Dynamic mode decomposition and the Koopman operator: algorithms and applications

Clancy Rowley

Joint work with Matt Williams, Yannis Kevrekidis

Princeton University

19 November 2014
IPAM MTWS4: Turbulence in Engineering Applications

Overview

- The Koopman operator is an infinite-dimensional linear operator that captures everything about a *nonlinear* dynamical system.
- Its eigenfunctions define a (nonlinear) change of coordinates in which the system becomes linear.
- Dynamic Mode Decomposition (DMD) is an algorithm that determines these eigenvalues and eigenfunctions directly from data. Sometimes...

- Applications to turbulence:
  - Finding coherent structures based on their dynamics (e.g., relevant timescales)
  - Quantify transport/mixing in turbulent flows
  - "Fuse" data from different sensors

Outline

Dynamic Mode Decomposition: original definition

Dynamic Mode Decomposition (DMD) was originally defined by an algorithm[1]:

- Collect snapshots of data $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$, equally spaced in time.
- Assume the data are linearly related:

$$\mathbf{x}_{k+1} = A\mathbf{x}_k$$

- Use an Arnoldi-like algorithm to approximate eigenvalues and eigenvectors of A (without ever determining A explicitly).

Hitch: Typically the dynamics are nonlinear, and the linear assumption does not hold.

[1] P.J. Schmid, APS 2008, JFM 2010

Dynamic Mode Decomposition: an alternative definition

Definition (DMD)

Suppose we have a discrete-time dynamical system $\mathbf{z} \mapsto \mathbf{F}(\mathbf{z})$ and two sets of data:

$$X = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_m \end{bmatrix} \qquad Y = \begin{bmatrix} \mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_m \end{bmatrix}$$

with $\mathbf{y}_j = \mathbf{F}(\mathbf{x}_j)$. The DMD modes are eigenvectors of

$$A = YX^+,$$

where $+$ denotes the Moore-Penrose pseudoinverse.

For a sequential time series $\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_m$, one takes $\mathbf{x}_{j+1} = \mathbf{y}_j = \mathbf{z}_j$.

- Under mild assumptions on the data (e.g., the columns of X are linearly independent), the data satisfy $\mathbf{y}_j = A\mathbf{x}_j$.
- Thus, there still seems to be the assumption that the dynamics are linear (more later...)

Alternative definition: picky points

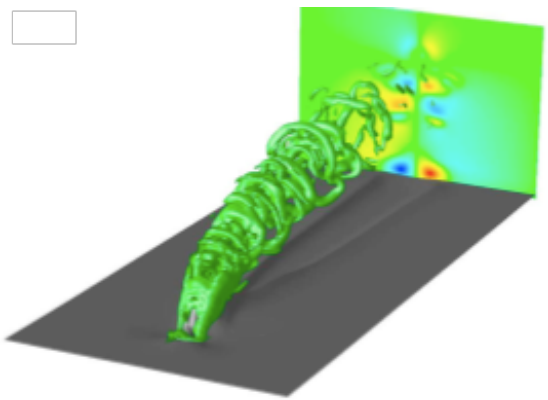
- The nonzero eigenvalues of A are precisely the same as the eigenvalues determined by the original DMD algorithm.
- The modes defined above are actually what we call "Exact DMD modes", and are not precisely identical to the DMD modes given by the original algorithm. The original algorithm gives "Projected DMD modes", which are the projection of these onto the range of X. In practice, these are nearly identical, so the distinction is unimportant.
- There are efficient algorithms for constructing DMD modes according to this definition, without literally computing the matrix A.[2]

[2] J.H. Tu, C.W. Rowley, D.M. Luchtenburg, S.L. Brunton, and J.N. Kutz, *J. Computational Dynamics*, in press.

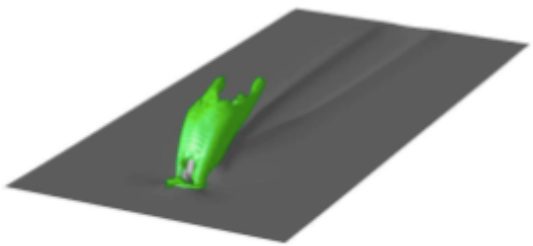
Example: jet in crossflow

Linearize a jet in crossflow about an unstable equilibrium.[3]

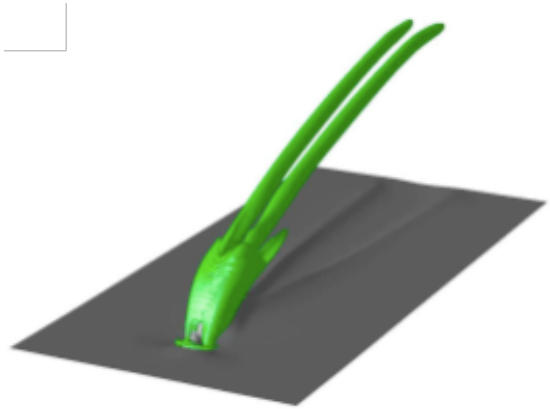
($\mathrm{Re}_{\delta^*_{\mathbf{0}}} = 165,\ V_{\mathbf{jet}}/U_\infty = 3,\ \delta^*_{\mathbf{0}}/D = 1/3$)



Instantaneous snapshot



Mean



Unstable equilibrium

Compute eigenvalues and compare with observed frequencies:

	Observed	Linear theory
Shear layer	St = 0.141	St = 0.169
Near wall	St = 0.0174	St = 0.043

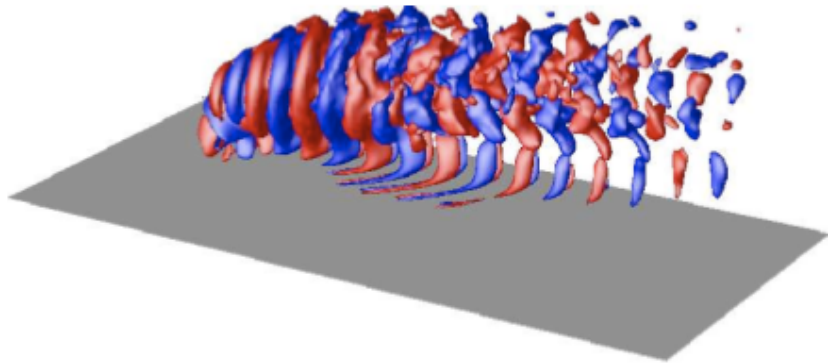
Frequency mismatch for near-wall structures: failure of linear theory.

[3] Bagheri, Schlatter, Schmid, Henningson, JFM 2009

DMD modes for jet in crossflow

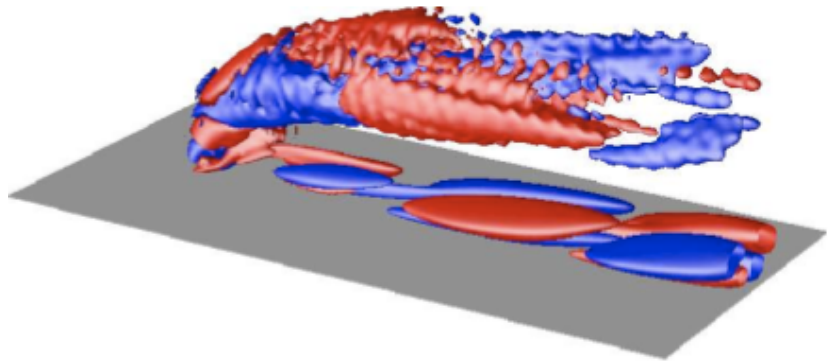
- DMD modes capture relevant structures and frequencies

High-frequency mode captures structures in the shear layer.



St = 0.141

Low-frequency mode captures near-wall structures associated with horseshoe vortex.



St = 0.017

But this is a nonlinear flow, and we're assuming the dynamics are linear! How should we interpret these results?

