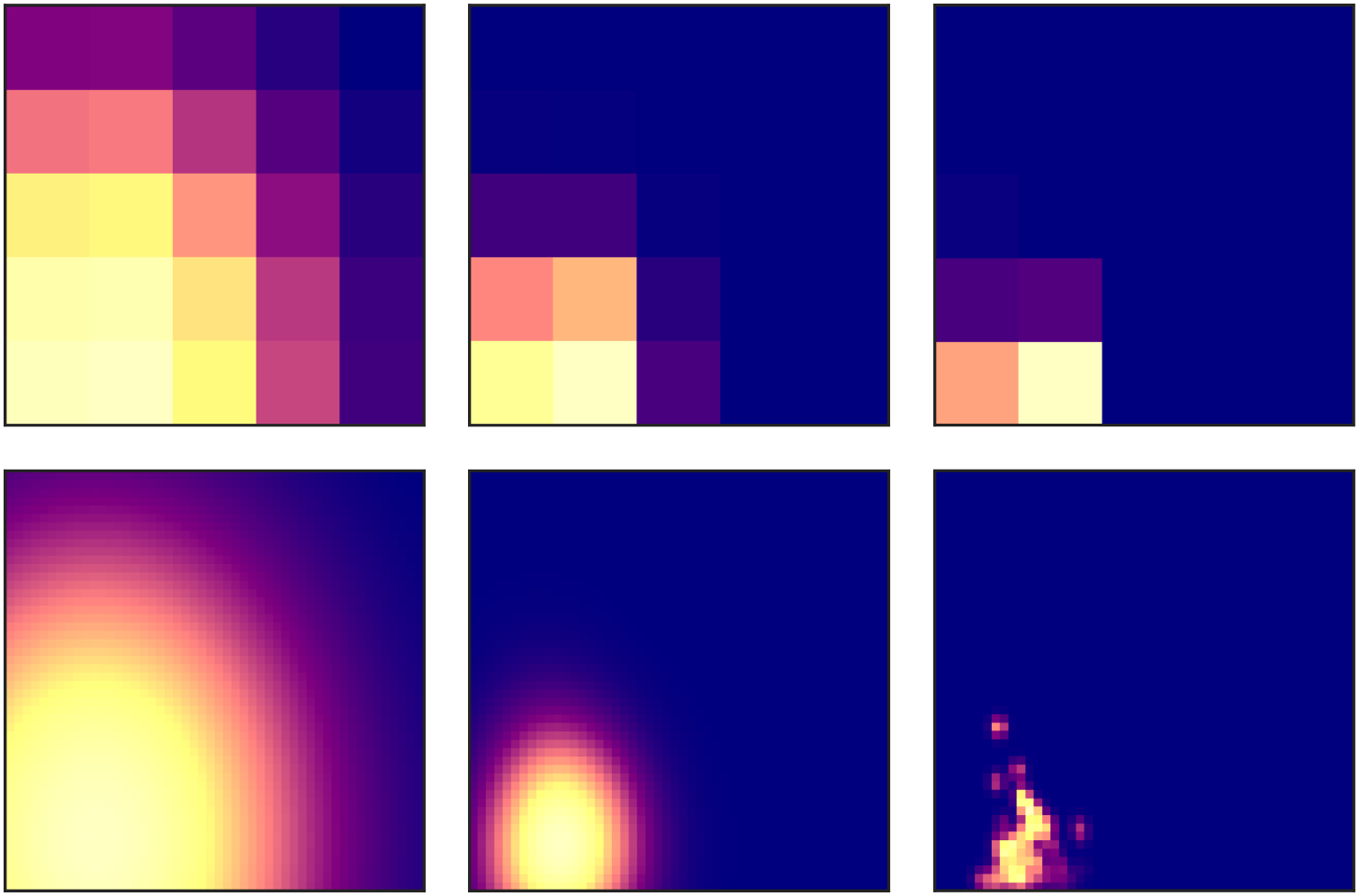


Figure: H_1 PIs of a noisy torus. The first row has resolution 5×5 while the second has 50×50. The columns have variance $\sigma = 0.2$, $\sigma = 0.01$, and $\sigma = 0.0001$, respectively.

STABILITY OF PERSISTENCE SURFACES

Persistence diagrams are stable (Lipschitz) with respect to the bottleneck metric (Cohen-Steiner, Edelsbrunner, Harer 2007)

$$W_\infty(B(f), B(g)) \leq ||f - g||_\infty$$

Want: $\|\rho_B - \rho_{B'}\|_p \leq C \cdot W_{p'}(B, B')$

STABILITY OF PERSISTENCE SURFACES

Persistence diagrams are stable (Lipschitz) with respect to the bottleneck metric (Cohen-Steiner, Edelsbrunner, Harer 2007)

$$W_\infty(B(f), B(g)) \leq ||f - g||_\infty$$

Want: $\|\rho_B - \rho_{B'}\|_p \leq C \cdot W_{p'}(B, B')$

Theorem

The persistence surface ρ is stable with respect to the 1-Wasserstein distance between two diagrams B, B'. That is,

$$\|\rho_B - \rho_{B'}\|_\infty \leq \sqrt{10}\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big)W_1(B, B').$$

STABILITY OF PERSISTENCE IMAGES

Theorem

The persistence image $I(\rho_B)$ is stable with respect to the 1-Wasserstein distance between diagrams. More precisely, if A is the maximum area of any pixel in the image, A' is the total area of the image, and n is the number of pixels in the image, then

$$\|I(\rho_B) - I(\rho_{B'})\|_\infty \leq \sqrt{10}A\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big)W_1(B,B')$$

$$\|I(\rho_B) - I(\rho_{B'})\|_1 \leq \sqrt{10}A'\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big)W_1(B,B')$$

$$\|I(\rho_B) - I(\rho_{B'})\|_2 \leq \sqrt{10n}A\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big)W_1(B,B').$$

STABILITY OF PERSISTENCE SURFACES USING GAUSSIANS

Theorem

The persistence surface ρ with Gaussian distributions is stable with respect to the 1-Wasserstein distance between two diagrams B, B. That is,

$$\|\rho_B - \rho_{B'}\|_1 \leq \left(\sqrt{5}|\nabla f| + \sqrt{\frac{10}{\pi}}\frac{\|f\|_\infty}{\sigma}\right) W_1(B, B').$$

STABILITY OF PERSISTENCE IMAGES USING GAUSSIANS

Theorem

The persistence image $I(\rho_B)$ with Gaussian distributions is stable with respect to the 1-Wasserstein distance between diagrams. More precisely,

$$\|I(\rho_B) - I(\rho_{B'})\|_1 \leq \left(\sqrt{5}|\nabla f| + \sqrt{\frac{10}{\pi}}\frac{\|f\|_\infty}{\sigma}\right) W_1(B, B')$$

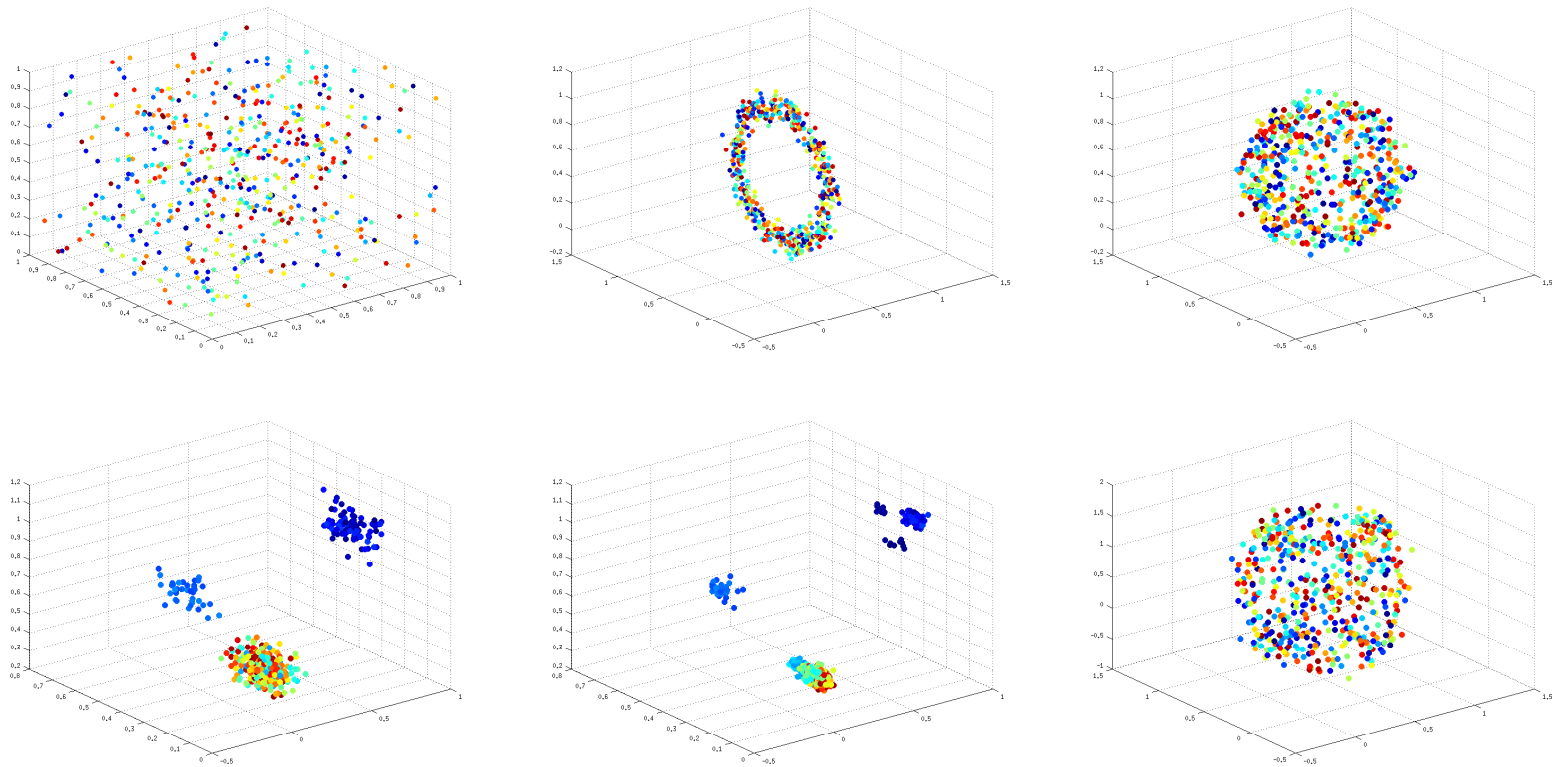
$$\|I(\rho_B) - I(\rho_{B'})\|_2 \leq \left(\sqrt{5}|\nabla f| + \sqrt{\frac{10}{\pi}}\frac{\|f\|_\infty}{\sigma}\right) W_1(B, B')$$

$$\|I(\rho_B) - I(\rho_{B'})\|_\infty \leq \left(\sqrt{5}|\nabla f| + \sqrt{\frac{10}{\pi}}\frac{\|f\|_\infty}{\sigma}\right) W_1(B, B').$$

DATA ANALYSIS

SHAPE DATA

Points sampled from six topological spaces: the solid cube, a circle, a sphere, three clusters, three clusters within three clusters, and a torus



25 point clouds from each space, consisting of 500 points, 2 levels of noise $\eta = 0.05, 0.1$

TRANSFORMING PDS INTO DISTANCE MATRICES

Goal:

Compare classification accuracy and computational time of shape data for

- the PD framework equipped with the Bottleneck, 1-Wasserstein, 2-Wasserstein metrics,
- the PL framework equipped with the L^1, L^2, L^∞ metrics, and
- the PI framework equipped with the L^1, L^2, L^∞ metrics.

Construct $3^2 \cdot 2^2 = 36$ distance matrices of size 150×150.

