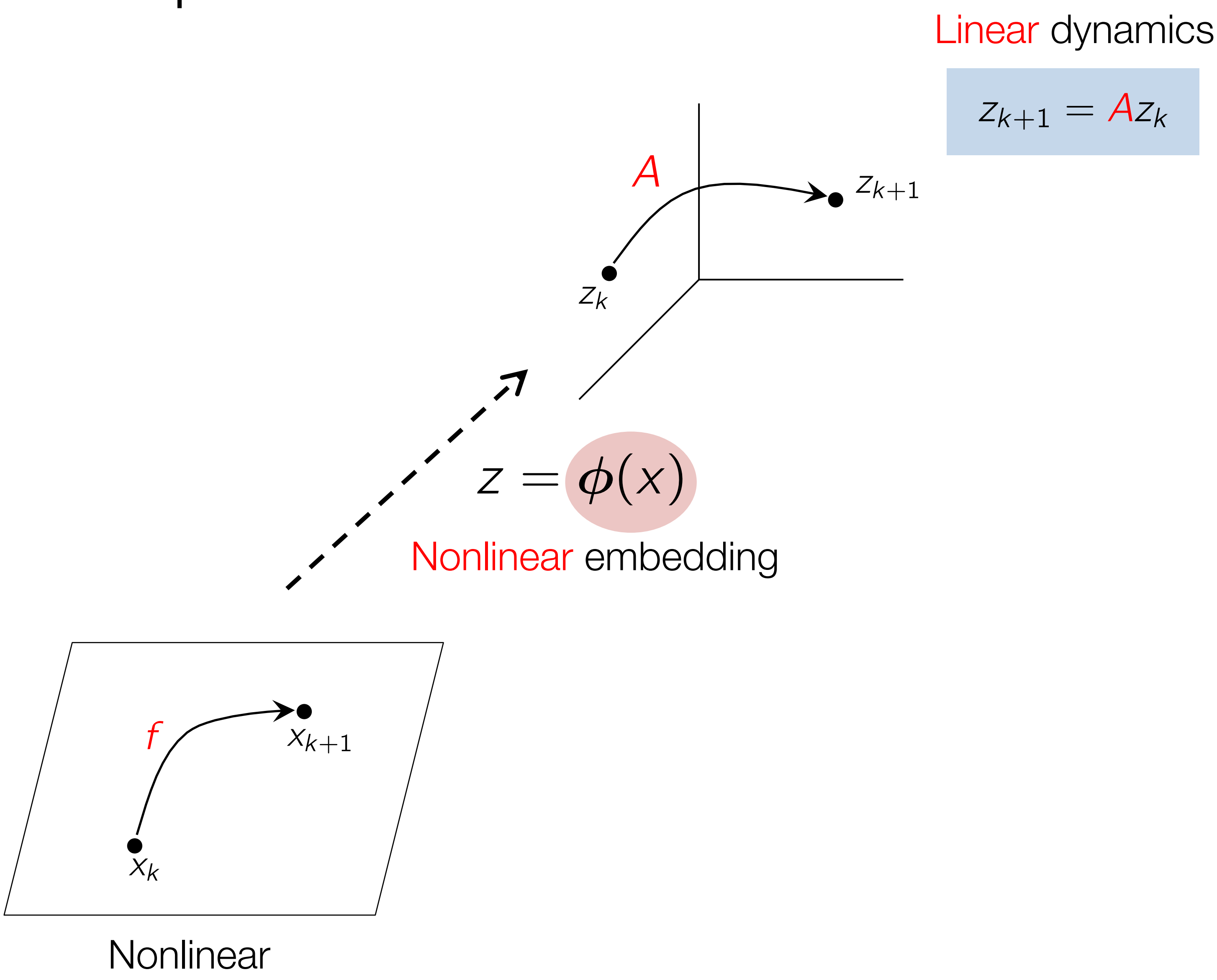
Learning Koopman eigenfunctions for prediction and control