Koopman operator

Definition (Koopman, PNAS 1931)

For a discrete-time dynamical system

$$\mathbf{z} \mapsto \mathbf{F}(\mathbf{z}),$$

where $z \in \mathcal{M}$, the Koopman operator $\mathcal{K}$ acts on scalar functions $g : \mathcal{M} \to \mathbb{C}$, as

$$\mathcal{K}g(\mathbf{z}) \triangleq g(\mathbf{F}(\mathbf{z})).$$

- Note that $\mathcal{K}$ acts on *functions* on $\mathcal{M}$, and is thus an infinite-dimensional operator. $\mathcal{K}$ is clearly linear.
- In order to relate Koopman to DMD, consider a set of *observables* $\psi_j : \mathcal{M} \to \mathbb{C}$, $j = 1, \ldots, n$, and let $\boldsymbol{\psi}$ denote the vector of observables. Consider a set of initial states $\{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$, and let

$$\mathbf{x}_k = \boldsymbol{\psi}(\mathbf{z}_k), \qquad \mathbf{y}_k = \boldsymbol{\psi}(\mathbf{F}(\mathbf{z}_k)).$$

Define matrices X and Y as before, and $A = YX^+$.

DMD eigenvalues are Koopman eigenvalues (sometimes)

Theorem (Koopman and DMD)

Let φ be an eigenfunction of $\mathcal{K}$ with eigenvalue λ, and suppose $\varphi \in \text{span}\{\psi_j\}$, so that

$$\varphi(\mathbf{z}) = w_1\psi_1(\mathbf{z}) + w_2\psi_2(\mathbf{z}) + \cdots + w_n\psi_n(\mathbf{z})$$

for some $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{C}^n$. If $\mathbf{w} \in \mathcal{R}(X)$, then $\mathbf{w}$ is a left eigenvector of A with eigenvalue λ: $\mathbf{w}^* A = \lambda \mathbf{w}^*$.

So Koopman eigenvalues are DMD eigenvalues, provided:

1. the set of observables is sufficiently large ($\varphi \in \text{span}\{\psi_j\}$)
2. the data is sufficiently rich ($\mathbf{w} \in \mathcal{R}(X)$).

Furthermore, we can calculate the Koopman eigenfunctions from the *left* eigenvectors of the DMD matrix A, as $\varphi(\mathbf{z}) = \mathbf{w}^* \boldsymbol{\psi}(\mathbf{z})$.

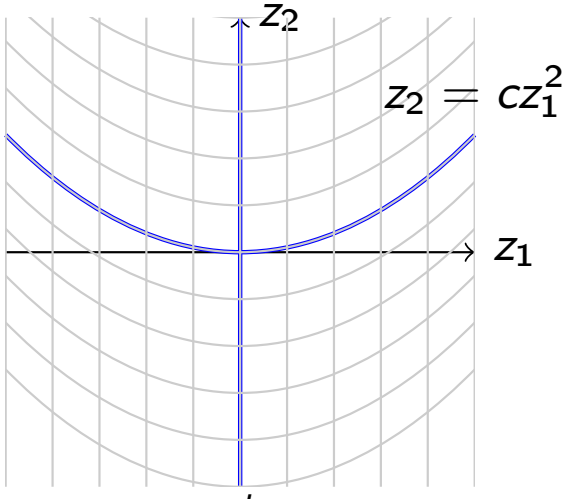
- Note: for a linear system, $\mathcal{K}$ has linear eigenfunctions, so the full-state observable $\boldsymbol{\psi}(\mathbf{z}) = \mathbf{z}$ is sufficient to capture these; for nonlinear systems, however, linear observables are typically not sufficient.

Example: 2d ODE

Consider the map

$$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \mapsto \begin{bmatrix} \lambda z_1 \\ \mu z_2 + (\lambda^2 - \mu) c z_1^2 \end{bmatrix}.$$

This system has a stable equilibrium at the origin, and invariant manifolds given by $z_1 = 0$ and $z_2 = cz_1^2$:



Koopman eigenvalues are λ, μ with eigenfunctions

$$\varphi_\lambda(\mathbf{z}) = z_1$$
$$\varphi_\mu(\mathbf{z}) = z_2 - cz_1^2.$$

In addition, φ_λ^k is an eigenfunction with eigenvalue λ^k, the product $\varphi_\lambda \varphi_\mu$ is an eigenfunction with eigenvalue $\lambda\mu$, etc.

DMD for ODE example

Apply DMD to this example, with initial states $\mathbf{z}$ given by $(1,1),(5,5),(-1,1),(-5,5)$, with $\lambda = 0.9$, $\mu = 0.5$.

- Case 1: observable $\psi(\mathbf{z}) = (z_1, z_2)$. If $c = 0$, so that the problem is linear, then DMD eigenvalues are 0.9 and 0.5: good!
  If $c = 1$, however, then the DMD eigenvalues are 0.9 and 2.002. These do not correspond to Koopman eigenvalues, and one might even presume the equilibrium is *unstable*!
- Case 2: observable $\psi(\mathbf{z}) = (z_1, z_2, z_1^2)$. Now, the DMD eigenvalues are 0.9, 0.5, and $0.81 = 0.9^2$, which agree with Koopman eigenvalues.
- Case 3: observable $\psi(\mathbf{z}) = (z_1, z_2, z_2^2)$. Now, the DMD eigenvalues are 0.9, 0.822, and 4.767. There is still a linear relationship between the snapshots ($\mathbf{y}_j = A\mathbf{x}_j$), but the eigenvalues do not correspond to Koopman eigenvalues because the Koopman eigenfunction φ_μ is not in the span of the observables.

Koopman modes

The DMD modes are *right* eigenvectors of A, but the Koopman eigenfunction are related to the *left* eigenvectors of A. How are the DMD modes related to Koopman?

Assume A has a full set of right eigenvectors $\mathbf{v}_j$, with eigenvalues λ_j, and corresponding left eigenvectors $\mathbf{w}_j$, normalized so that $\mathbf{w}_i^* \mathbf{v}_j = \delta_{ij}$. Then any vector $\mathbf{q}$ may be expanded as

$$\mathbf{q} = \sum_{j=1}^{n} (\mathbf{w}_j^* \mathbf{q}) \mathbf{v}_j.$$

In particular, the vector of observables $\boldsymbol{\psi}(\mathbf{z})$ may be expanded this way. Defining $\varphi_j \triangleq \mathbf{w}_j^* \boldsymbol{\psi}$, we then have

$$\boldsymbol{\psi}(\mathbf{z}) = \sum_{j=1}^{n} \varphi_j(\mathbf{z}) \mathbf{v}_j.$$

Koopman modes (continued)

Summary: for $\varphi_j \triangleq \mathbf{w}_j^* \boldsymbol{\psi}$, we have

$$\boldsymbol{\psi}(\mathbf{z}) = \sum_{j=1}^{n} \varphi_j(\mathbf{z}) \mathbf{v}_j.$$

If all of the φ_j in this sum correspond to Koopman eigenfunctions (i.e., if the conditions of the theorem are satisfied), then

$$\boldsymbol{\psi}(\mathbf{F}(\mathbf{z})) = \sum_{j=1}^{n} \lambda_j \varphi_j(\mathbf{z}) \mathbf{v}_j.$$

The terms $\mathbf{v}_j$ in this sum are called Koopman modes. (Note that the Koopman modes depend on the observables $\boldsymbol{\psi}$.)
These $\mathbf{v}_j$ are just the DMD modes, so this gives meaning to DMD modes for a nonlinear system, provided the conditions of the theorem are satisfied.

Outline

Approximating the Koopman operator

We can use spectral methods to approximate the Koopman operator. For the discrete-time dynamical system $\mathbf{x} \mapsto \mathbf{F}(\mathbf{x})$,

$$(\mathcal{K}\varphi)(\mathbf{x}) = \varphi(\mathbf{F}(\mathbf{x})) = (\varphi \circ \mathbf{F})(\mathbf{x}).$$

Then we expand a function φ (and $\mathcal{K}\varphi$) in terms of basis functions ψ_j:

$$\varphi(\mathbf{x}) = \sum_{j=1}^{N} a_j \psi_j(\mathbf{x}), \qquad \mathcal{K}\varphi(\mathbf{x}) = \sum_{j=1}^{N} b_j \psi_j(\mathbf{x})$$

This approximation takes the form of a matrix that maps from $\mathbf{a}$ to $\mathbf{b}$.