K-MEDOIDS FOR CLASSIFICATION

K-MEDOIDS FOR CLASSIFICATION

- Depends only on distances from one point to another.

K-MEDOIDS FOR CLASSIFICATION

- Depends only on distances from one point to another.
- Partitions a metric space into *K* clusters by choosing most centrally located point in a cluster, the medoid, and assigning each point to closest medoid.

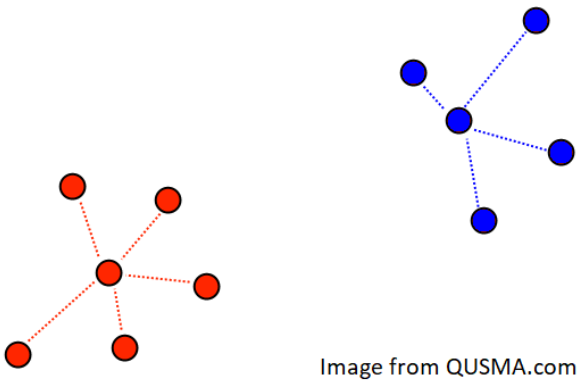


Image from QUSMA.com

K-MEDOIDS FOR CLASSIFICATION

- Depends only on distances from one point to another.
- Partitions a metric space into K clusters by choosing most centrally located point in a cluster, the medoid, and assigning each point to closest medoid.

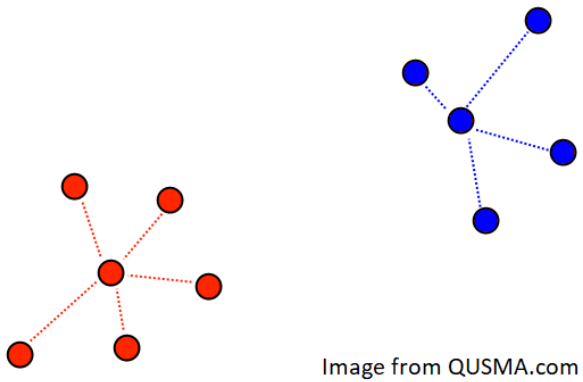


Image from QUSMA.com

- Cluster score is sum of distances from each point to closest medoid.
  - Ideally, want minimal clustering score.
  - In practice, choose large number of random initializations (1,000 with $K = 6$ for us). Return clustering with lowest score.

K-MEDOIDS FOR CLASSIFICATION

- Depends only on distances from one point to another.
- Partitions a metric space into K clusters by choosing most centrally located point in a cluster, the medoid, and assigning each point to closest medoid.

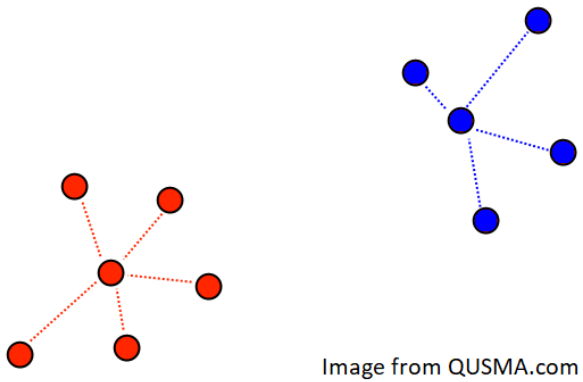


Image from QUSMA.com

- Cluster score is sum of distances from each point to closest medoid.
  - Ideally, want minimal clustering score.
  - In practice, choose large number of random initializations (1,000 with $K = 6$ for us). Return clustering with lowest score.
- Compute accuracy (percentage of shape point clouds identified with a medoid of the same shape class).

K-MEDOIDS COMPARISON OF TOPOLOGICAL REPRESENTATION

Distance Matrix	Accuracy (Noise 0.05)	Time (Noise 0.05)	Accuracy (Noise 0.1)	Time (Noise 0.1)
PD, H_0, L^1	96.0%	37346s	96.0%	42613s
PD, H_0, L^2	91.3%	24656s	91.3%	25138s
PD, H_0, L^∞	60.7%	1133s	63.3%	1149s
PD, H_1, L^1	100%	657s	96.0%	703s
PD, H_1, L^2	100%	984s	97.3%	1042s
PD, H_1, L^∞	81.3%	527s	66.7%	564s
PL, H_0, L^1	92.7%	29s	96.7%	33s
PL, H_0, L^2	77.3%	29s	82.0%	34s
PL, H_0, L^∞	60.7%	2s	63.3%	2s
PL, H_1, L^1	83.3%	36s	80.7%	48s
PL, H_1, L^2	83.3%	50s	66.7%	69s
PL, H_1, L^∞	74.7%	8s	66.7%	9s
PI, H_0, L^1	93.3%	9s	95.3%	9s
PI, H_0, L^2	92.7%	9s	95.3%	9s
PI, H_0, L^∞	94.0%	9s	96.0%	9s
PI, H_1, L^1	100%	17s	95.3%	18s
PI, H_1, L^2	100%	17s	96.0%	18s
PI, H_1, L^∞	100%	17s	96.0%	18s

EXTRACTING FEATURES WITH SPARSE SVM

- 1-norm regularized linear support vector machine (aka sparse SVM) classifies data by generating a separating hyperplane that depends on very few input space features.
- Can reduce data dimension and/or select discriminatory features.
- Linear SSVM feature selection implemented on vectors (such as PIs to select discriminatory pixels during classification).

EXTRACTING FEATURES WITH SPARSE SVM

- 1-norm regularized linear support vector machine (aka sparse SVM) classifies data by generating a separating hyperplane that depends on very few input space features.
- Can reduce data dimension and/or select discriminatory features.
- Linear SSVM feature selection implemented on vectors (such as PIs to select discriminatory pixels during classification).

Goal:

Use Sparse SVM to simultaneously classify data and select input space features that contribute to the classification process.

- Adopt the one-against-all SSVM to separate one class from members of all other classes of PIs of the shape classes using resolution 20×20, variance 0.0001, and noise level 0.05.

SSVM-BASED FEATURE (PIXEL) SELECTION

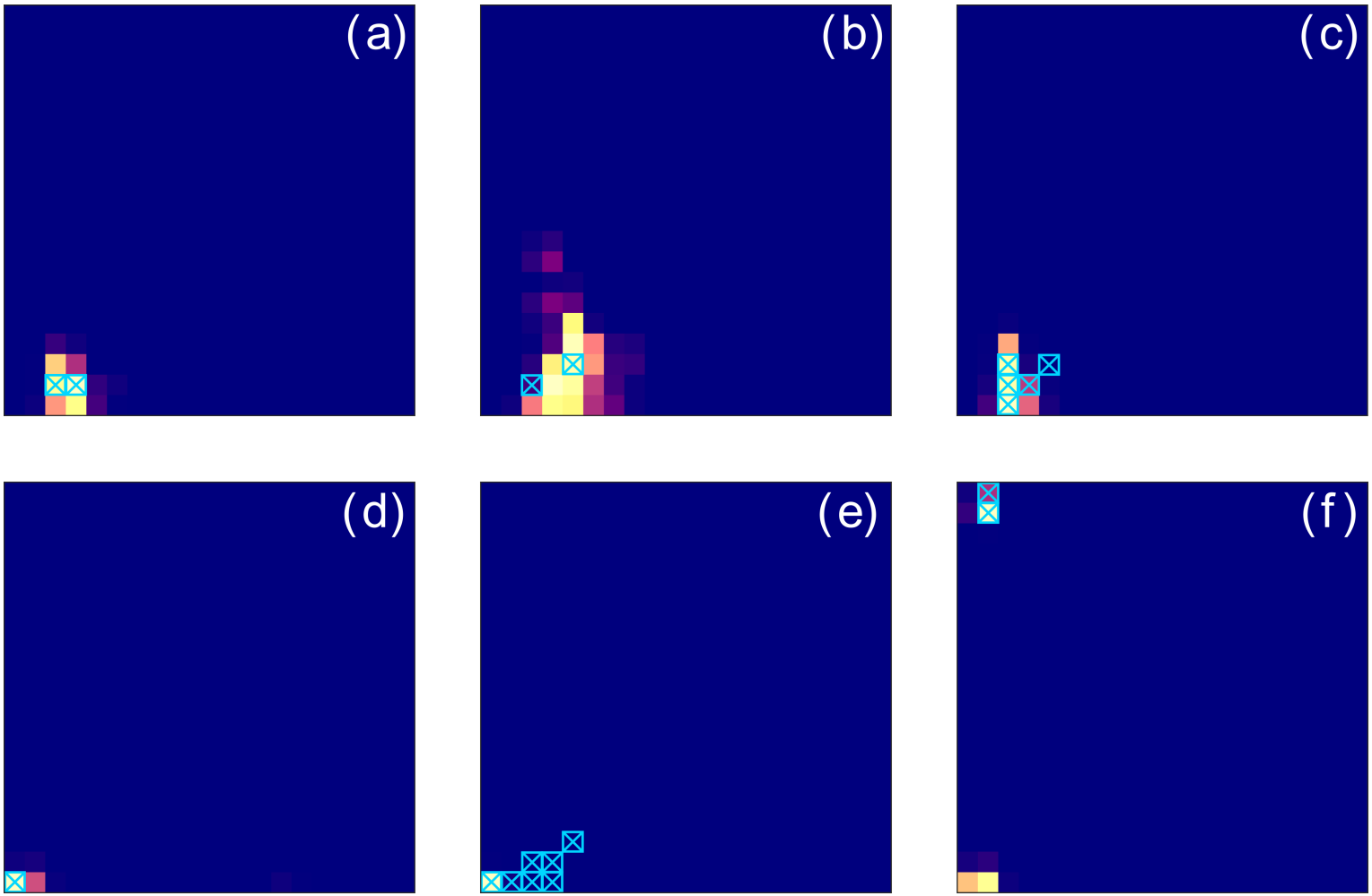


Figure: H_1 PIs of shape data with selected pixels marked by blue crosses. (a) Solid cube (b) Torus (c) Sphere (d) Three clusters (e) Three clusters within three clusters (f) Circle

ANISOTROPIC KURAMOTO-SIVASHINSKY (AKS) EQUATION

ANISOTROPIC KURAMOTO-SIVASHINSKY (AKS) EQUATION

- Partial differential equation for a function $u(x, y, t)$ of spatial variables x, y, and time t.
- Kuramoto-Sivashinsky equation derived in problems involving pattern formation such as surface nanopatterning by ion-beam erosion, epitaxial growth, and solidification from a melt.
- Anisotropic Kuramoto-Sivashinsky (aKS) Equation is given by

$$\frac{\partial}{\partial t}u = -\nabla^2 u - \nabla^2\nabla^2 u + r\left(\frac{\partial}{\partial x}u\right)^2 + \left(\frac{\partial}{\partial y}u\right)^2,$$

  where $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$, and the real parameter r controls the degree of anisotropy.
- For a fixed time t^*, $u(x, y, t^*)$ is a patterned surface (periodic in both x and y) defined over the (x, y)-plane.