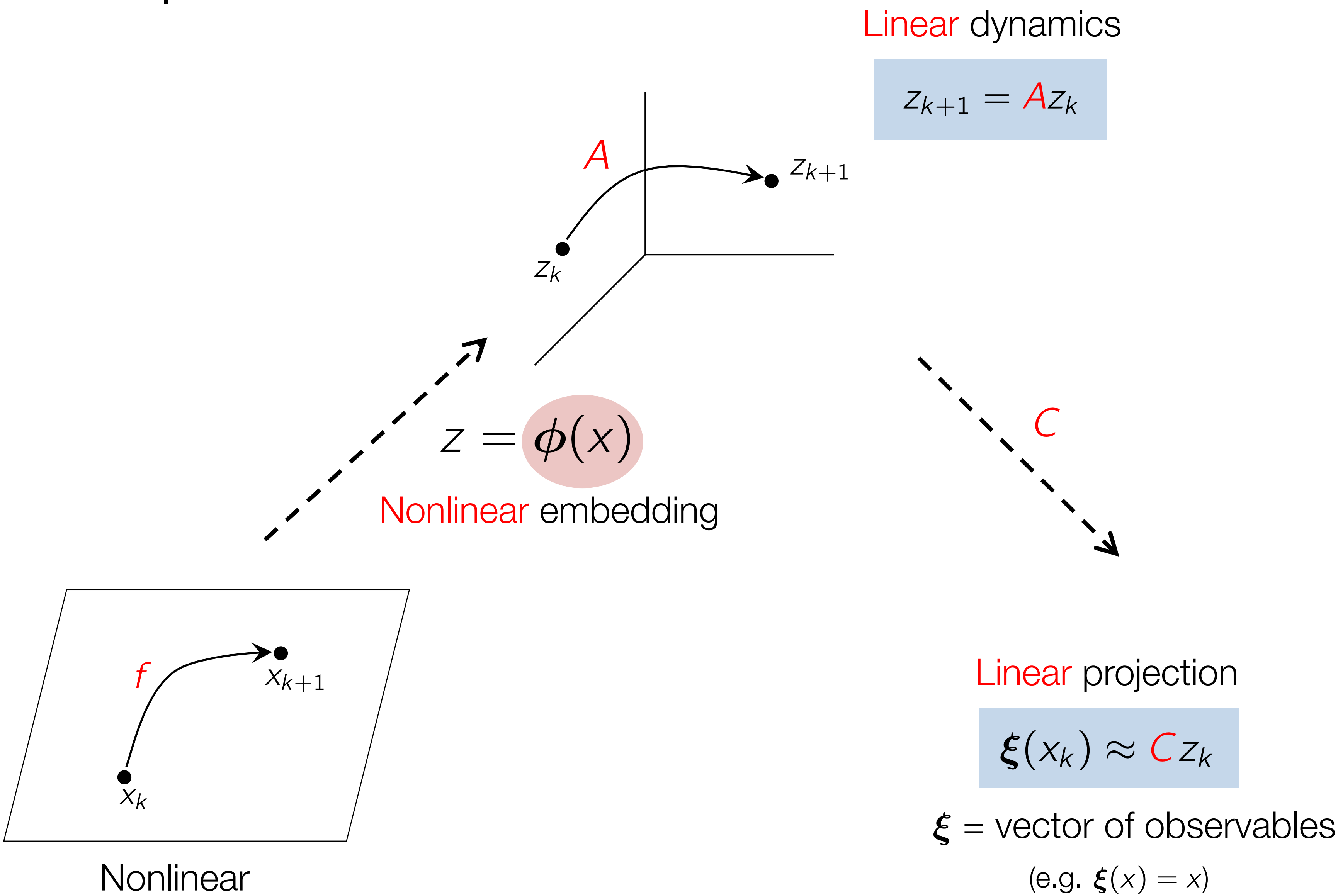
Milan Korda
(LAAS-CNRS)

Igor Mezić
(University of California, Santa Barbara)

Linear prediction



Linear prediction



Why linear predictors?

$$
\begin{aligned}
z_{k+1} &= A z_k \\
z_0 &= \phi(x_0) \\
\hat{y}_k &= C z_k
\end{aligned}
$$

$$\hat{y}_k \approx \boldsymbol{\xi}(x_k)$$

Why linear predictors?

$$
\begin{aligned}
z_{k+1} &= Az_k \\
z_0 &= \phi(x_0) \\
\hat{y}_k &= Cz_k
\end{aligned}
$$

$$\hat{y}_k \approx \xi(x_k)$$

Nonlinear feedback control & estimation using linear techniques

$\Rightarrow$ Model predictive control *[Korda & Mezic, 2018]*

$\Rightarrow$ State estimation *[Surana & Banaszuk, 2016]*

Mature & well understood

Fast computation (linear algebra / convex optimization)

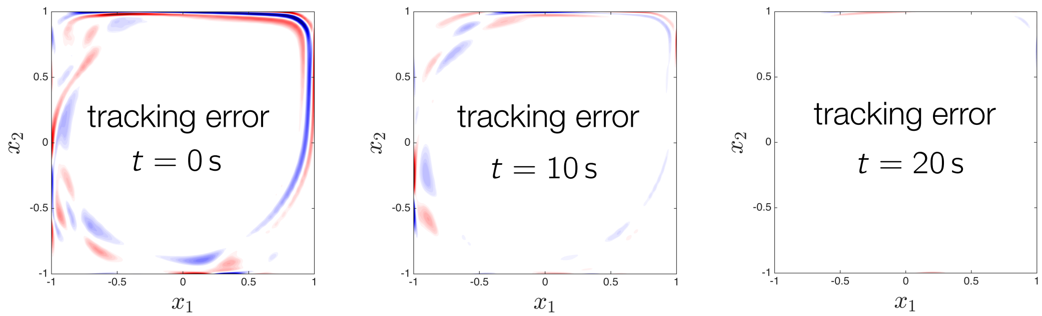
Rapid deployment in applications

Koopman MPC - applications

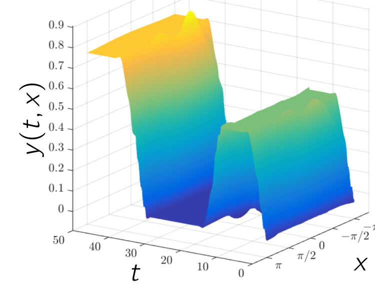
Fluid dynamics

[Arbabi et al. 2018]