Using a weighted residual method, $\mathbf{b} = {\Psi_X}^{+} \Psi_Y \mathbf{a}$, with

$$\Psi_X = \begin{bmatrix} \langle W_1, \psi_1 \rangle & \cdots & \langle W_1, \psi_N \rangle \\ \vdots & & \vdots \\ \langle W_M, \psi_1 \rangle & \cdots & \langle W_M, \psi_N \rangle \end{bmatrix}, \qquad \Psi_Y = \begin{bmatrix} \langle W_1, \psi_1 \circ \mathbf{F} \rangle & \cdots & \langle W_1, \psi_N \circ \mathbf{F} \rangle \\ \vdots & & \vdots \\ \langle W_M, \psi_1 \circ \mathbf{F} \rangle & \cdots & \langle W_M, \psi_N \circ \mathbf{F} \rangle \end{bmatrix}$$

where $\langle W_i, \cdot \rangle$ denotes the inner product with the ith weight function.

A collocation method

Given only data, $\{\mathbf{x}_j, \mathbf{y}_j\}$, with $\mathbf{y}_j = \mathbf{F}(\mathbf{x}_j)$, computing inner products is difficult. However, if we pick $W_i(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_i)$, this gives

$$\Psi_X = \begin{bmatrix} \psi_1(\mathbf{x}_1) & \psi_2(\mathbf{x}_1) & \cdots & \psi_N(\mathbf{x}_1) \\ \psi_1(\mathbf{x}_2) & \psi_2(\mathbf{x}_2) & \cdots & \psi_N(\mathbf{x}_2) \\ \vdots & & & \vdots \\ \psi_1(\mathbf{x}_M) & \psi_2(\mathbf{x}_M) & \cdots & \psi_N(\mathbf{x}_M) \end{bmatrix}, \quad \Psi_Y = \begin{bmatrix} \psi_1(\mathbf{y}_1) & \psi_2(\mathbf{y}_1) & \cdots & \psi_N(\mathbf{y}_1) \\ \psi_1(\mathbf{y}_2) & \psi_2(\mathbf{y}_2) & \cdots & \psi_N(\mathbf{y}_2) \\ \vdots & & & \vdots \\ \psi_1(\mathbf{y}_M) & \psi_2(\mathbf{y}_M) & \cdots & \psi_N(\mathbf{y}_M) \end{bmatrix}$$

The approximation of the Koopman operator is:

$$\mathbf{K} \triangleq \Psi_X{}^{+}\Psi_Y.$$

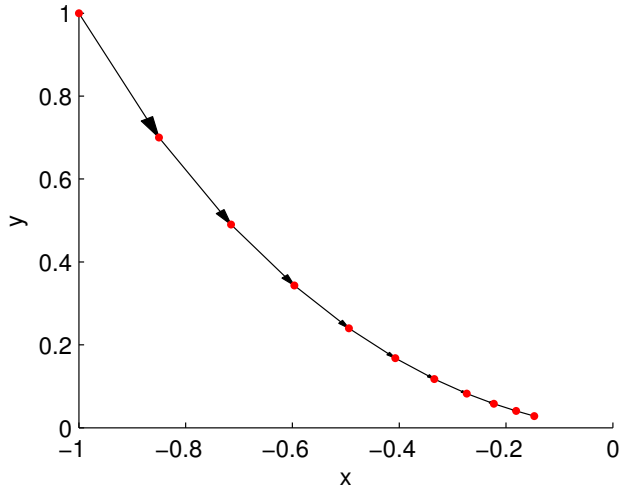
This choice is a *pragmatic one*

- This choice of test functions, W_i, though not optimal, allows the inner products to be computed
- The computationally obtained Koopman modes are the DMD modes if $\psi_i(\mathbf{x}) = \mathbf{e}_i^T\mathbf{x}$, where $\mathbf{e}_i$ is the i-th unit vector.

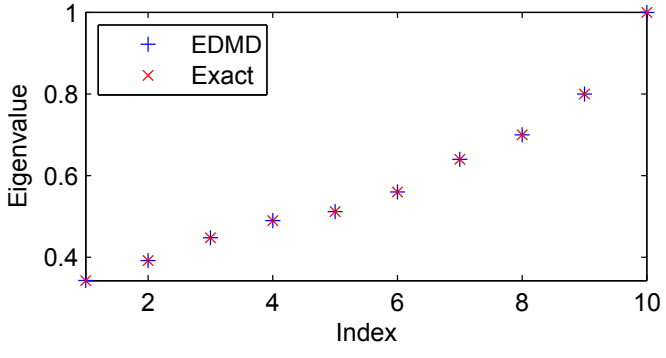
Computing Koopman eigenfunctions: a linear example

- $\vec{x}_{n+1} = \begin{bmatrix} 0.8 & -0.05 \\ 0 & 0.7 \end{bmatrix} \vec{x}_n$, with $\lambda = 0.8$ and 0.7.
- The data are a single time series consisting of 11 snapshots
- Basis functions are: $\psi_{ij}(x, y) = x^i y^j$ with $i, j \in [0, 3]$.