PARAMETER OF THE AKS EQUATION

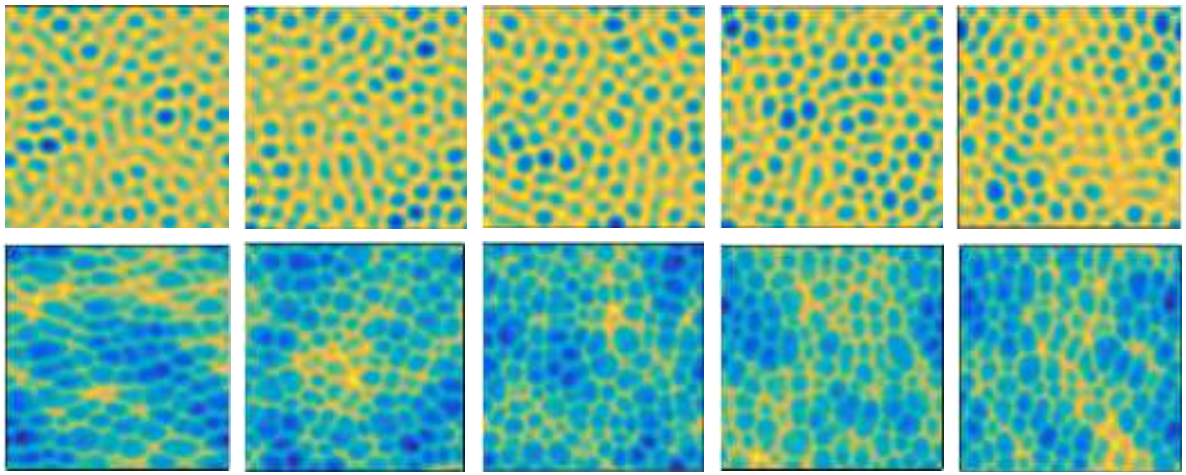


Figure: Plots of $u(x, y, \cdot)$ from simulations of the aKS equation. Columns represent parameters $r = 1$, 1.25, 1.5, 1.75 and 2. Rows represent time: $t = 3$ (top) and $t = 5$ (bottom).

PARAMETER OF THE AKS EQUATION

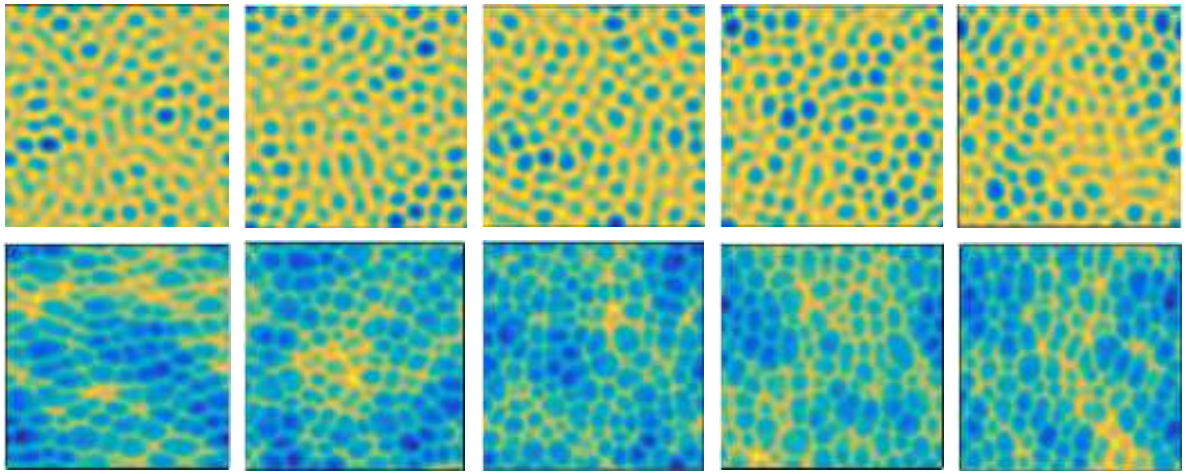


Figure: Plots of $u(x, y, \cdot)$ from simulations of the aKS equation. Columns represent parameters $r = 1$, 1.25, 1.5, 1.75 and 2. Rows represent time: $t = 3$ (top) and $t = 5$ (bottom).

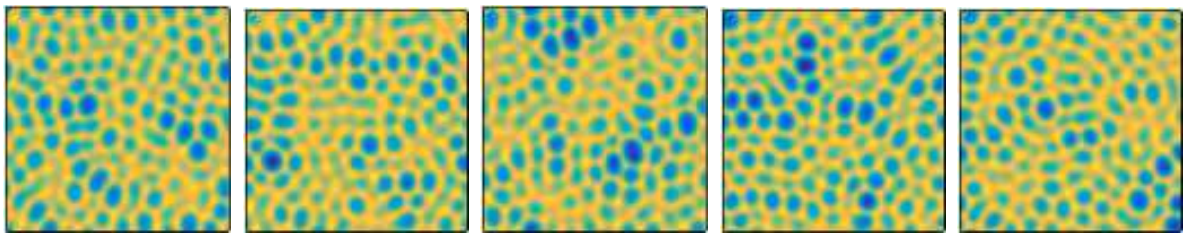


Figure: Surfaces $u(x, y, 3)$ for $r = 1.75$ or $r = 2$. Can you group the images by eye?

PARAMETER OF THE AKS EQUATION

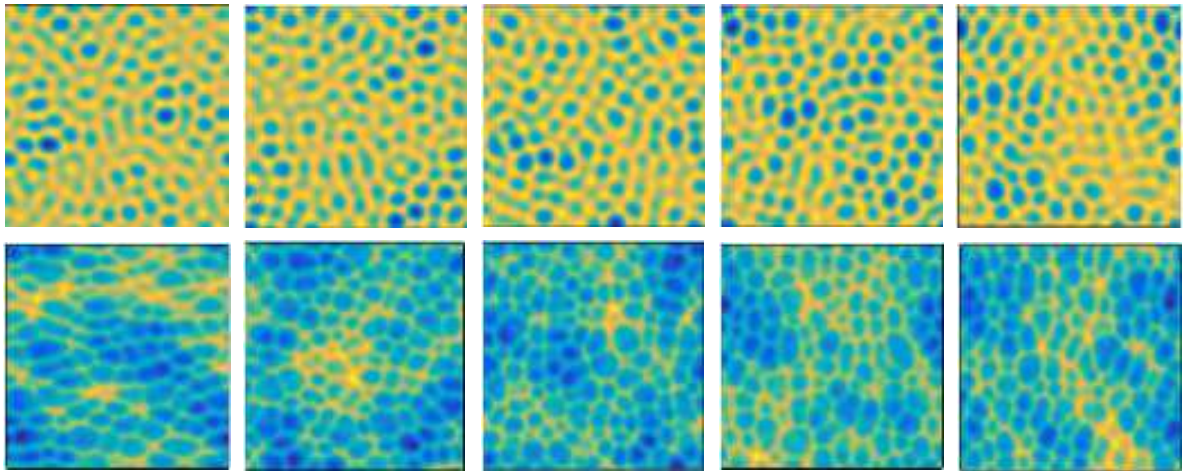


Figure: Plots of $u(x, y, \cdot)$ from simulations of the aKS equation. Columns represent parameters $r = 1$, 1.25, 1.5, 1.75 and 2. Rows represent time: $t = 3$ (top) and $t = 5$ (bottom).

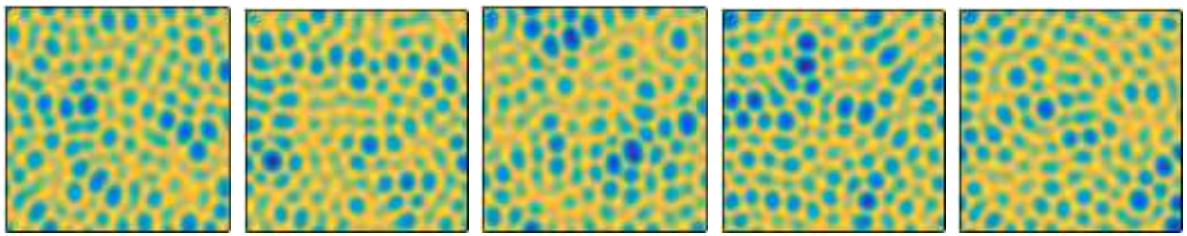


Figure: Surfaces $u(x, y, 3)$ for $r = 1.75$ or $r = 2$. Can you group the images by eye?

Answer: (from left) $r = 1.75, 2, 1.75, 2, 2.$

AKS CLASSIFICATION METHODS

Goal:

Classify (150, 30 for each parameter) trials of the aKS Equation by parameter (5 values) using snapshots of the surfaces $u(x, y, \cdot)$ as they evolve in time (5 values).

Methods of classification:

AKS CLASSIFICATION METHODS

Goal:

Classify (150, 30 for each parameter) trials of the aKS Equation by parameter (5 values) using snapshots of the surfaces $u(x, y, \cdot)$ as they evolve in time (5 values).

Methods of classification:

1. Surfaces viewed as points in $\mathbb{R}^{266144}$. Reduce resolution to 10×10 by coarsening the discretization of the spatial domain. Classify with Subspace Discriminant Ensemble.

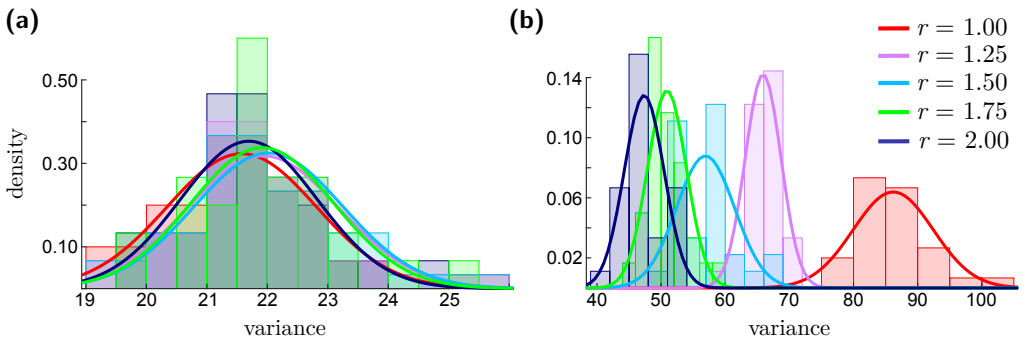
AKS CLASSIFICATION METHODS

Goal:

Classify (150, 30 for each parameter) trials of the aKS Equation by parameter (5 values) using snapshots of the surfaces $u(x, y, \cdot)$ as they evolve in time (5 values).

Methods of classification:

1. Surfaces viewed as points in $\mathbb{R}^{266144}$. Reduce resolution to 10×10 by coarsening the discretization of the spatial domain. Classify with Subspace Discriminant Ensemble.
2. Parameter r influences mean and amplitude of pattern. Use a normal distribution-based classifier built on the variances of the surface heights.



AKS CLASSIFICATION METHODS

Goal:

Classify (150, 30 for each parameter) trials of the aKS Equation by parameter (5 values) using snapshots of the surfaces $u(x, y, \cdot)$ as they evolve in time (5 values).