Cavity flow control

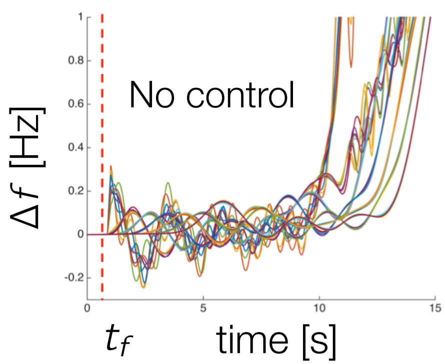


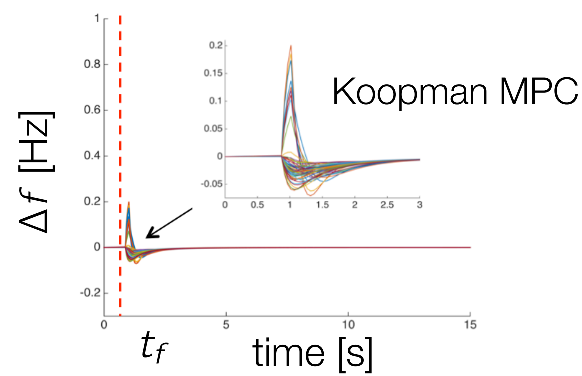
Kortweg-de Vries



Powergrid

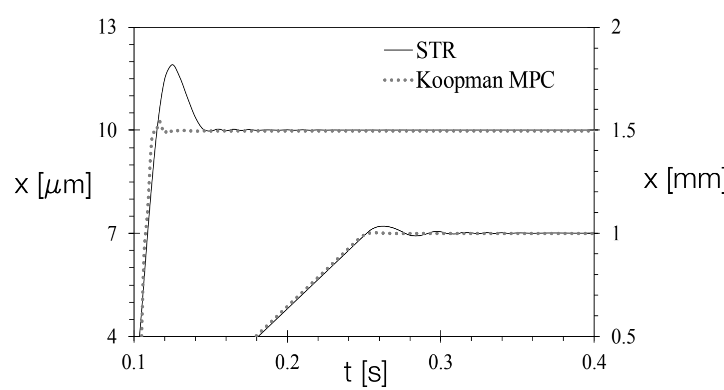
[Korda et al. 2017]





High-precision positioning

[Kamenar et al. 2018]



Choosing the embedding

$$z_{k+1} = A z_k$$
$$z_0 = \phi(x_0)$$
$$\hat{y}_k = C z_k$$

When can we predict exactly?

$$\hat{y}_k = \xi(x_k)$$

Choosing the embedding

$$
\begin{aligned}
z_{k+1} &= A z_k \\
z_0 &= \phi(x_0) \\
\hat{y}_k &= C z_k
\end{aligned}
$$

$$\hat{y}_k = \boldsymbol{\xi}(x_k)$$

if

$\mathrm{span}\{\phi_1, \ldots, \phi_N\}$ is Koopman invariant & $\boldsymbol{\xi} \in \mathrm{span}\{\phi_1, \ldots, \phi_N\}$

$\Longleftrightarrow$ $\Longleftrightarrow$

ϕ_i's are (generalized) Koopman eigenfunctions

(or linear combinations thereof)