The Data



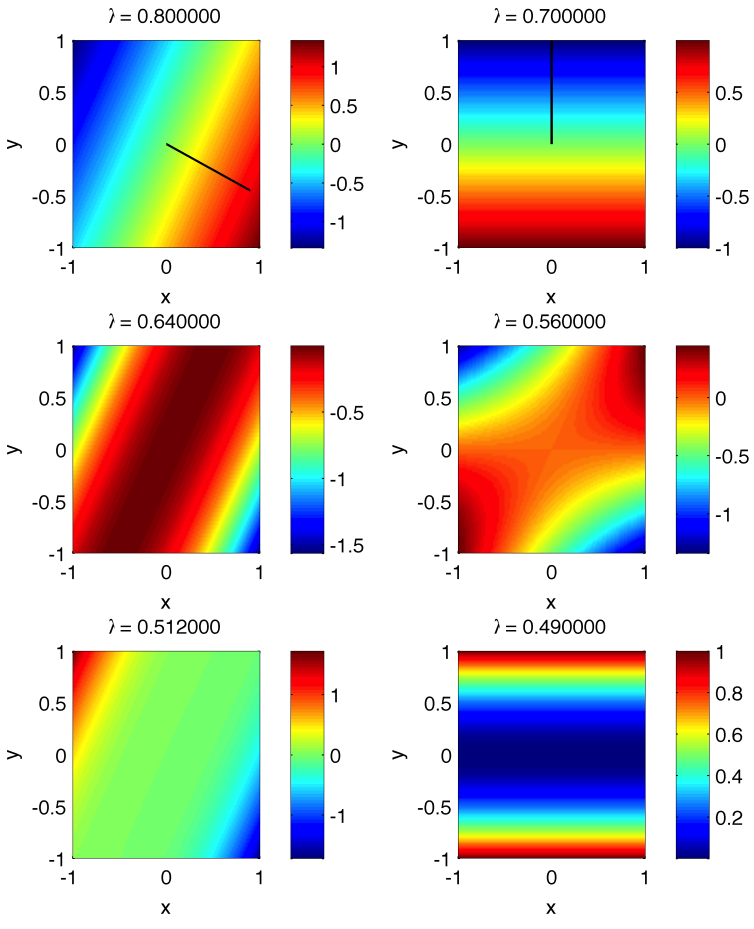
Computed Eigenvalues



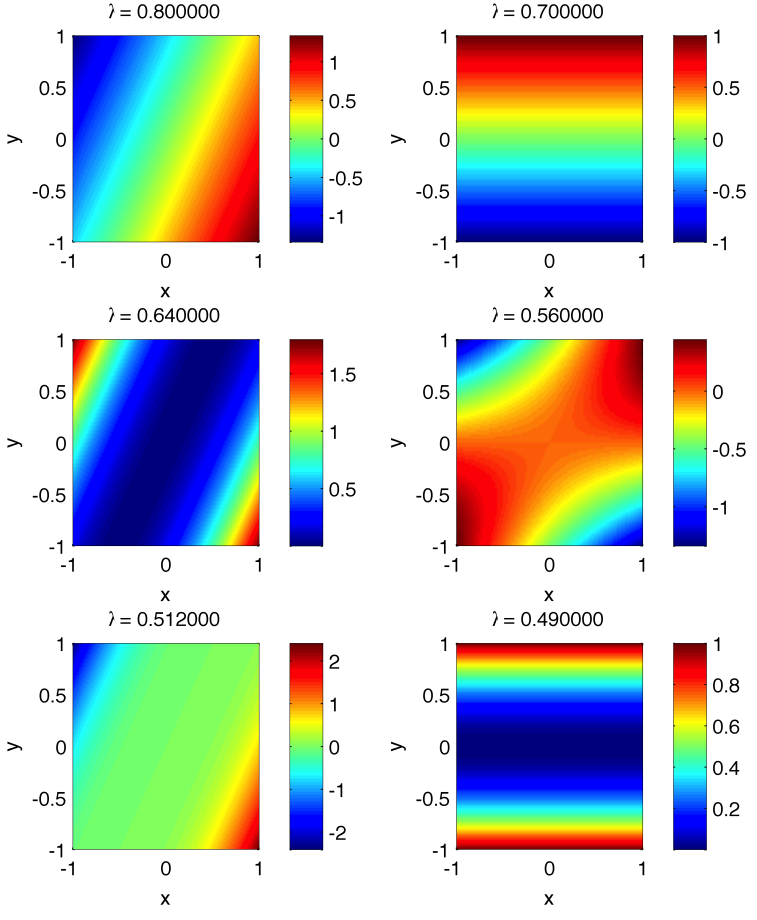
- The true eigenfunctions: $\varphi_{ij}(x, y) \approx (.894x - .447y)^i y^j$ for $i, j \in \mathbb{N}$
- $\lambda_{ij} = (0.8)^i (0.7)^j$
- Excellent agreement between the true and computed eigenvalues

Computing Koopman eigenfunctions: a linear example

EDMD Eigenfunctions



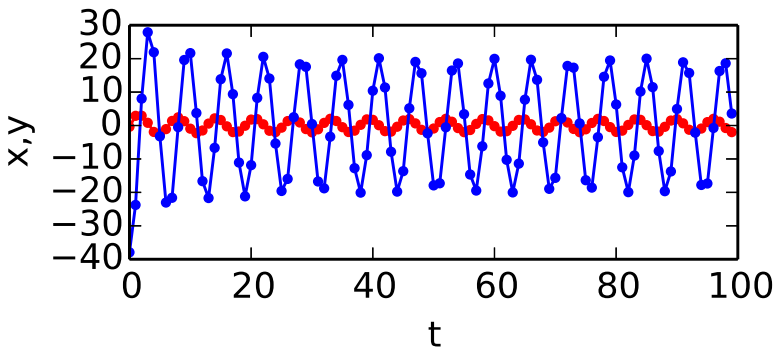
Exact Eigenfunctions



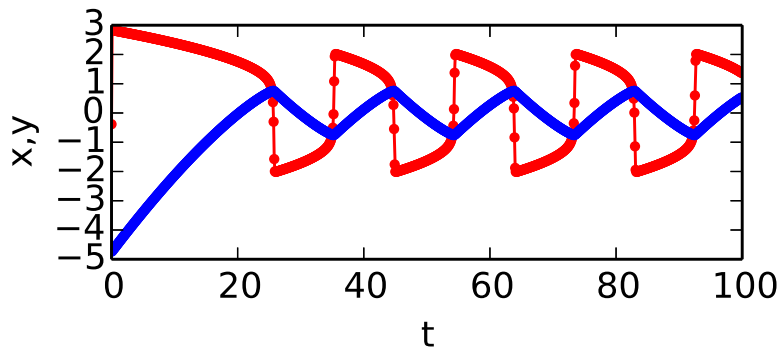
Computing isochrons with EDMD

Isochrons match points in state space to points on a periodic orbit. As it turns out, they are also the level sets of the "phase" of a particular Koopman eigenfunction[4].
Using EDMD, we compute these isochrons for the van der Pol oscillator, which exhibits relaxation oscillations as the parameter ϵ becomes large.

$\epsilon = 0.1$



$\epsilon = 10$



We apply EDMD with:

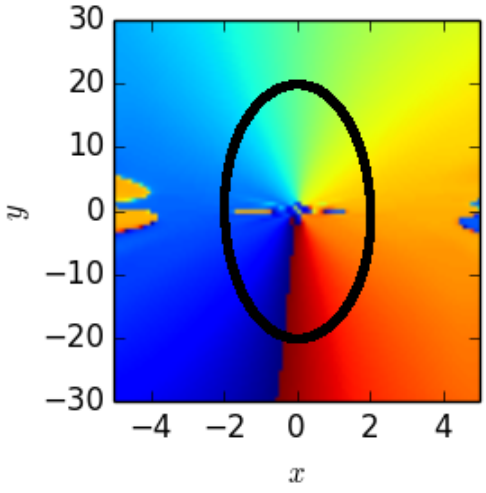
- 250 uniformly distributed trajectories with 10 snapshots each
- Basis functions of the form $\psi_{ij} = r^i \cos(j\theta)$ or $\psi_{ij} = r^i \sin(j\theta)$ for $i = -3, \ldots, 5$ and $j = 0, \ldots, 32$. Here, r, θ are the standard polar coordinates

[4] M. Budisic, R. Mohr, & I. Mezic, "Applied Koopmanism," Chaos **22** 2012.

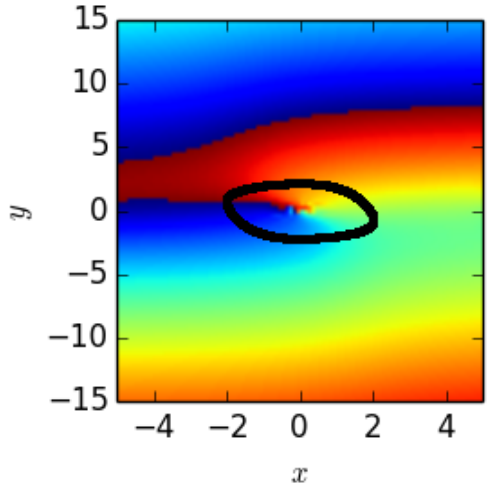
The numerically computed Koopman eigenfunctions

To determine the isochrons, we take $\angle\varphi_\omega$ for the eigenfunction whose frequency matches that of the limit cycle:

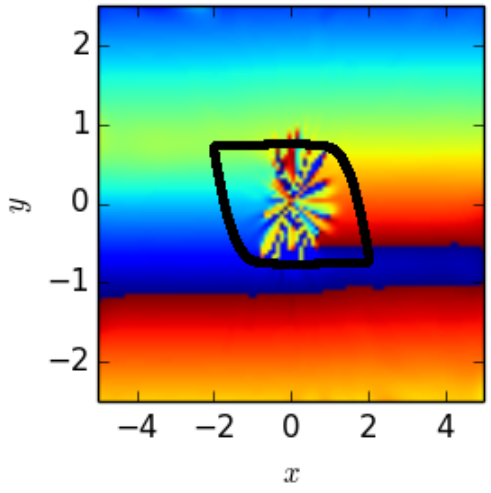
$\epsilon = 0.1$



$\epsilon = 1.0$



$\epsilon = 10$

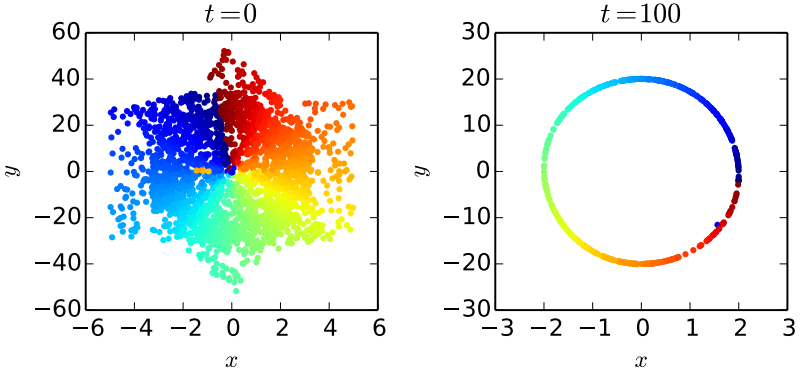


- The prediction of "phase" is ϵ dependent
- Errors are clearly evident near the origin (due to a singularity there) and near the edges (due to lack of data)