Methods of classification:

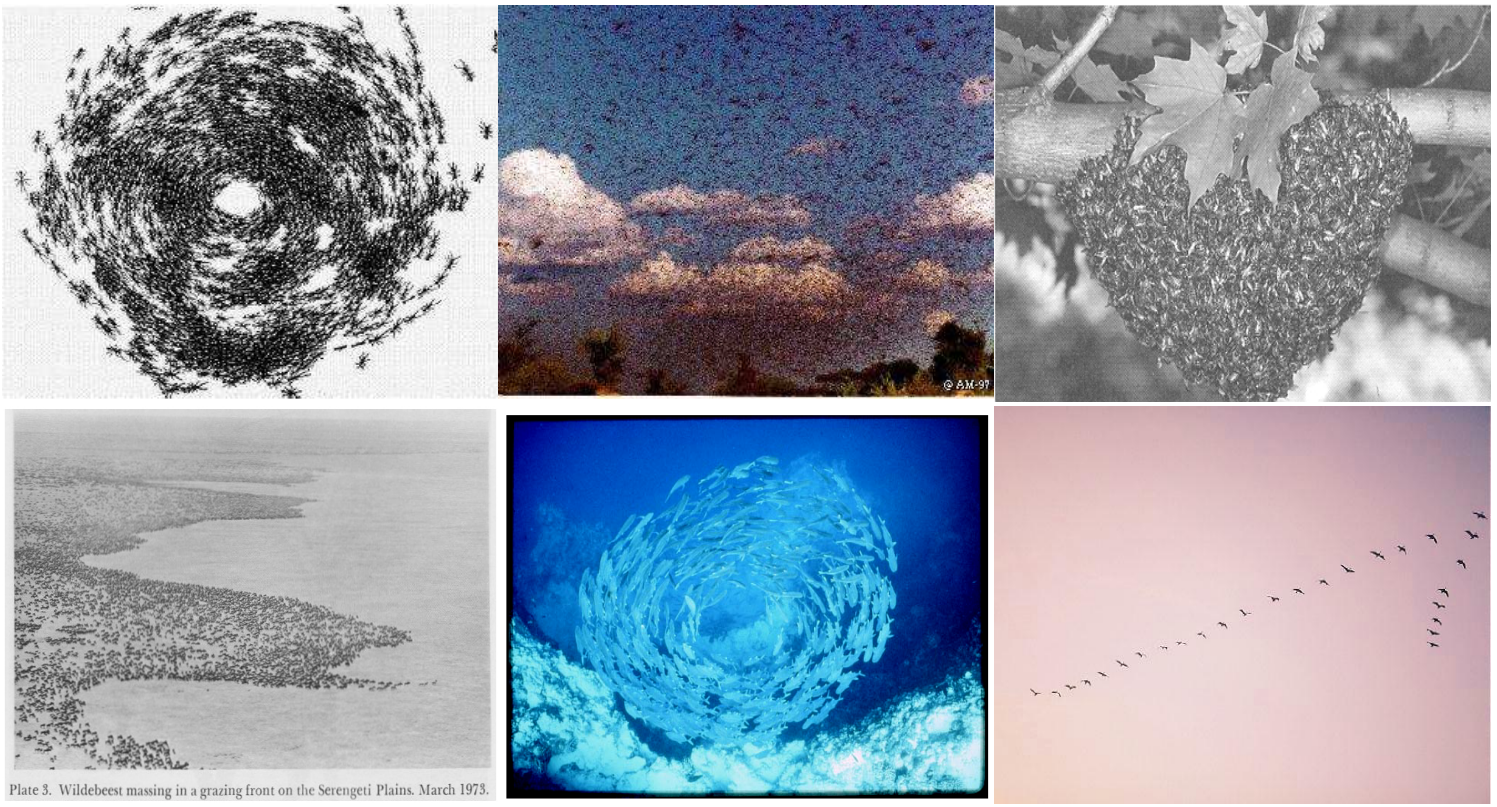
1. Surfaces viewed as points in $\mathbb{R}^{266144}$. Reduce resolution to 10×10 by coarsening the discretization of the spatial domain. Classify with Subspace Discriminant Ensemble.
2. Parameter r influences mean and amplitude of pattern. Use a normal distribution-based classifier built on the variances of the surface heights.
3. Sublevel set filtration PD. Generate PIs with resolution 10×10 and variance 0.01. Classify H_0, H_1 and concatenated PIs using Subspace Discriminant Ensemble.

AKS EQUATION CLASSIFICATION ACCURACY

Classification Approach	Time t=3	Time t=5	Time t=10
Subspace Discriminant Ensemble, Resized Surfaces	26.0 %	19.3%	19.3 %
Variance Normal Distribution Classifier	20.74%	75.2%	77.62 %
Subspace Discriminant Ensemble, H_0 PIs	58.3 %	96.0 %	94.7 %
Subspace Discriminant Ensemble, H_1 PIs	67.7 %	87.3 %	93.3%
Subspace Discriminant Ensemble, H_0 and H_1 PIs	72.7 %	95.3 %	97.3 %

TOPOLOGY EVOLVING IN TIME

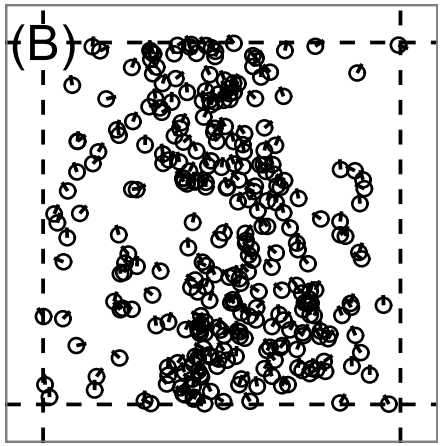
BIOLOGICAL AGGREGATIONS



In many natural systems, particles, organisms, or agents interact locally according to rules that produce aggregate behavior.

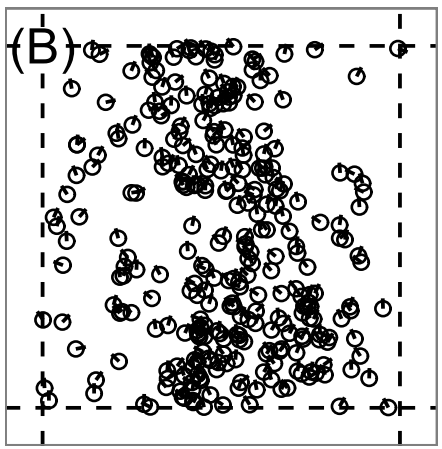
CLASSIC WAY TO ANALYZE BIOLOGICAL AGGREGATIONS

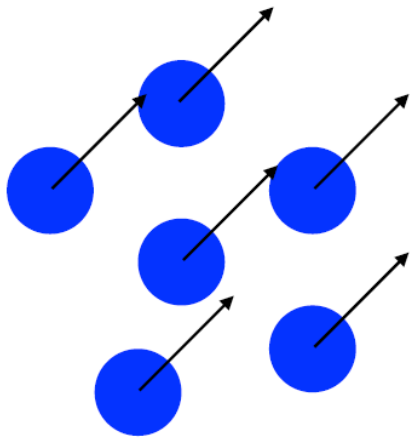
Alignment Order Parameter: $\varphi(t) = \frac{1}{Nv_0}\left|\sum_{i=1}^{N}\vec{v}_i(t)\right|$



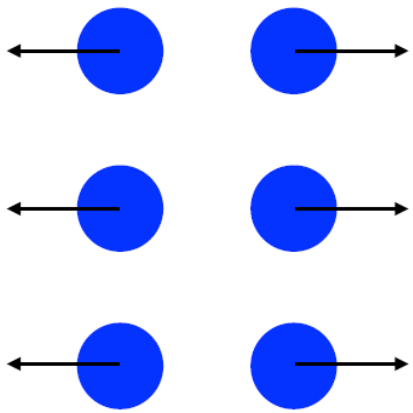
CLASSIC WAY TO ANALYZE BIOLOGICAL AGGREGATIONS

Alignment Order Parameter: $\varphi(t) = \frac{1}{Nv_0}\left|\sum_{i=1}^{N}\vec{v}_i(t)\right|$





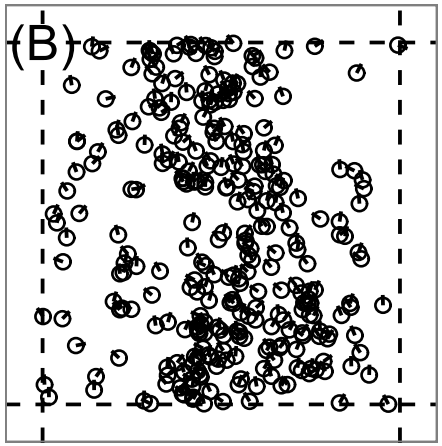
$\varphi = 1$

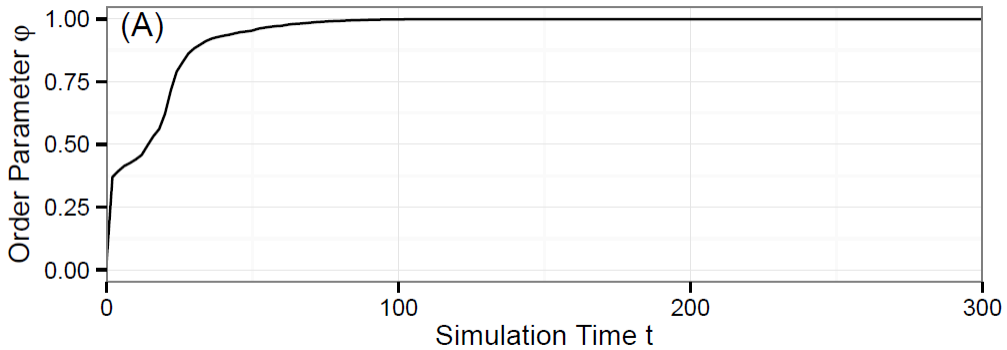


$\varphi = 0$

CLASSIC WAY TO ANALYZE BIOLOGICAL AGGREGATIONS

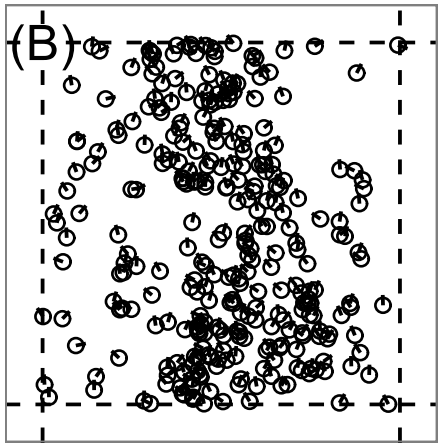
Alignment Order Parameter: $\varphi(t) = \frac{1}{Nv_0}\left|\sum_{i=1}^{N}\vec{v}_i(t)\right|$

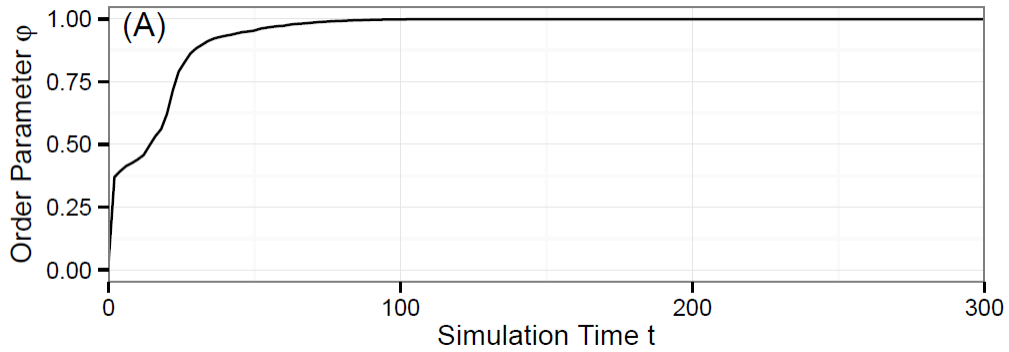




CLASSIC WAY TO ANALYZE BIOLOGICAL AGGREGATIONS

Alignment Order Parameter: $\varphi(t) = \frac{1}{Nv_0}\left|\sum_{i=1}^{N}\vec{v}_i(t)\right|$





Goal:

Use topology to analyze the collective behavior of interacting agents' positions and velocities as time evolves.