Span of ϕ_i's is rich enough

Eigenfunction construction

Eigenfunction construction

$$\dot{x} = f(x)$$

Eigenfunction

$$\phi(S_t(x)) = e^{\lambda t}\phi(x)$$

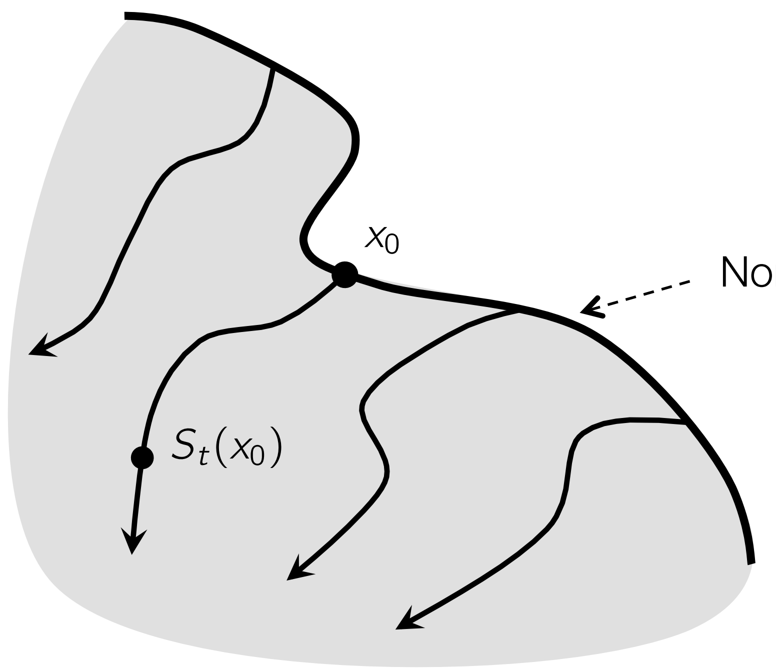
Eigenfunction construction

$$\dot{x} = f(x)$$

Eigenfunction

$$\phi(S_t(x)) = e^{\lambda t}\phi(x)$$

Key observation: Non-recurrent surface ⇒ uncountably many eigenfunctions



Eigenfunction construction

$$\dot{x} = f(x)$$

Eigenfunction

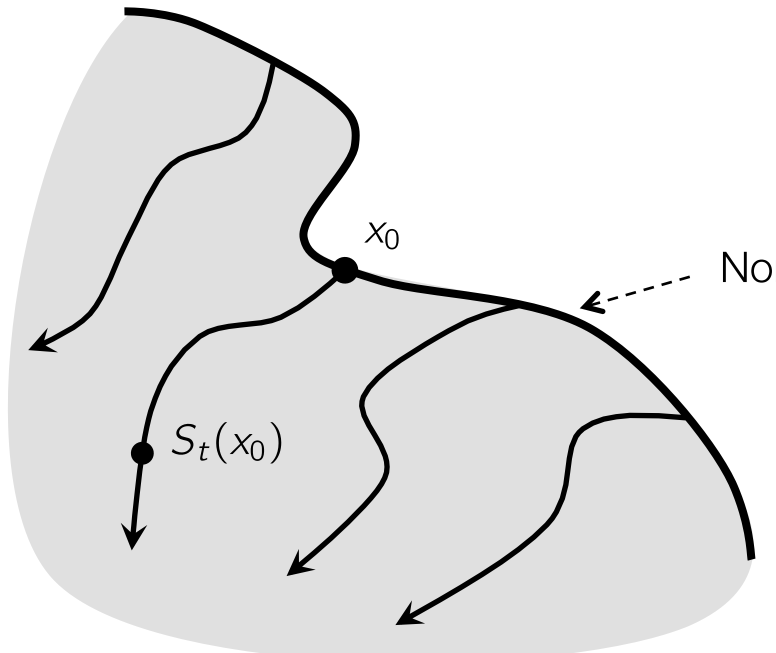
$$\phi(S_t(x)) = e^{\lambda t}\phi(x)$$

Key observation: Non-recurrent surface $\Rightarrow$ uncountably many eigenfunctions

$g =$ arbitrary continuous function
$\lambda =$ arbitrary complex number

$$\phi_{\lambda,g}(S_t(x_0)) = e^{\lambda t} g(x_0) \quad x_0 \in \Gamma$$

$\phi_{\lambda,g} = g$ on Γ



Eigenfunction construction

$$\dot{x} = f(x)$$

Eigenfunction

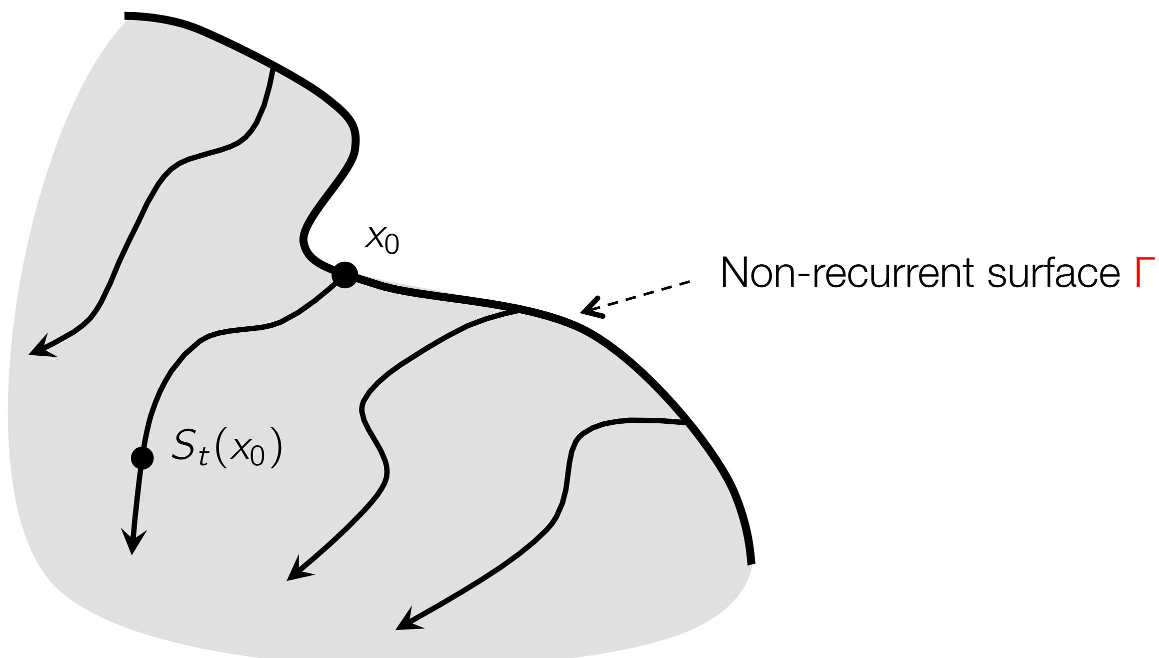
$$\phi(S_t(x)) = e^{\lambda t}\phi(x)$$

Key observation: Non-recurrent surface $\Rightarrow$ uncountably many eigenfunctions

g = arbitrary continuous function
λ = arbitrary complex number

$$\phi_{\lambda,g}(S_t(x_0)) = e^{\lambda t} g(x_0) \quad x_0 \in \Gamma$$

$$\phi_{\lambda,g} = g \quad \text{on} \quad \Gamma$$



Lemma: Γ non-recurrent & g continuous $\Rightarrow$ $\phi_{\lambda,g}$ is a continuous eigenfunction