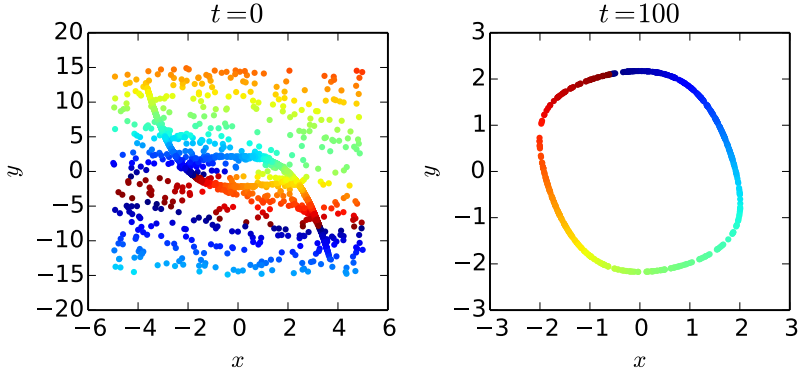
Predicting the phase at long times

To demonstrate the efficacy of the EDMD, we take the original set of data, color it by $\angle\varphi_\omega$, and evolve each particle for 100 times longer than the sampling interval

$\epsilon = 0.1$

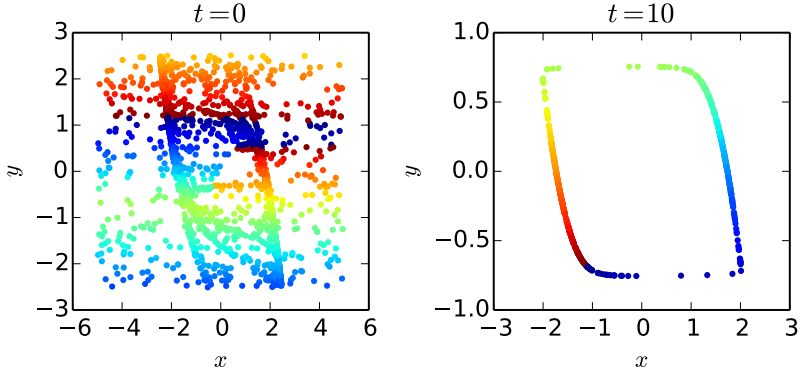


$\epsilon = 1$



- EDMD produces an effective approximation of the needed Koopman eigenfunctions
- EDMD will be inaccurate in regions with few data points

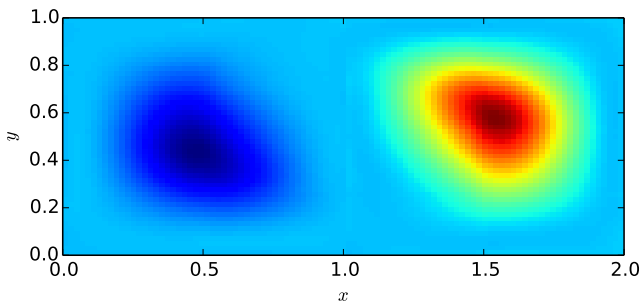
$\epsilon = 10$



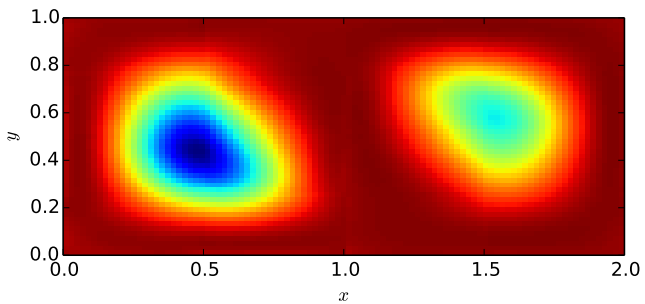
Example: mixing in the double gyre

- We can also use the Koopman operator to study mixing.
- The adjoint of Koopman is called the *Perron-Frobenius* operator; it describes how densities (measures) evolve.
- Can use its eigenfunctions to compute *almost invariant sets*. [5]

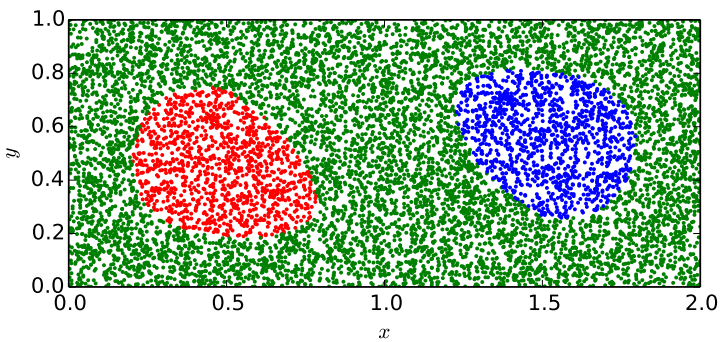
Second eigenfunction



Third eigenfunction



Computed almost invariant sets

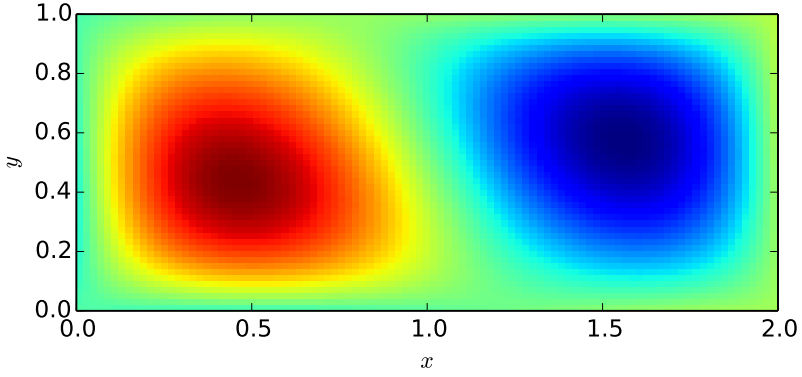


[5] Froyland & Dellnitz, SIAM J. Sci. Comp., 2003; Froyland, Physica D, 2005.

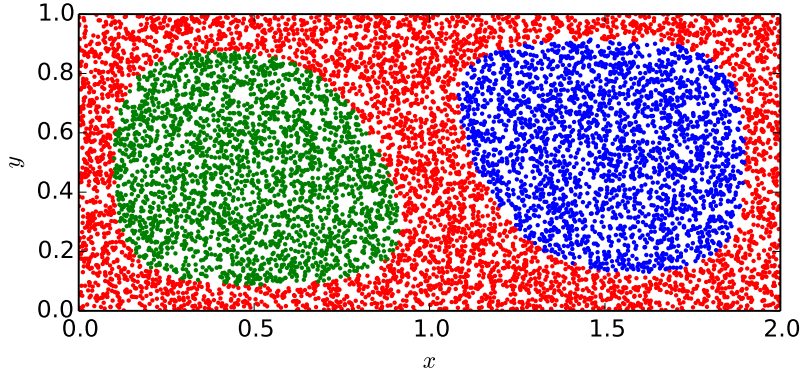
Example: double gyre

With a small amount of data (400 uniformly placed points), EDMD extracts a usable approximation of the almost invariant sets:

Second Eigenfunction



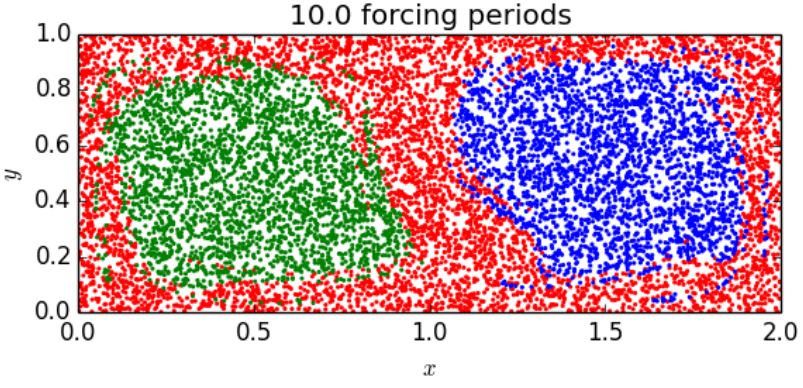
Computed Sets



Three sets identified:

- Use up to 3rd order Legendre polynomials in x and y
- Roughly approximate almost invariant sets

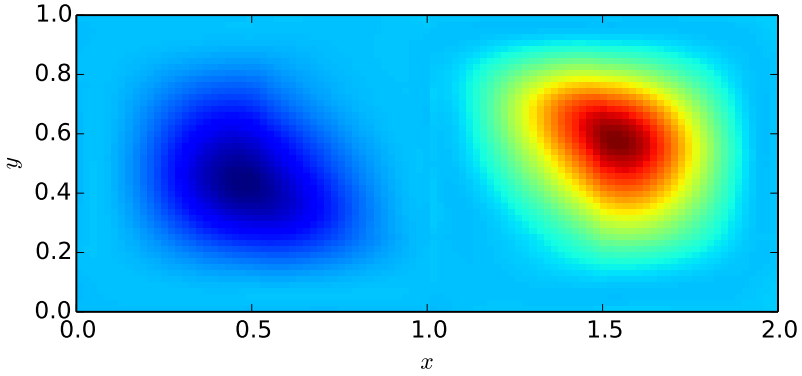
Evolution of Particles



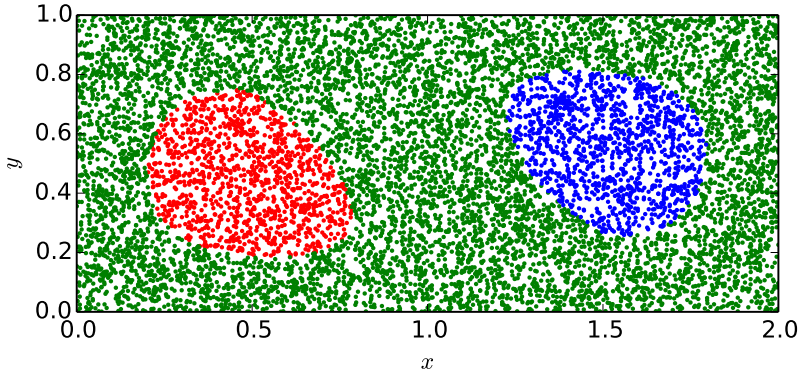
Example: double gyre (continued)

With larger amounts of data (6400 points), EDMD extracts a better approximation:

Second Eigenfunction



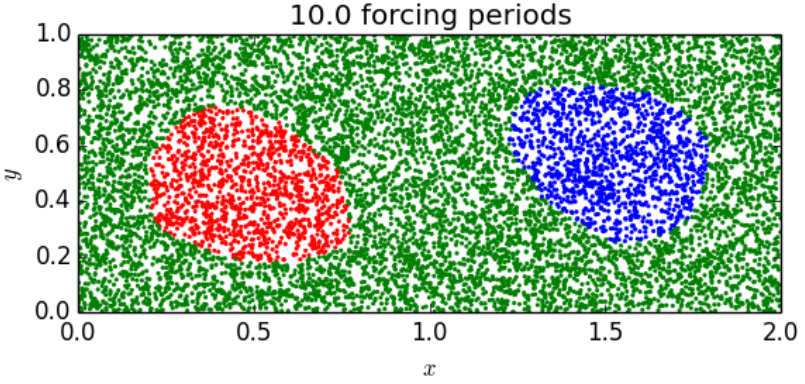
Computed Sets



Three sets identified:

- Use basis of 16 spectral elements (cubic Legendre)
- Fewer "leaked" particles than the previous case
- Not a match with the output of GAIO, but EDMD still creates effective almost invariant sets

Evolution of Particles



Outline

Large state spaces and the "curse of dimensionality"

- We would like to apply Extended DMD to high-dimensional systems (e.g., PDEs such as Navier-Stokes).
- With $N = 400$ states (e.g., gridpoints), if we choose the basis to be all monomials on $\mathbb{R}^{400}$ up to degree 10, this requires $K \approx 10^{19}$ functions; too large for easy computation.
- First trick: we can obtain the same set of EDMD eigenvalues/vectors from the matrix

$$\hat{K} \triangleq G^{+} A,$$

where with M snapshot pairs, $\hat{K}, G, A \in \mathbb{C}^{M \times M}$, with

$$G_{ij} = \sum_{k=1}^{K} \psi_k(\mathbf{x}_i)\bar{\psi}_k(\mathbf{x}_j), \qquad A_{ij} = \sum_{k=1}^{K} \psi_k(\mathbf{y}_i)\bar{\psi}_k(\mathbf{x}_j).$$

This results in $M^2 O(K)$ computations. (Better than K^2 computations of $O(M)$, but still too big.)

The kernel trick

Second trick: the "kernel trick," common in machine learning:

- Write
$$G_{ij} = \langle \psi(\mathbf{x}_i), \psi(\mathbf{x}_j) \rangle, \qquad A_{ij} = \langle \psi(\mathbf{y}_i), \psi(\mathbf{x}_j) \rangle.$$

- Instead of defining the ψ_j explicitly, define a kernel function $f : \mathcal{M} \times \mathcal{M} \to \mathbb{C}$, and let
$$G_{ij} = f(\mathbf{x}_i, \mathbf{x}_j), \qquad A_{ij} = f(\mathbf{y}_i, \mathbf{x}_j),$$
which implicitly determines the ψ_k.
- We use a polynomial kernel $f(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{y}^* \mathbf{x})^\alpha$ with $\alpha = 10$, which results in $M^2 O(N)$ computations. This is much better than $M^2 O(K)$ computations when $N = 400$, $K = 10^{19}$!
- In fact, the cost is identical to regular DMD.

The kernel trick

Why this polynomial kernel $f(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{y}^* \mathbf{x})^\alpha$?

Let $f(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{y}^* \mathbf{x})^2$, with $\mathbf{x}, \mathbf{y} \in \mathbb{R}^2$.

$$\begin{aligned} f(\mathbf{x}, \mathbf{y}) &= (1 + x_1 y_1 + x_2 y_2)^2 \\ &= 1 + 2x_1 y_1 + 2x_2 y_2 + x_1 x_2 y_1 y_2 + x_1^2 y_1^2 + x_2^2 y_2^2 \\ &= \boldsymbol{\psi}(\mathbf{x}) \cdot \boldsymbol{\psi}(\mathbf{y}) \end{aligned}$$

with $\boldsymbol{\psi}(\mathbf{x}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1 x_2, x_1^2, x_2^2)$.

- Note that $\boldsymbol{\psi}(\mathbf{x})$ contains all linear and quadratic monomials in components of $\mathbf{x}$.
- In general, $f(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{y}^* \mathbf{x})^d$ corresponds to the inner product $\boldsymbol{\psi}(\mathbf{x}) \cdot \boldsymbol{\psi}(\mathbf{y})$, where $\boldsymbol{\psi}$ is a vector of all monomials up to degree d.
- Thus, f lets us compute inner products in *feature space* (dimension $K = O(N^d)$) with the computational cost of an inner product in *state space* (dimension N).

Example: the Fitzhugh-Nagumo PDE

The Fitzhugh-Nagumo model is a reaction-diffusion PDE in one spatial dimension:

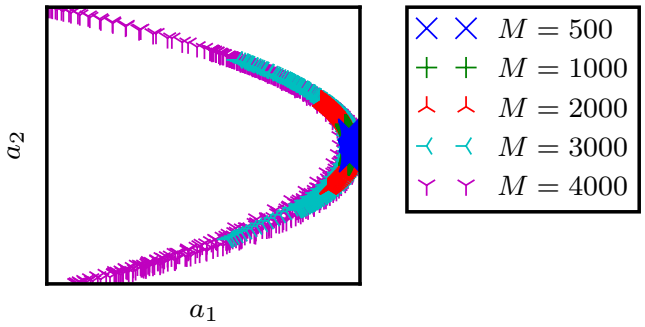
$$\partial_t v = \partial_{xx} v + v - w - v^3$$
$$\partial_t w = \delta \partial_{xx} w + \epsilon(v - c_1 w - c_0),$$

where v is the activation field, w is the inhibition field.