TOPOLOGICAL DATA ANALYSIS

1. Envision data as a point cloud
    - *e.g.* position-velocity for on snapshot in time

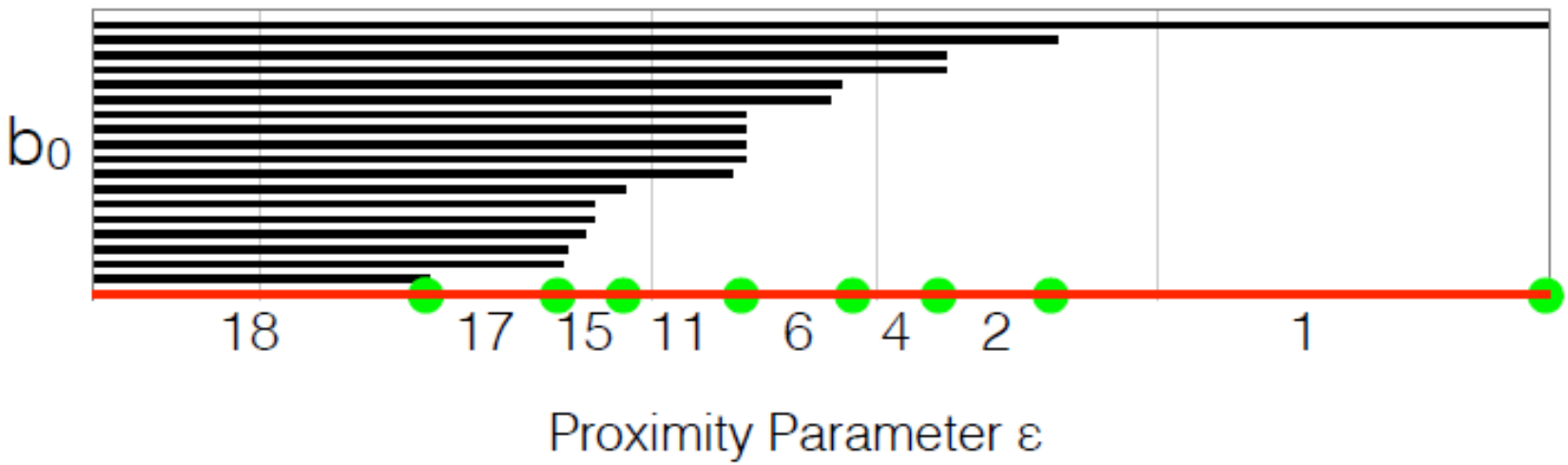
2. Create connections between proximate points
    - build simplicial complex

3. Determine topological structure of complex
    - compute homology (measure # holes)

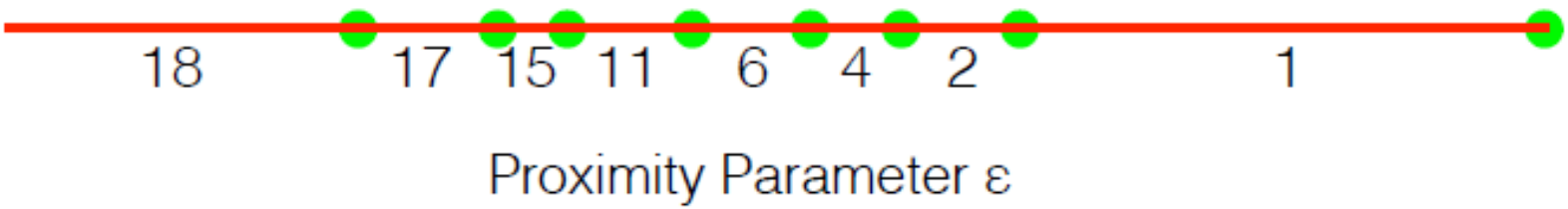
4. Vary proximity parameter to assess different scales
    - calculate persistent homology

5. **Evolve in time**
    - **CROCKER plots**

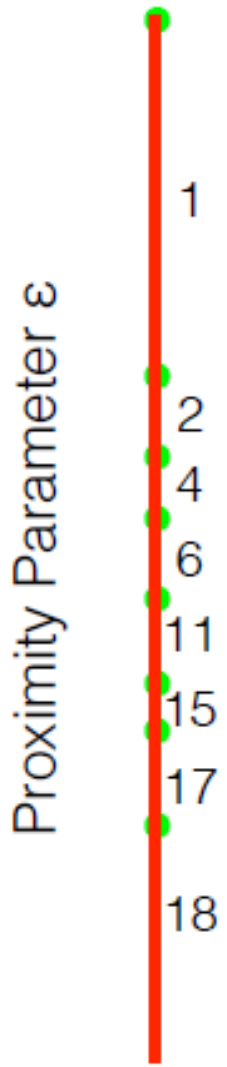
COMPUTE PERSISTENT HOMOLOGY



COMPUTE PERSISTENT HOMOLOGY

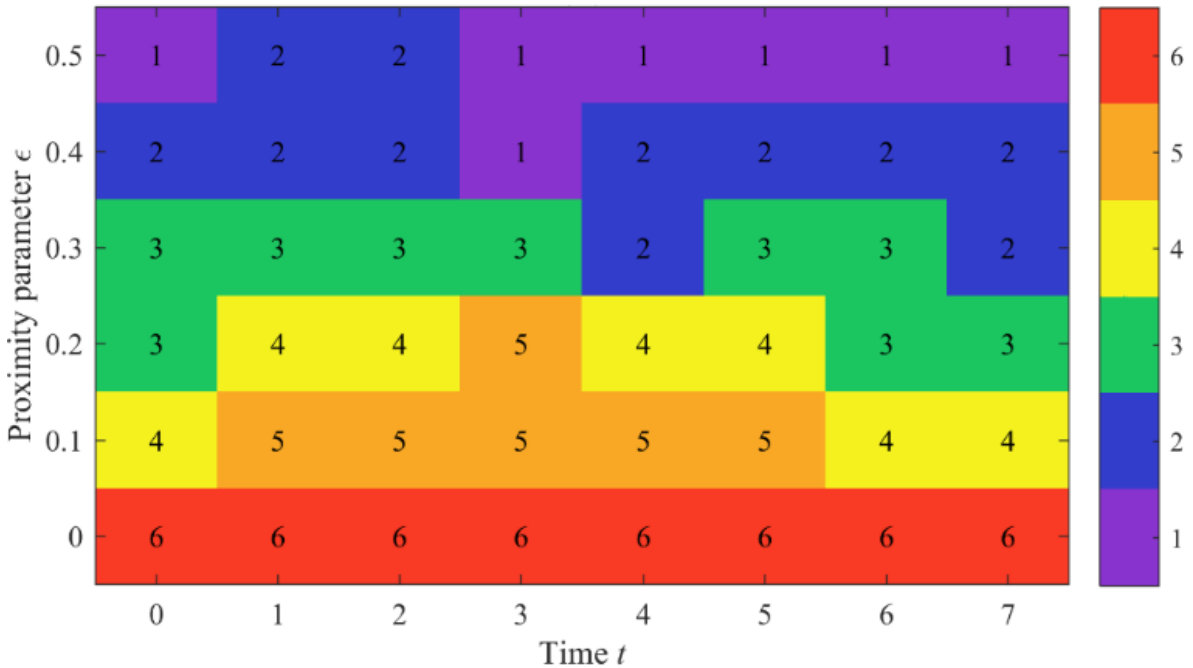


COMPUTE PERSISTENT HOMOLOGY



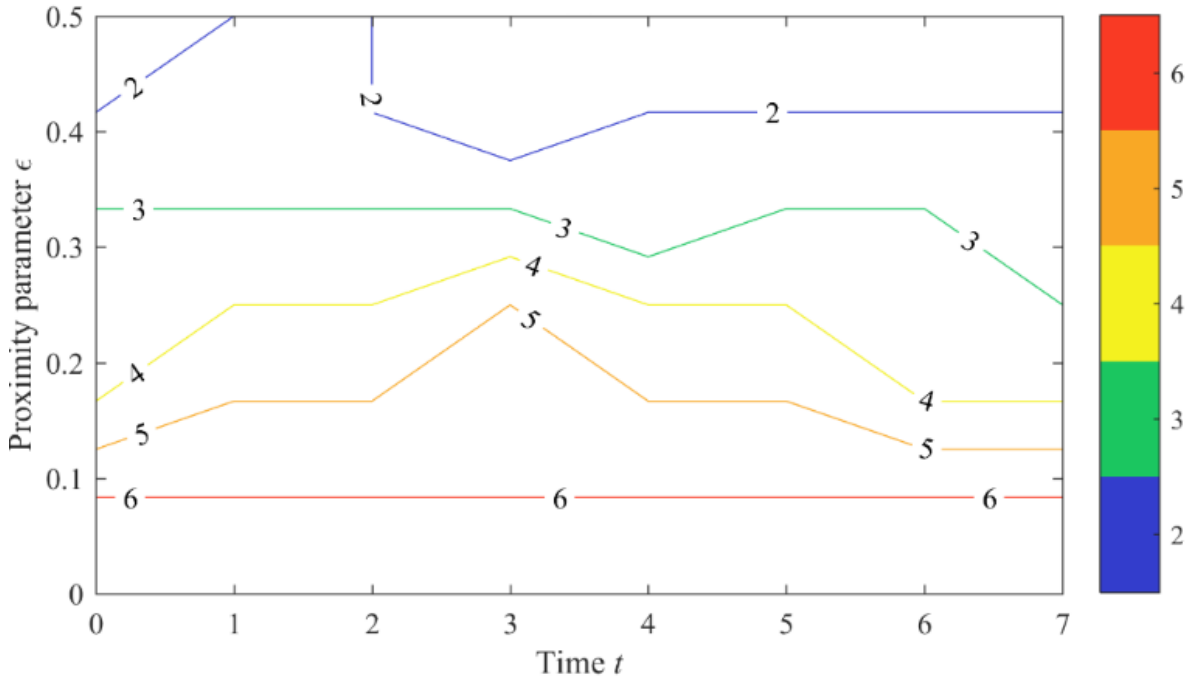
EVOLVE IN TIME

Compute the kth Betti number $\mathrm{b}_k(\varepsilon, t)$,



EVOLVE IN TIME

Compute the *k*th Betti number $b_k(\varepsilon, t)$,



CROCKER plot

(Topaz, Z., Halverson 2015) Contour Realization Of Computed *K*-dimensional hole Evolution in the Rips complex (CROCKER)

VICSEK MODEL

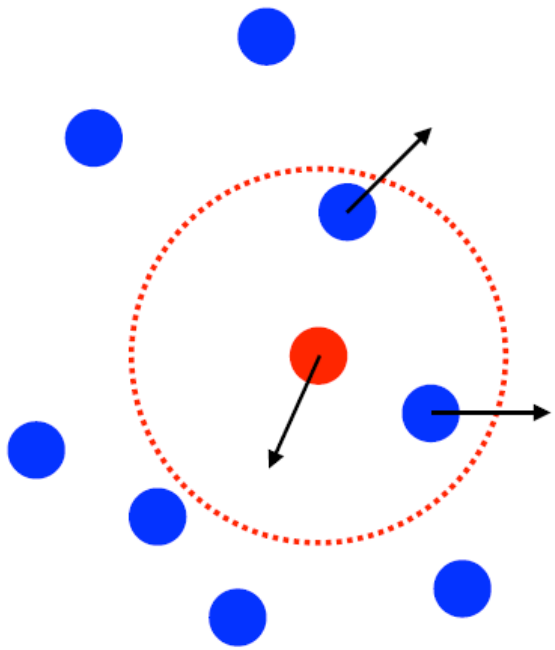
VICSEK MODEL

- Highly cited dynamical system in discrete time and continuous space.
- Describes motion of interacting point particles in a square with periodic boundary conditions.
- Model written as:

$$
\begin{aligned}
\theta_i(t+\Delta t) &= \frac{1}{N}\left(\sum_{|\vec{x}_i-\vec{x}_j|\le R}\theta_j(t)\right) + U(-\eta/2,\eta/2) \\
\vec{v}_i(t+\Delta t) &= v_0\left(\cos\theta_i(t+\Delta t), \sin\theta_i(t+\Delta t)\right) \\
\vec{x}_i(t+\Delta t) &= \vec{x}_i(t) + \vec{v}_i(t+\Delta t)\,\Delta t
\end{aligned}
$$