Eigenfunction construction

$$\dot{x} = f(x)$$

Eigenfunction

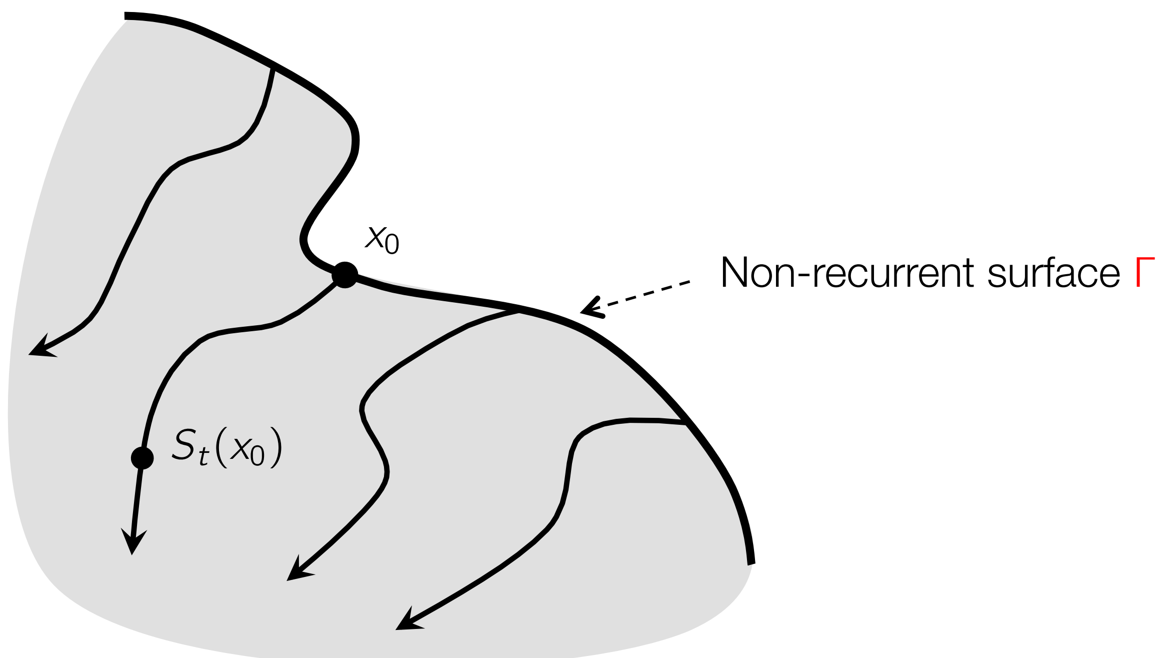
$$\phi(S_t(x)) = e^{\lambda t}\phi(x)$$

Key observation: Non-recurrent surface ⇒ uncountably many eigenfunctions

g = arbitrary continuous function
λ = arbitrary complex number

$$\phi_{\lambda,g}(S_t(x_0)) = e^{\lambda t} g(x_0) \quad x_0 \in \Gamma$$

$\phi_{\lambda,g} = g$ on Γ



cf. **Open eigenfunctions** *[Mezic 2017]*

Richness

Key question: how rich is the class of eigenfunctions obtained in this way?

Richness

Key question: how rich is the class of eigenfunctions obtained in this way?

Λ = subset of complex numbers

G = subset of continuous functions

$\Phi_{\Lambda,G}$ = all eigenfunctions associated to $(\lambda, g) \in (\Lambda, G)$

Richness

Key question: how rich is the class of eigenfunctions obtained in this way?

$\Lambda =$ subset of complex numbers

$G =$ subset of continuous functions

$\Phi_{\Lambda,G} =$ all eigenfunctions associated to $(\lambda, g) \in (\Lambda, G)$

$\Lambda \supset \text{lattice}(\Lambda_0)$

$G = \{g_i\}_{i=1}^{\infty}$ with $\text{span}\{G\}$ dense in $\mathcal{C}(\Gamma)$

Theorem: Γ non-recurrent, $\Lambda_0 = \bar{\Lambda}_0$ & $\exists \lambda \in \Lambda_0$ with $\text{Re}(\lambda) \neq 0$

$$\Rightarrow \text{span}\{\Phi_{\Lambda,G}\} \text{ dense in } \mathcal{C}(X_T)$$

Richness

Key question: how rich is the class of eigenfunctions obtained in this way?

$\Lambda =$ subset of complex numbers

$G =$ subset of continuous functions

$\Phi_{\Lambda,G} =$ all eigenfunctions associated to $(\lambda, g) \in (\Lambda, G)$

$\Lambda \supset \text{lattice}(\Lambda_0)$

$G = \{g_i\}_{i=1}^{\infty}$ with $\text{span}\{G\}$ dense in $\mathcal{C}(\Gamma)$

Theorem: Γ non-recurrent, $\Lambda_0 = \bar{\Lambda}_0$ & $\exists \lambda \in \Lambda_0$ with $\text{Re}(\lambda) \neq 0$

$\Rightarrow \text{span}\{\Phi_{\Lambda,G}\}$ dense in $\mathcal{C}(X_T)$

For every continuous function ξ and every $\epsilon > 0$ there exists $\phi_1, \ldots, \phi_N \in \Phi_{\Lambda,G}$ such that

$$\sup_x \left| \xi(x) - \sum_{i=1}^{N} c_i \phi_i(x) \right| < \epsilon$$

for some coefficients $c_1, \ldots, c_N$

Richness

Key question: how rich is the class of eigenfunctions obtained in this way?