- Parameters chosen so that the model has a spatio-temporal limit cycle: $c_0 = -0.03$, $c_1 = 2.0$, $\delta = 4.0$, $\epsilon = 0.017$, with $x \in [0, 20]$, and Neumann boundary conditions.
- 200 gridpoints for each field, so $N = 400$ states.
- Consider different datasets, with varying "amounts of nonlinearity".

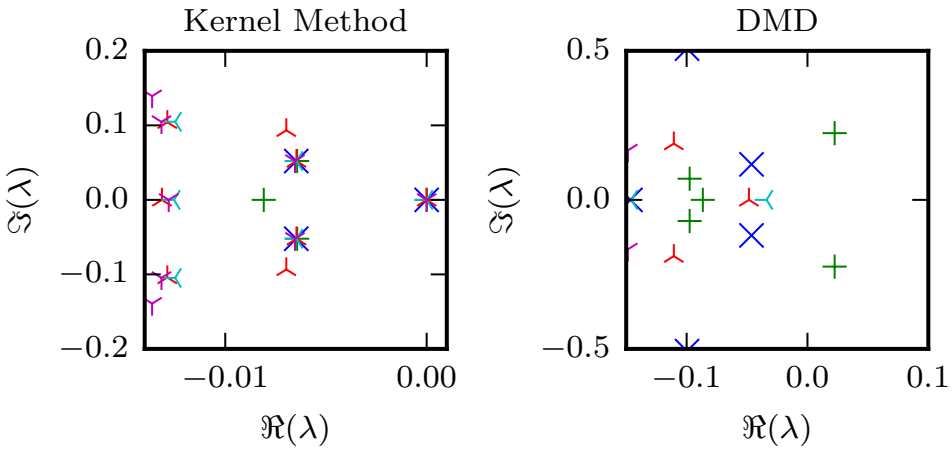
Example: the Fitzhugh-Nagumo PDE

The Data Sets (in $\mathbb{R}^{400}$)

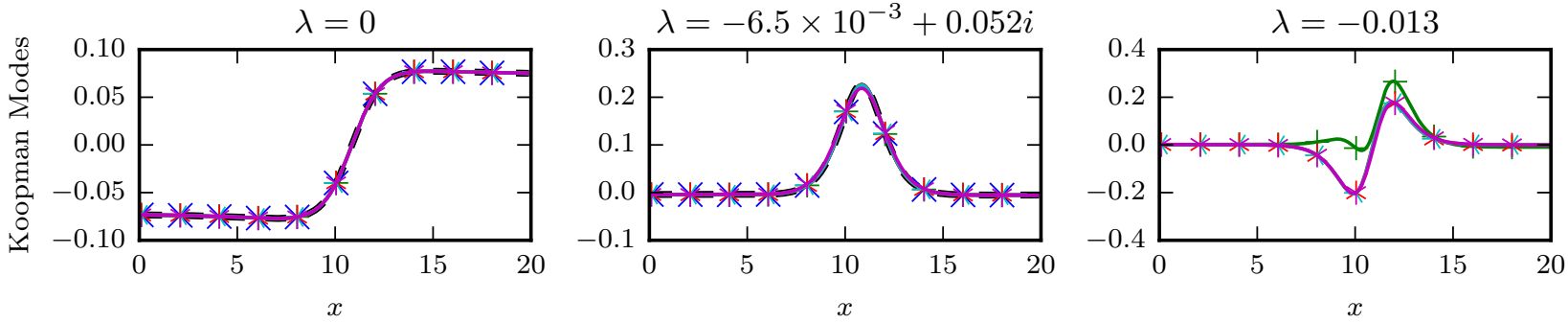


- Shown: the coeffs. of the leading two POD modes

The Computed Spectra



The Computed Koopman Modes



Outline

Nonlinear state reconstruction and data fusion

The Goal: Given two different sets of measurements of the same physical phenomenon, identify a mapping from one set of measurements to the other that requires a minimal number of "joint" measurements.

Other Algorithms: This is a common task, which could be accomplished using:

- Linear (and quadratic) stochastic estimation
- Kalman filters

However, they require many joint measurements to determine a mapping

Our Approach requires (in principle) a single joint measurement, and relies on three properties of the Koopman eigenvalues and eigenfunctions:

1. The *Koopman eigenvalues* are invariant to invertible transformations of the dynamical system
2. The value of a *Koopman eigenfunction* is "invariant" to different representations of the state (i.e., equivalent states produce the same values)
3. The Koopman eigenfunctions parameterize state space

DMD for data fusion

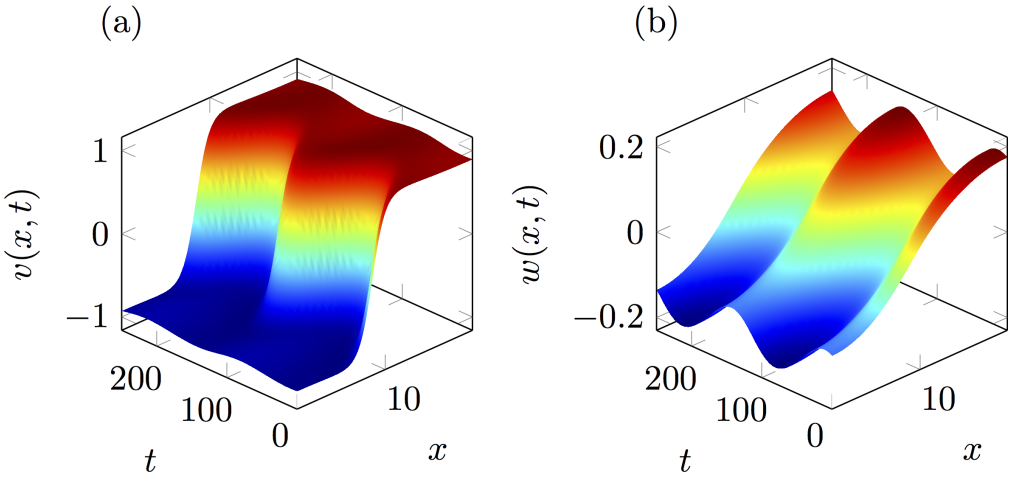
Our data fusion procedure requires:

1. Two sets of different measurement data that must capture the same underlying dynamics, but could be obtained independently
2. A small set of *registration data* that contains two different measurements of the same underlying state

For example, one set of measurements could be PIV measurements; another could be an array of pressure sensors along the wall.

Fitzhugh-Nagumo PDE

We generated this data using the FitzHugh-Nagumo PDE:



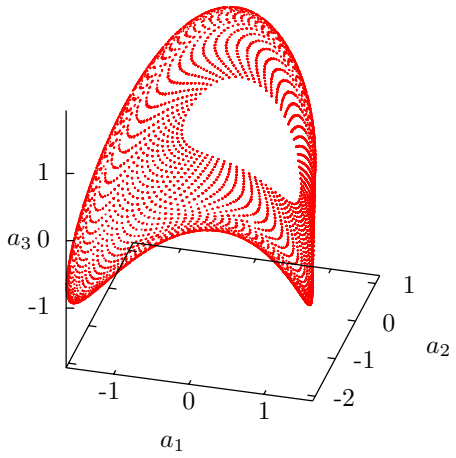
- 4×10^4 total data points generated by perturbing the unstable fixed point, and recording the approach to the limit cycle
- This procedure results in 40 randomly initialized trajectories of 10^3 points

Gathering the datasets

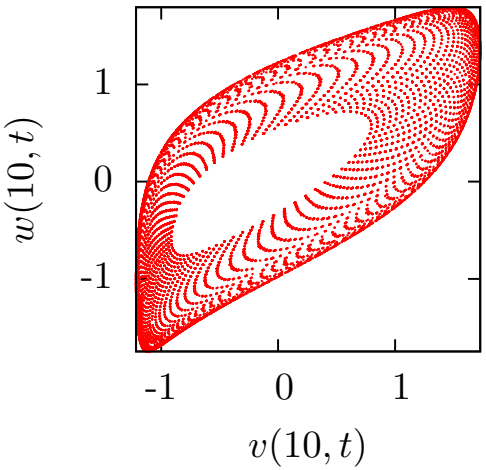
To obtain the data:

1. PCA was performed on the first 20 trajectories, and the coefficients recorded.
2. The values of v and w at $x = 10$ were recorded for the second 20 trajectories.
3. One randomly selected point was used as the registration set.