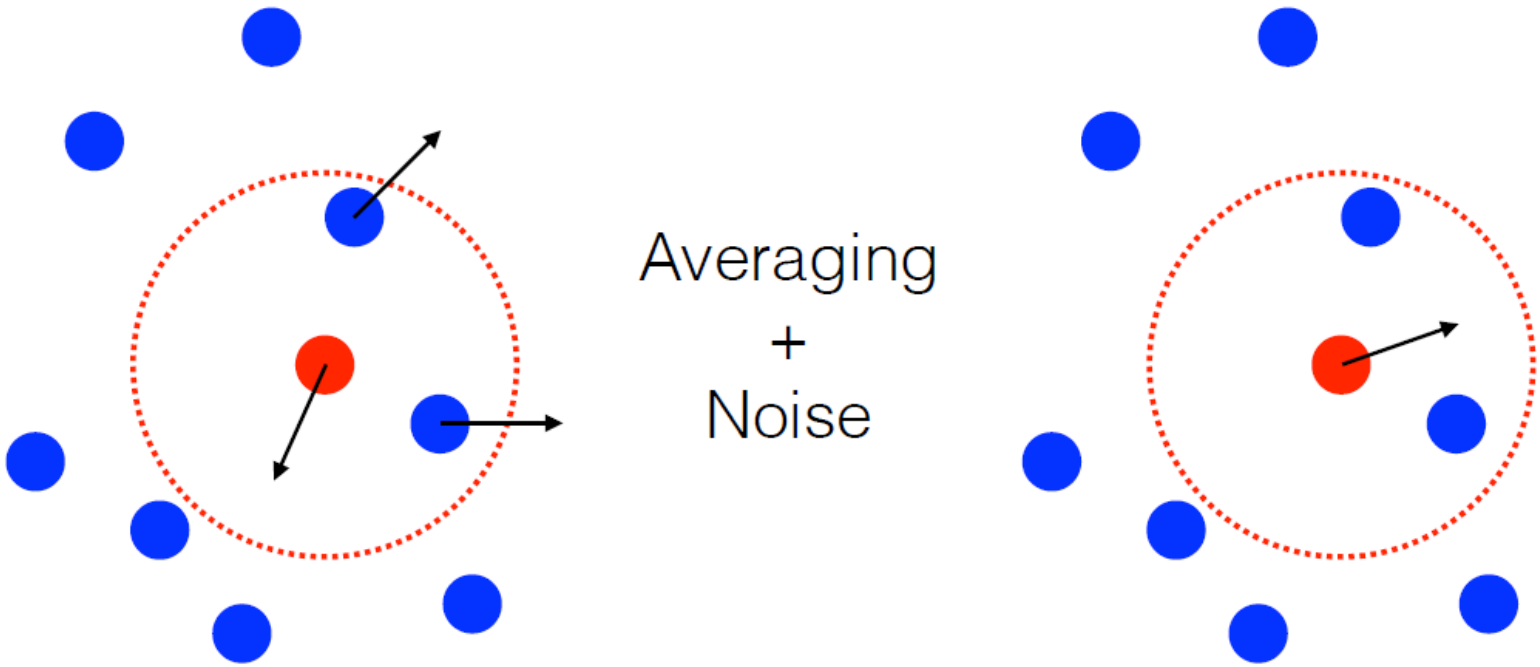
VICSEK MODEL

- Highly cited dynamical system in discrete time and continuous space.
- Describes motion of interacting point particles in a square with periodic boundary conditions.

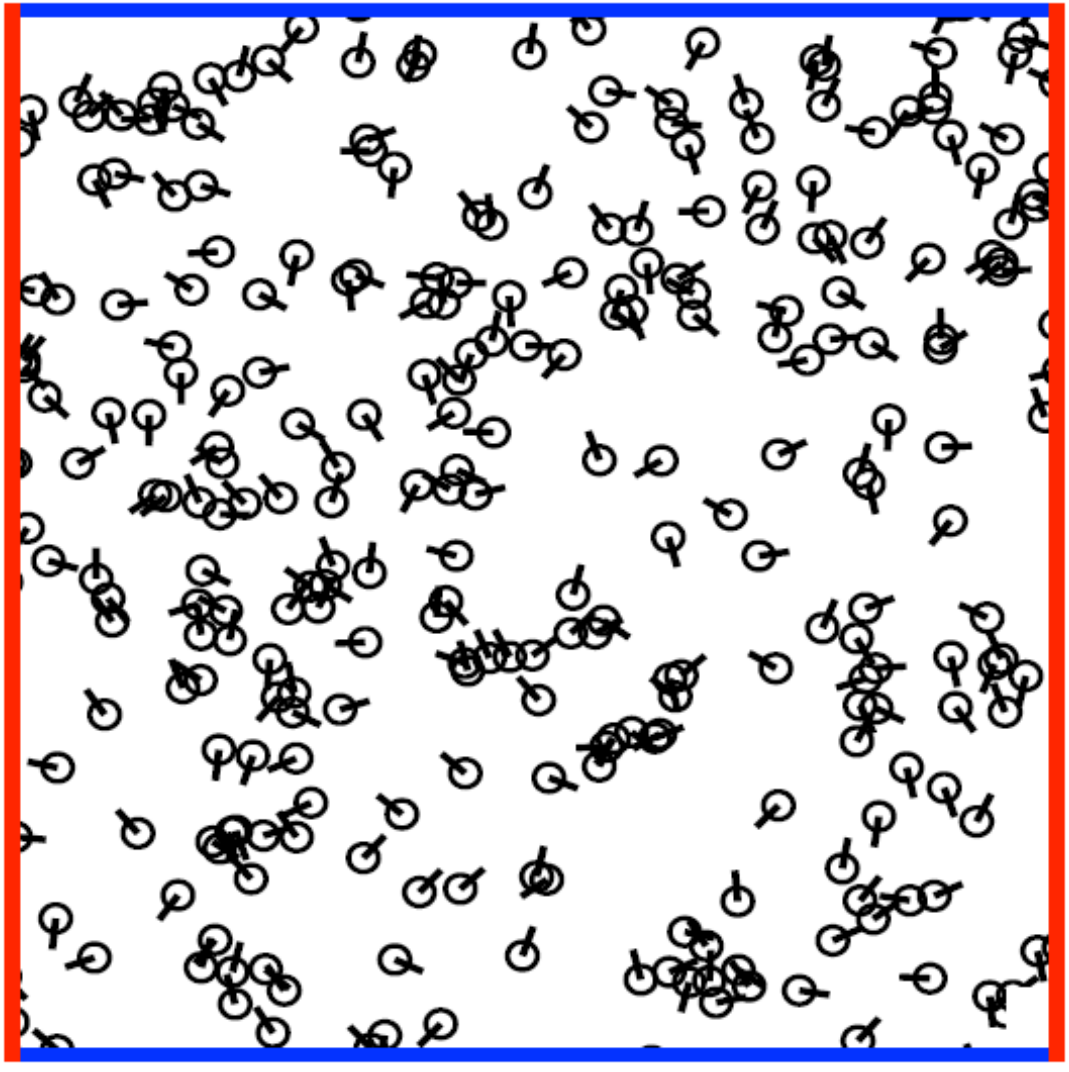


VICSEK MODEL

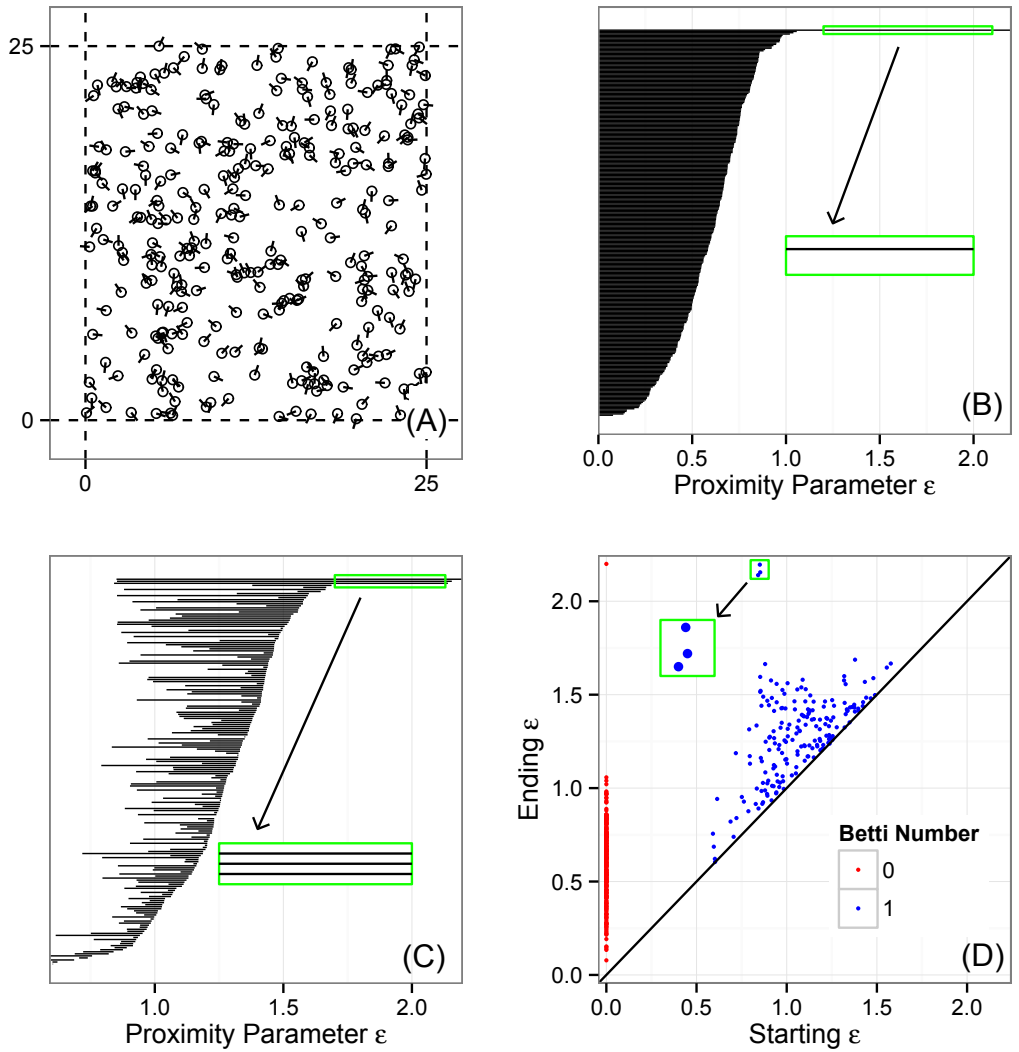
- Highly cited dynamical system in discrete time and continuous space.
- Describes motion of interacting point particles in a square with periodic boundary conditions.