$\Lambda =$ subset of complex numbers

$G =$ subset of continuous functions

$\Phi_{\Lambda,G} =$ all eigenfunctions associated to $(\lambda, g) \in (\Lambda, G)$

$\Lambda \supset \text{lattice}(\Lambda_0)$

$G = \{g_i\}_{i=1}^{\infty}$ with $\text{span}\{G\}$ dense in $\mathcal{C}(\Gamma)$

Theorem: Γ non-recurrent, $\Lambda_0 = \bar{\Lambda}_0$ & $\exists \lambda \in \Lambda_0$ with $\text{Re}(\lambda) \neq 0$

$$\Rightarrow \text{span}\{\Phi_{\Lambda,G}\} \text{ dense in } \mathcal{C}(X_T)$$

Proof: Apply Stone-Weirstrass to $\text{span}\{\Phi_{\Lambda,G}\}$

Thanks Corbi!

Richness

Key question: how rich is the class of eigenfunctions obtained in this way?

$\Lambda =$ subset of complex numbers

$G =$ subset of continuous functions

$\Phi_{\Lambda,G} =$ all eigenfunctions associated to $(\lambda, g) \in (\Lambda, G)$

$\Lambda \supset \text{lattice}(\Lambda_0)$

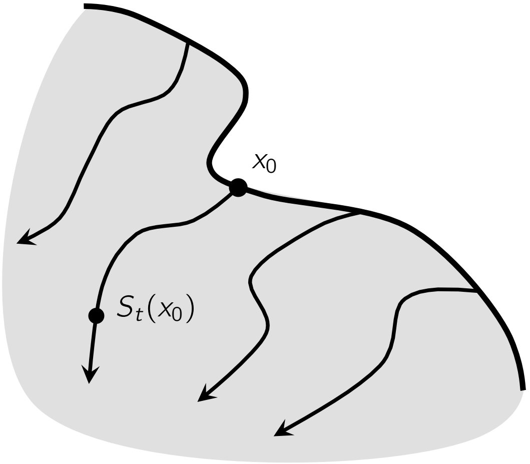
$G = \{g_i\}_{i=1}^{\infty}$ with $\text{span}\{G\}$ dense in $\mathcal{C}(\Gamma)$

Theorem: Γ non-recurrent, $\Lambda_0 = \bar{\Lambda}_0$ & $\exists \lambda \in \Lambda_0$ with $\text{Re}(\lambda) \neq 0$

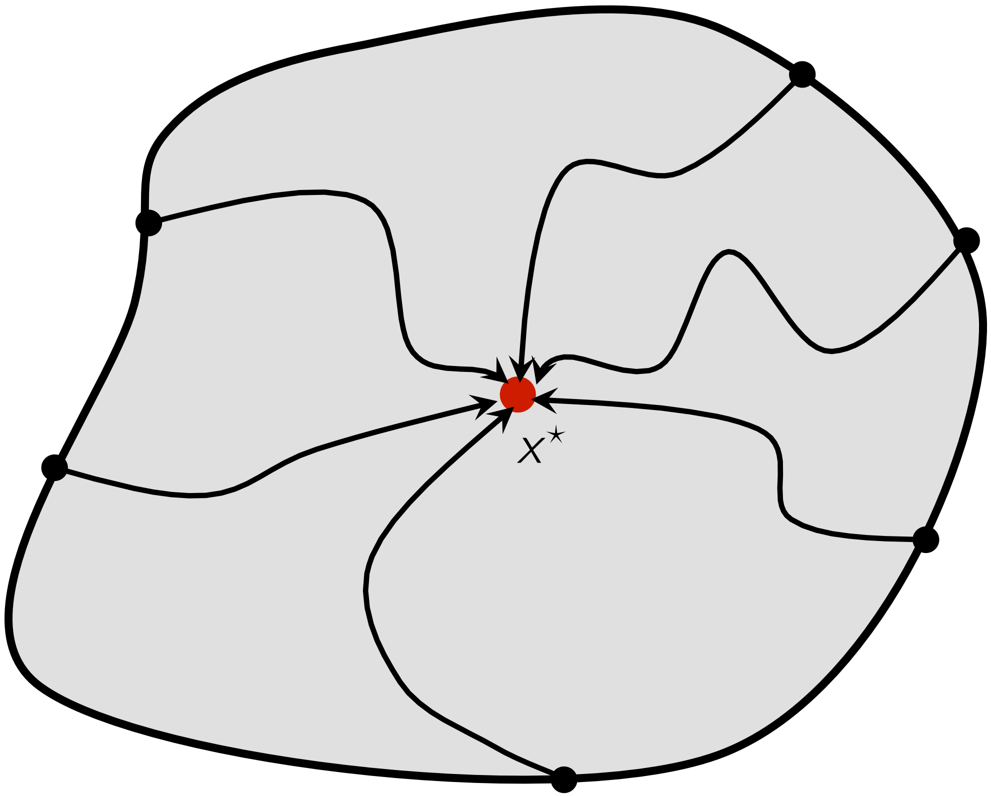
$\Rightarrow \text{span}\{\Phi_{\Lambda,G}\}$ dense in $\mathcal{C}(X_T)$