PCA Data



Pointwise Data

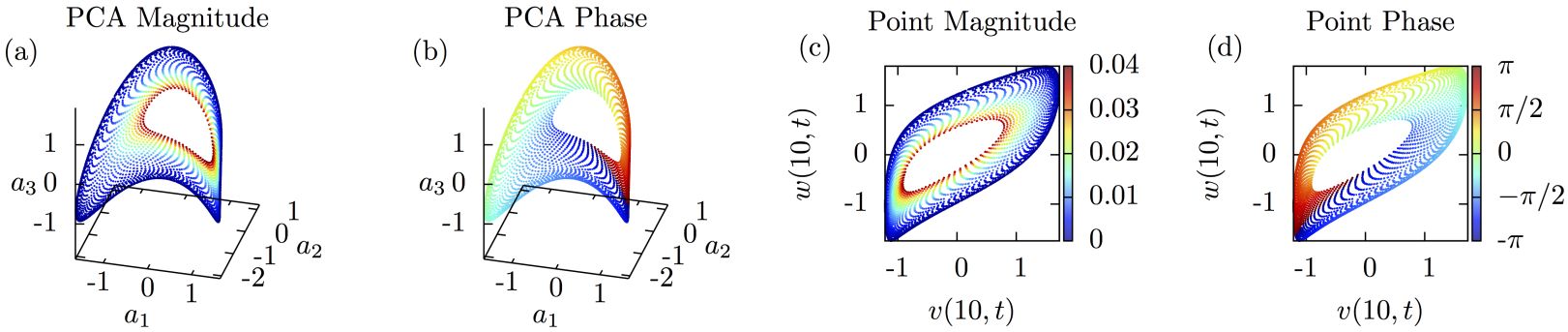


Combined Data

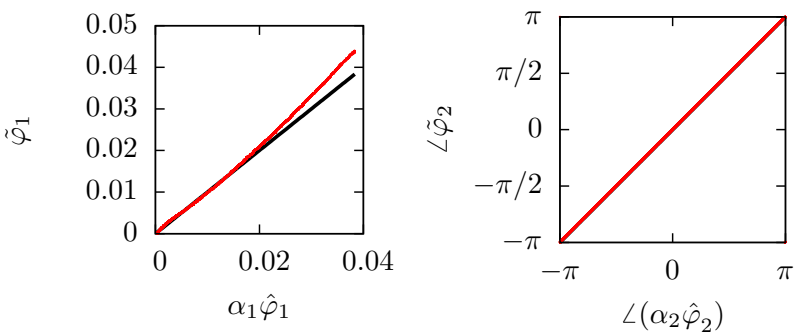
$$a_1 = -1.52$$
$$a_2 = -1.60$$
$$a_3 = -1.02$$
$$v(10) = 1.65$$
$$w(10) = 0.85$$

Koopman eigenfunctions

- Compute the Koopman eigenfunctions and eigenvalues using the first two data sets and EDMD:



- Note: these eigenfunctions are "matched" using their eigenvalues
- Use the third data set to determine the missing constant α_k, which is required because eigenfunctions are defined only up to a constant.

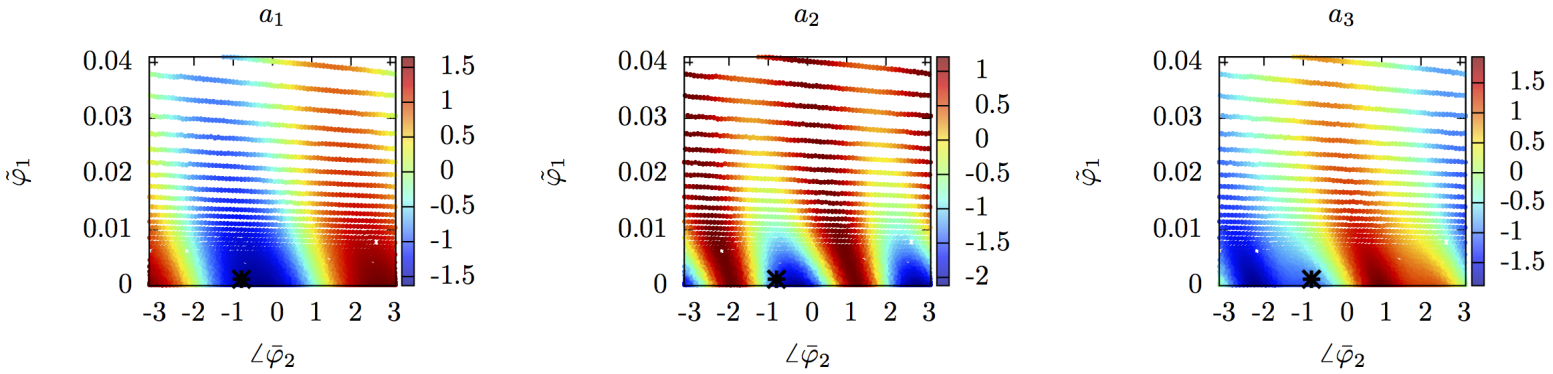


- Good agreement until $\varphi_1 \approx 0.02$, and everywhere for φ_2
- Intuition: EDMD will do well near the limit cycle

State reconstruction

The final step: mapping from the eigenfunctions to states

- Because we have a large amount of data for a single set of measurements, we use simple interpolation



- Note: the Koopman modes offer an alternative means of state reconstruction, but will require more than two eigenfunctions to be approximated

Therefore, the procedure for a new point is:

1. Compute the values of $\hat{\varphi}_1(\mathbf{x}_{\text{new}})$ and $\hat{\varphi}_2(\mathbf{x}_{\text{new}})$ using the EDMD approximation of the Koopman eigenfunctions
2. Multiply by α_1 and α_2 to approximate $\tilde{\varphi}_1$ and $\tilde{\varphi}_2$
3. Interpolate to obtain a_1, a_2, and a_3

Conclusions

- DMD can be used for nonlinear systems (via Koopman), but only when used correctly: one needs to choose a sufficiently rich "set of observables" to span the Koopman eigenspace you are trying to capture.
- Extended DMD explicitly approximates Koopman: one chooses a basis for a subspace of function space, and determines a finite-dimensional approximation of Koopman with respect to that basis. The approximation can be determined directly from data (snapshots).
- Kernel DMD makes this practical for high-dimensional systems: one uses a potentially enormous basis, with the same computational cost as regular DMD. This gives vast improvement for problems with nonlinearity.
- Applications
  - identifying coherent structures based on dynamics
  - quantifying transport/mixing
  - "fusing" data from different sensors

References and acknowledgements

- Supported by NSF and AFOSR

- References

  - JH Tu, CW Rowley, DM Luchtenburg, SL Brunton, and JN Kutz. On dynamic mode decomposition: theory and applications. *J. Computational Dynamics*, in press. arxiv:1312.0041
  - MO Williams, IG Kevrekidis, and CW Rowley. Approximating the Koopman operator using data: extending dynamic mode decomposition. arXiv:1408.4408
  - MO Williams, CW Rowley, and IG Kevrekidis. A kernel approach to data-driven Koopman spectral analysis. arXiv:1411.2260
  - MO Williams, CW Rowley, I Mezic, and IG Kevrekidis. Data fusion via intrinsic dynamic variables: an application of data-driven Koopman spectral analysis. (will appear soon on arXiv)