TOPOLOGY OF INITIAL CONDITION

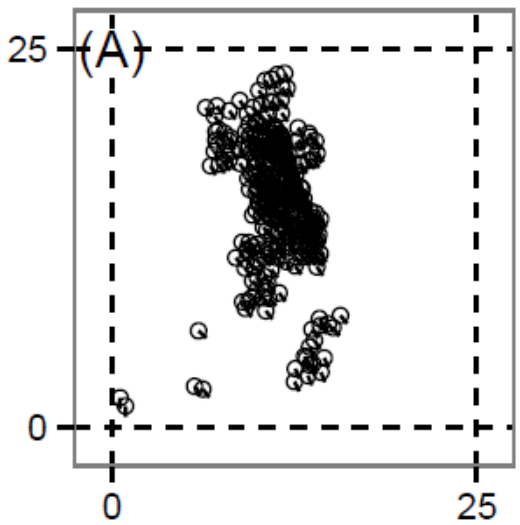


TOPOLOGY OF INITIAL CONDITION

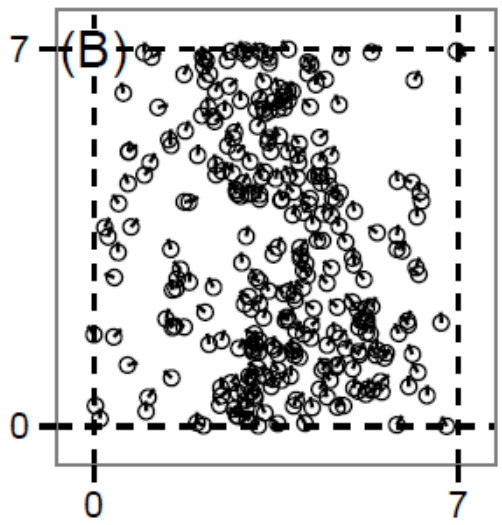


Three-torus T^3: $b = (1, 3, 3, 1, 0, \ldots)$

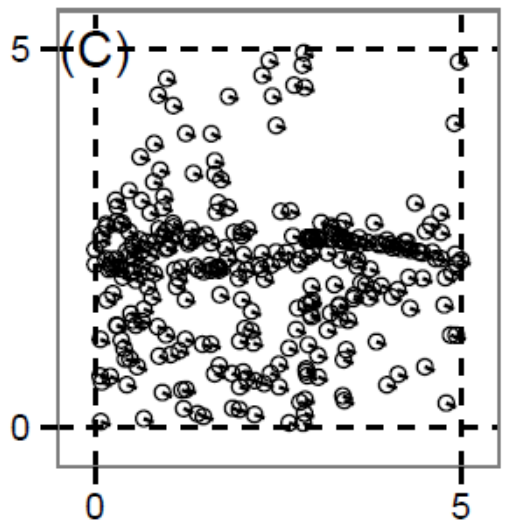
DIFFERENT LONG-TERM BEHAVIORS OF VICSEK MODEL



Clusters?

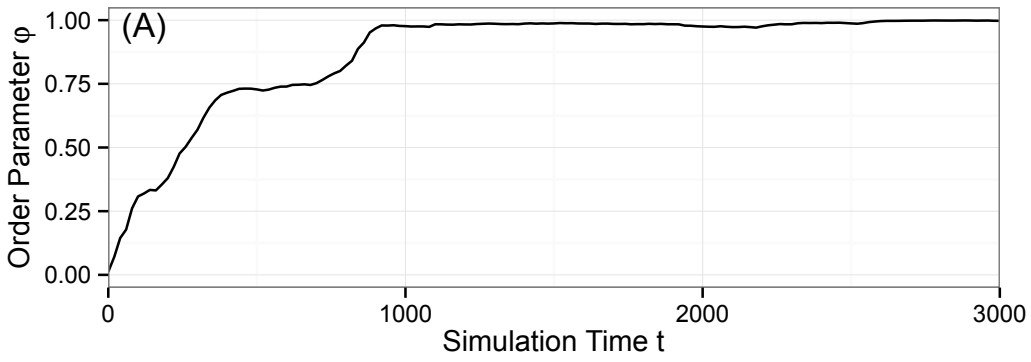


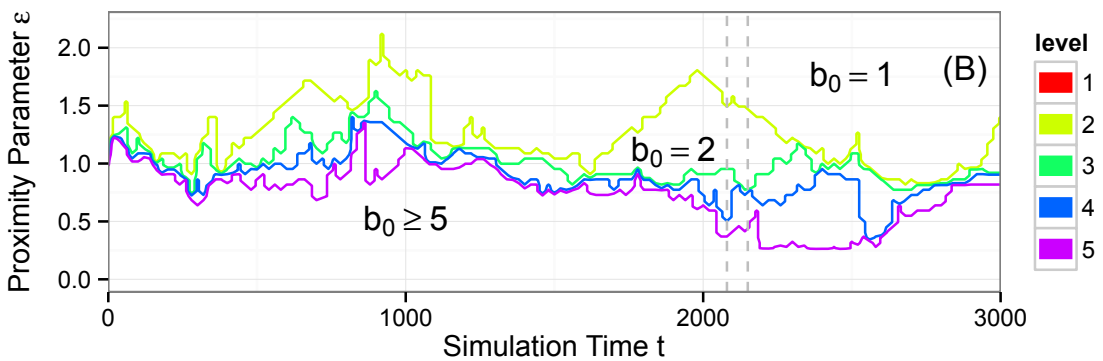
Loose alignment?

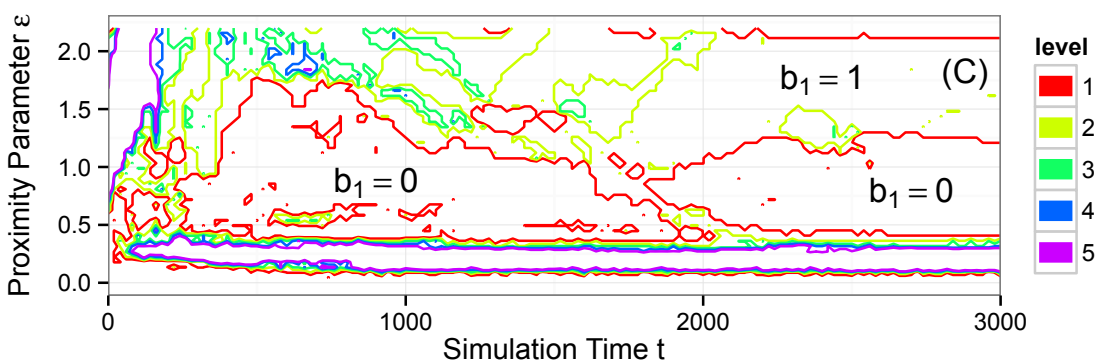


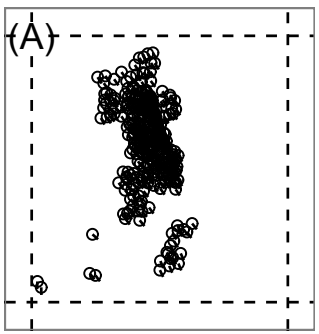
Strong alignment?