Proof: Apply Stone-Weirstrass to $\text{span}\{\Phi_{\Lambda,G}\}$

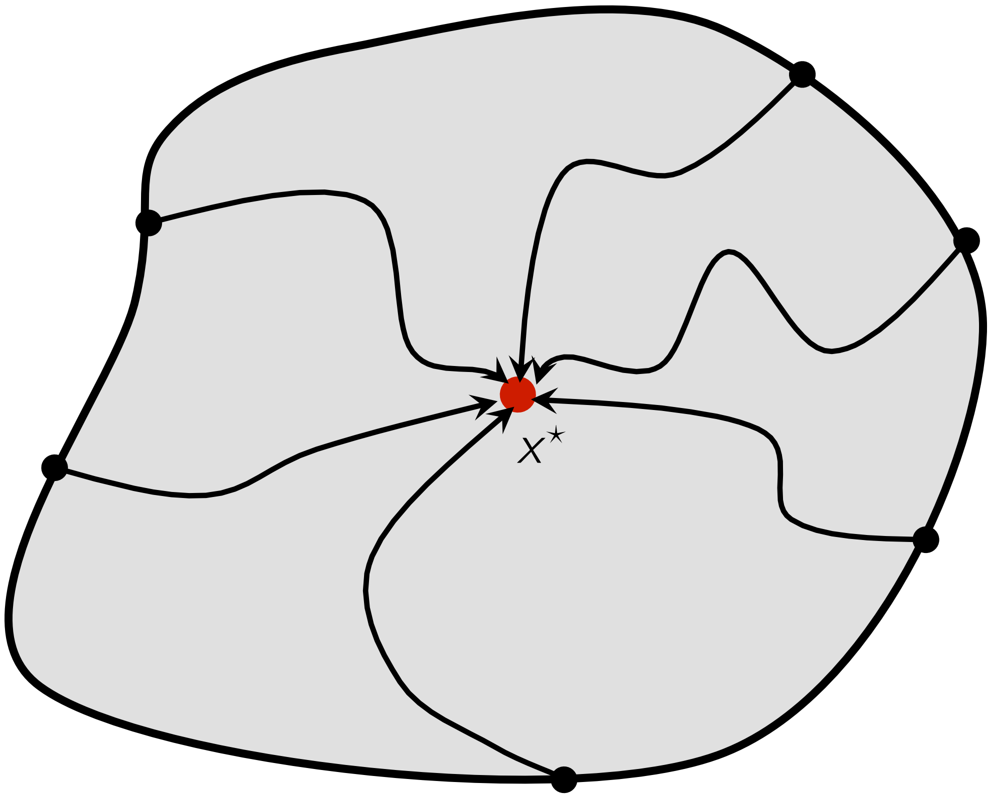
Thanks Corbi!



Singularities don't matter in $\mathcal{C}$



But they do in higher regularity spaces



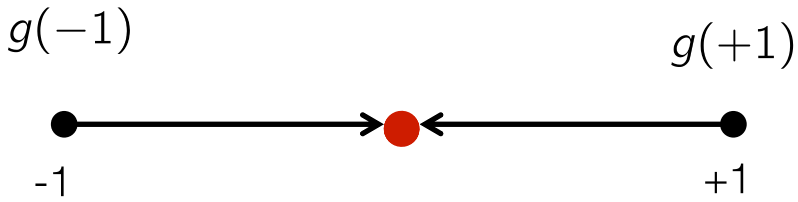
[Mezić 2017]

Stable equilibrium $x^\star$ + conditions on eigenvalues of $Df(x^\star) \Rightarrow$ real analytic eigenfunctions dense in $\mathcal{C}$ exist

M. Kvalheim's poster: regular eigenfunctions $\Rightarrow$ principal eigenvalues

Example

$$\dot{x} = ax\,, \quad a < 0$$



$$\Gamma = \{-1, 1\}$$

$$\phi_{\lambda,g}(x) = \begin{cases} g(-1)|x|^{\frac{\lambda}{a}} & x < 0 \\ g(+1)|x|^{\frac{\lambda}{a}} & x > 0 \end{cases}$$

Observations:

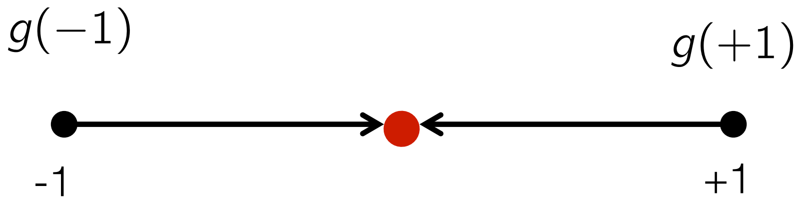
Continuous on $[-1, 1]$ if $\lambda < 0$ (and on $[-1, 1] \setminus \{0\}$ for any λ)

Analytic if $\lambda = k \cdot a$, $k \in \mathbb{N}$ and $g(-1) = (-1)^k g(+1)$

$\mathrm{span}\{|x|^{\frac{k\lambda}{a}} \,:\, k \in \mathbb{N}\}$ dense in $\mathcal{C}([0, 1])$ if $\lambda < 0$

Example

$$\dot{x} = ax\,, \quad a < 0$$



$$\Gamma = \{-1, 1\}$$

$$\phi_{\lambda,g}(x) = \begin{cases} g(-1)|x|^{\frac{\lambda}{a}} & x < 0 \\ g(+1)|x|^{\frac{\lambda}{a}} & x > 0 \end{cases}$$