VICSEK SIMULATION A ANALYSIS

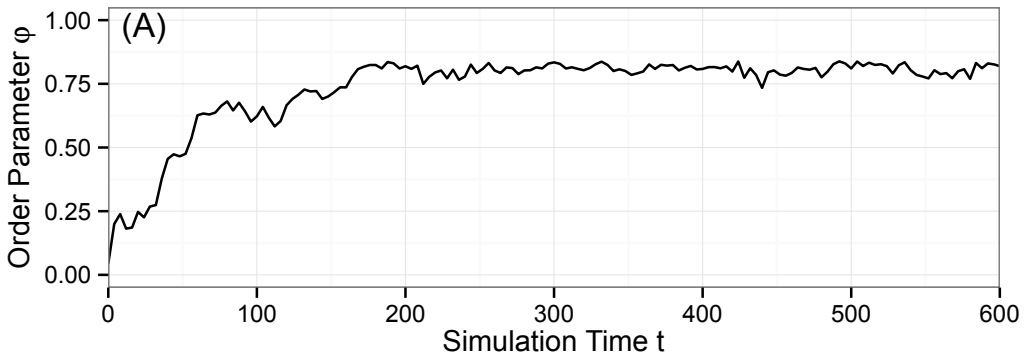


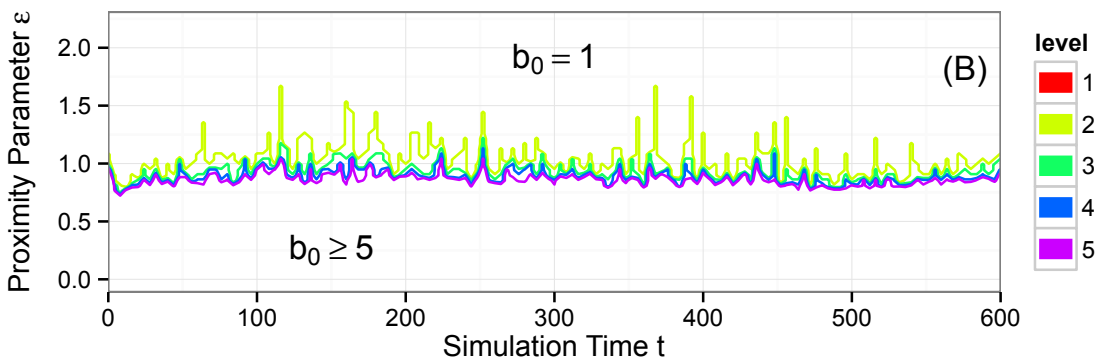


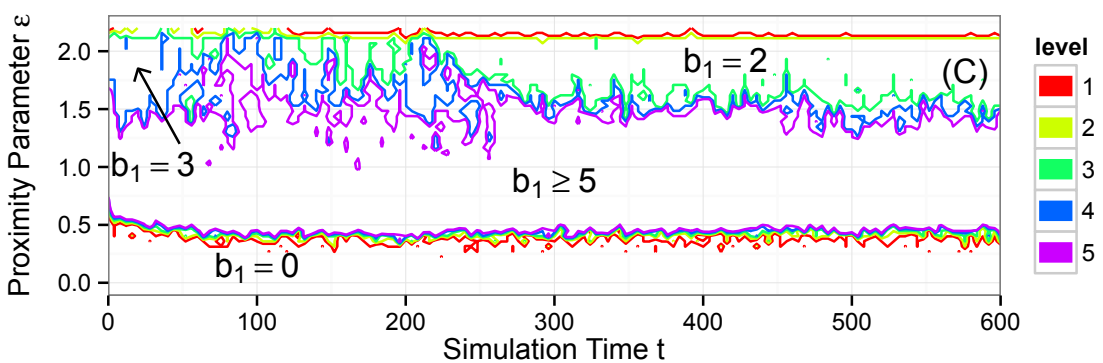


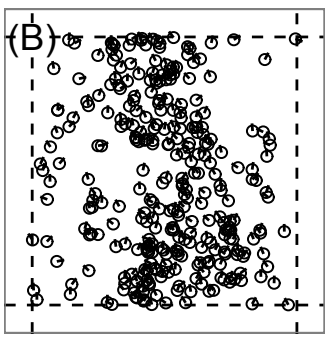


VICSEK SIMULATION B ANALYSIS

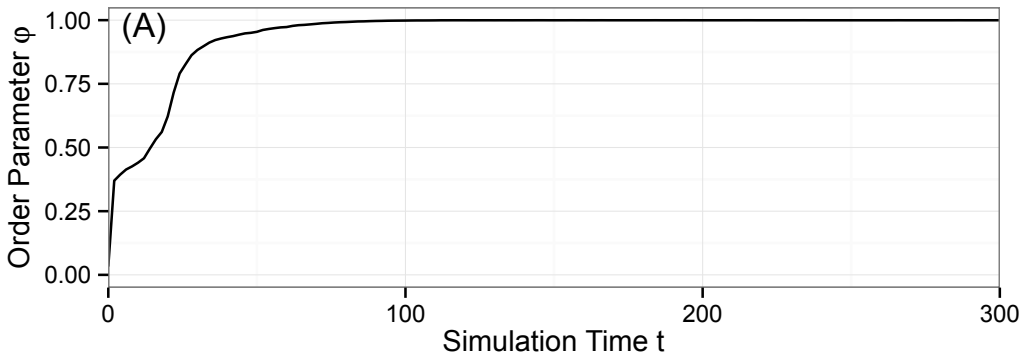


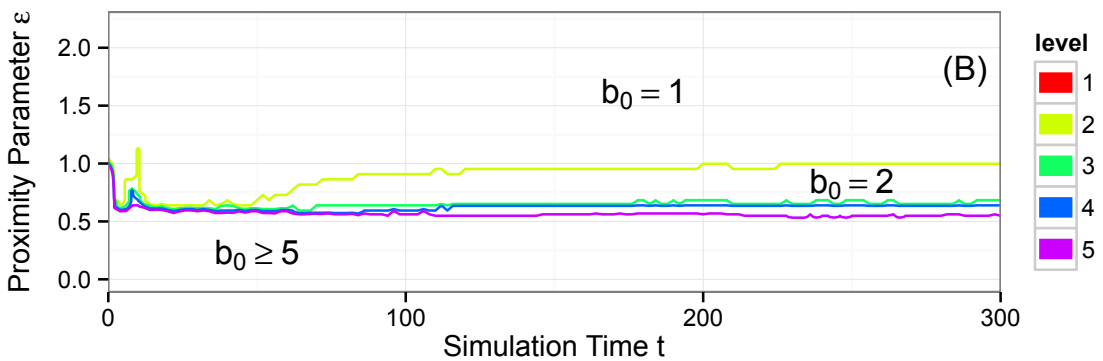


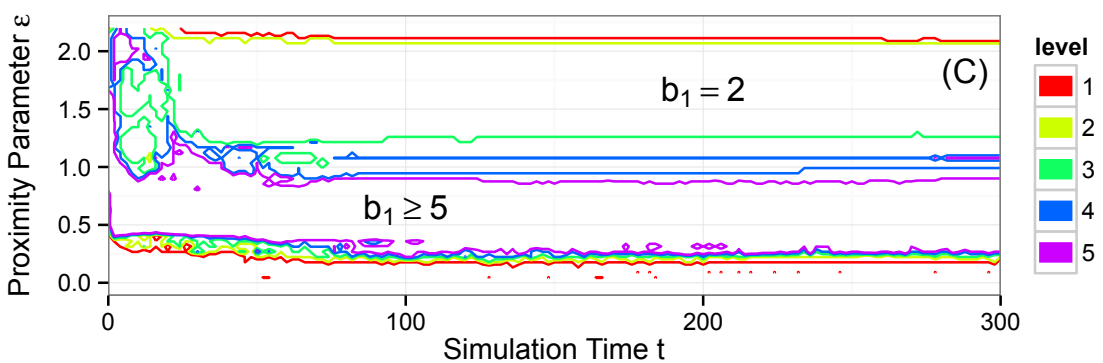


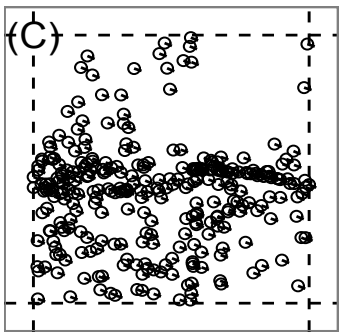


VICSEK SIMULATION C ANALYSIS









IDENTIFYING PARAMETERS WITH TDA AND MACHINE LEARNING

Goal:

Given simulated data from the Vicsek model, can we use machine learning algorithms to recover the unknown underlying noise parameter?

IDENTIFYING PARAMETERS WITH TDA AND MACHINE LEARNING

Goal:

Given simulated data from the Vicsek model, can we use machine learning algorithms to recover the unknown underlying noise parameter?

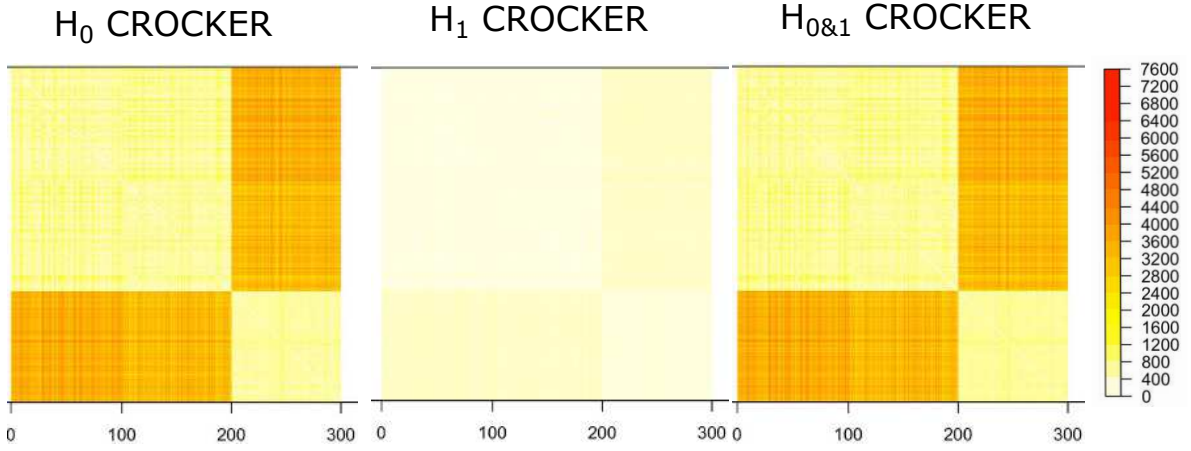
- Generate 100 simulations of different parameter choices
- Compute alignment order parameter and H_0 and H_1 CROCKER plots
- Compare pairwise (Euclidean) distances between simulations
- Cluster with K-medoids

EXPERIMENT 1

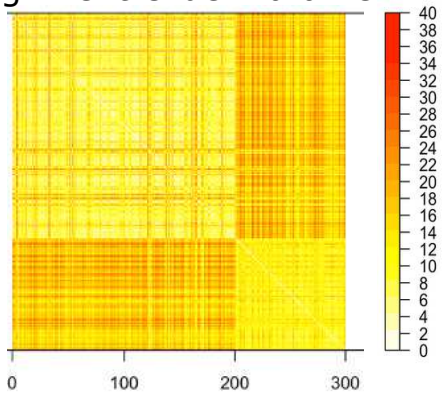
Noise parameters:

$$\eta = 0.01, 0.1, 1$$

Pairwise Distance Matrices:



Alignment Order Parameter



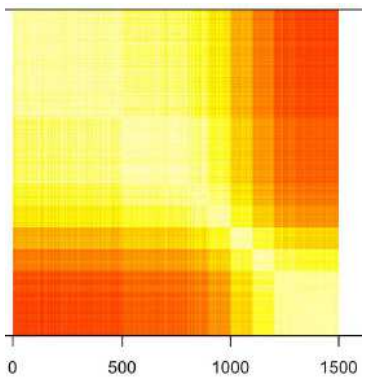
EXPERIMENT 2

Noise parameters:

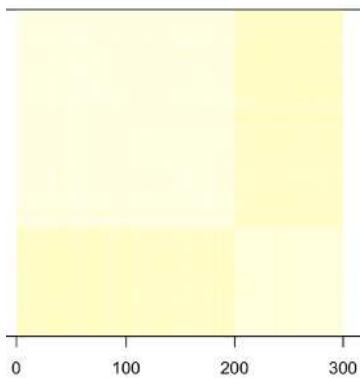
$\eta = 0.01, 0.02, 0.03, 0.05, 0.1, 0.19, 0.2, 0.21, 0.3, 0.5, 1, 1.5, 1.9, 1.99, 2$

Pairwise Distance Matrices:

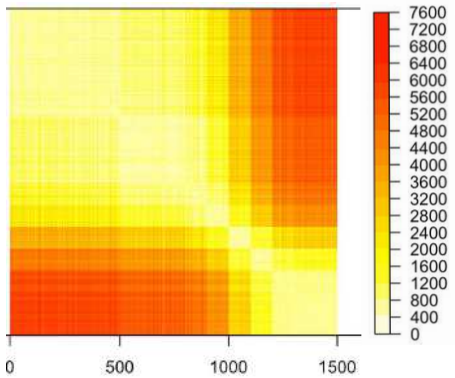
H_0 CROCKER



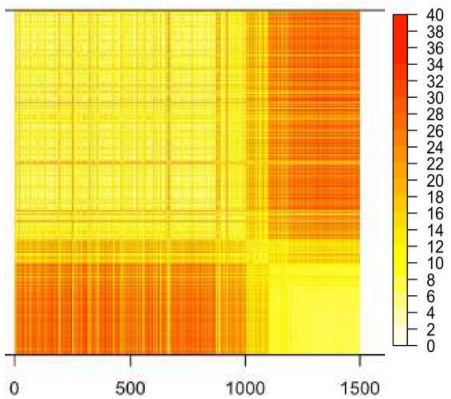
H_1 CROCKER



$H_{0\&1}$ CROCKER



Alignment Order Parameter



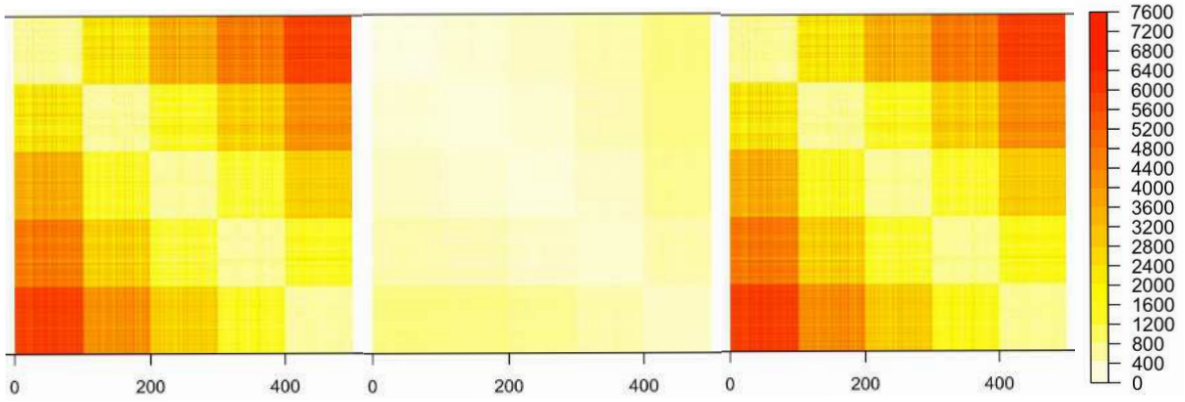
EXPERIMENT 3

Noise parameters:

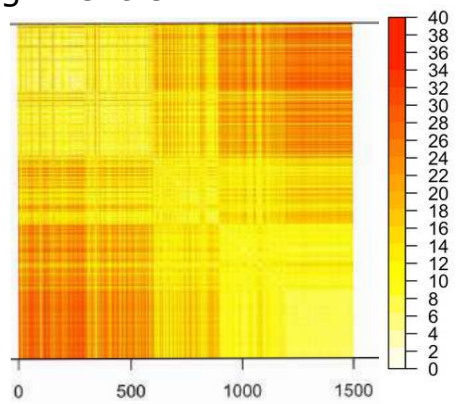
$$\eta = 0.01, 0.5, 1, 1.5, 2$$

Pairwise Distance Matrices:

H_0 CROCKER H_1 CROCKER $H_{0\&1}$ CROCKER



Alignment Order Parameter



K-MEDOIDS CLUSTERING RESULTS

	Exp 1		Exp 2		Exp 3	
	$H_{0\&1}$	Align	$H_{0\&1}$	Align	$H_{0\&1}$	Align
Accuracy	77.0%	63.3%	23.3%	14.3%	99.6 %	63.0%
Silhouette Width	0.61	0.45	0.43	0.3	0.77	0.46

CONCLUSION

CONCLUSION

Persistence Images

PIs present a method for vectorization of topological characteristics of data that:

- have an interpretable connection to PDs
- are stable
- can be incorporated with a variety of metrics and machine learning tools

CROCKER Plots

CROCKER plots present a method for vectorization of topological characteristics of time-varying (or two parameter) data that:

- are simple
- can provide higher classification accuracy than classic order parameters
- can be incorporated with a variety of metrics and machine learning tools

QUESTIONS?

Thank you!

Persistence Images
Henry Adams, Sofya Chepushtanova, Tegan Emerson, Eric Hanson, Michael Kirby, Francis Motta, Rachel Neville, Chris Peterson, and Patrick Shipman
CROCKER Plots and Viscek Model
Henry Adams, Tom Halverson, Chad Topaz, and Lu Xian

Lori Ziegelmeier
lziegel1@macalester.edu
Department of Mathematics, Statistics, and Computer Science

MACALESTER