Optimal choice of λ and g depends on $\boldsymbol{\xi}$

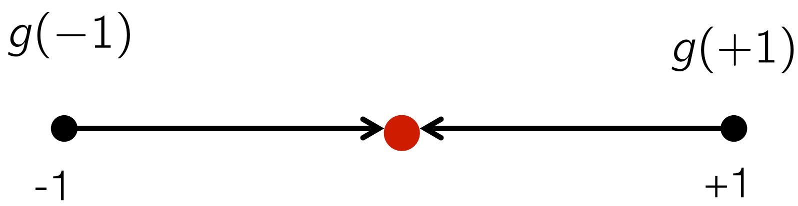
$$\boldsymbol{\xi} = x \quad \Rightarrow \quad \lambda = a \ \text{ and } \ g(+1) = 1,\, g(-1) = -1$$

$$\boldsymbol{\xi} = |x| \quad \Rightarrow \quad \lambda = a \ \text{ and } \ g(+1) = 1,\, g(-1) = +1$$

$$\boldsymbol{\xi} = \max(0, x) \quad \Rightarrow \quad \lambda = a \ \text{ and } \ g(+1) = 1,\, g(-1) = 0$$

Example

$$\dot{x} = ax\,, \quad a < 0$$



$$\Gamma = \{-1, 1\}$$

$$\phi_{\lambda,g}(x) = \begin{cases} g(-1)|x|^{\frac{\lambda}{a}} & x < 0 \\ g(+1)|x|^{\frac{\lambda}{a}} & x > 0 \end{cases}$$

Optimal choice of λ and g depends on ξ

$$\xi = x \quad \Rightarrow \quad \lambda = a \;\text{ and }\; g(+1) = 1,\; g(-1) = -1$$

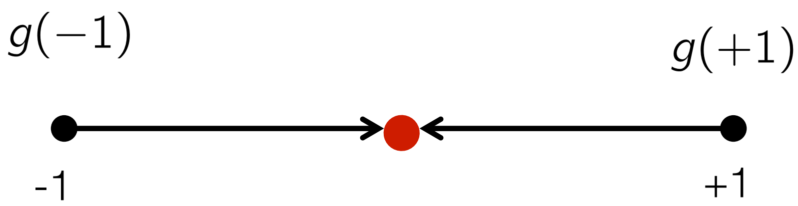
$$\xi = |x| \quad \Rightarrow \quad \lambda = a \;\text{ and }\; g(+1) = 1,\; g(-1) = +1$$

$$\xi = \max(0, x) \quad \Rightarrow \quad \lambda = a \;\text{ and }\; g(+1) = 1,\; g(-1) = 0$$

$$\xi = |x|^b \quad \Rightarrow \quad \lambda = ba \;\text{ and }\; g(+1) = 1,\; g(-1) = +1$$

Example

$$\dot{x} = ax\,, \quad a < 0$$



$$\Gamma = \{-1, 1\}$$

$$\phi_{\lambda,g}(x) = \begin{cases} g(-1)|x|^{\frac{\lambda}{a}} & x < 0 \\ g(+1)|x|^{\frac{\lambda}{a}} & x > 0 \end{cases}$$

Optimal choice of λ and g depends on $\boldsymbol{\xi}$

$\boldsymbol{\xi} = x \quad \Rightarrow \quad \lambda = a$ and $g(+1) = 1,\ g(-1) = -1$

$\boldsymbol{\xi} = |x| \quad \Rightarrow \quad \lambda = a$ and $g(+1) = 1,\ g(-1) = +1$

$\boldsymbol{\xi} = \max(0, x) \quad \Rightarrow \quad \lambda = a$ and $g(+1) = 1,\ g(-1) = 0$

$\boldsymbol{\xi} = |x|^b \quad \Rightarrow \quad \lambda = ba$ and $g(+1) = 1,\ g(-1) = +1$

But $\mathrm{span}\{\phi_{k\lambda,g} : k \in \mathbb{N},\, g \in \mathcal{C}(\Gamma)\}$ dense in $\mathcal{C}([-1,1])$ for any $\lambda < 0$

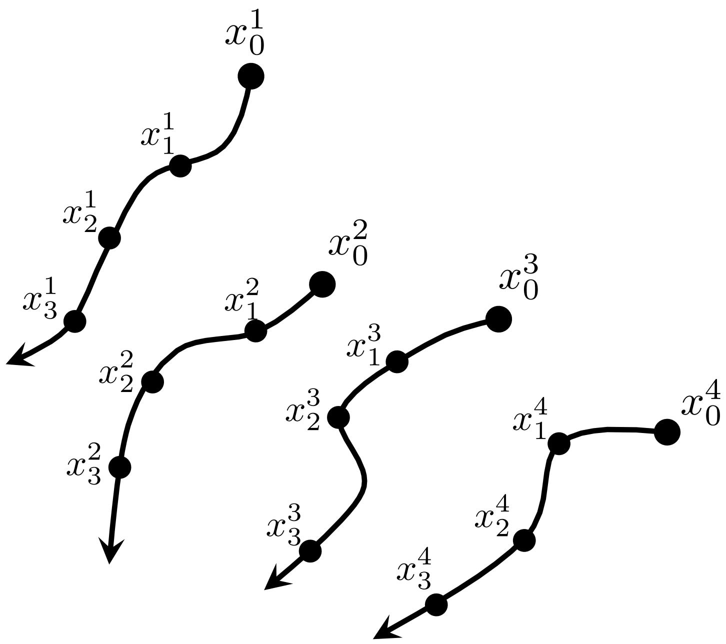
Data-driven construction

Data-driven construction

g = arbitrary continuous function
λ = arbitrary complex number

eigenfunction $\phi_{\lambda,g}$ defined on data

$$\phi_{\lambda,g}(x_k^j) := e^{\lambda k T_s} g(x_0^j)$$

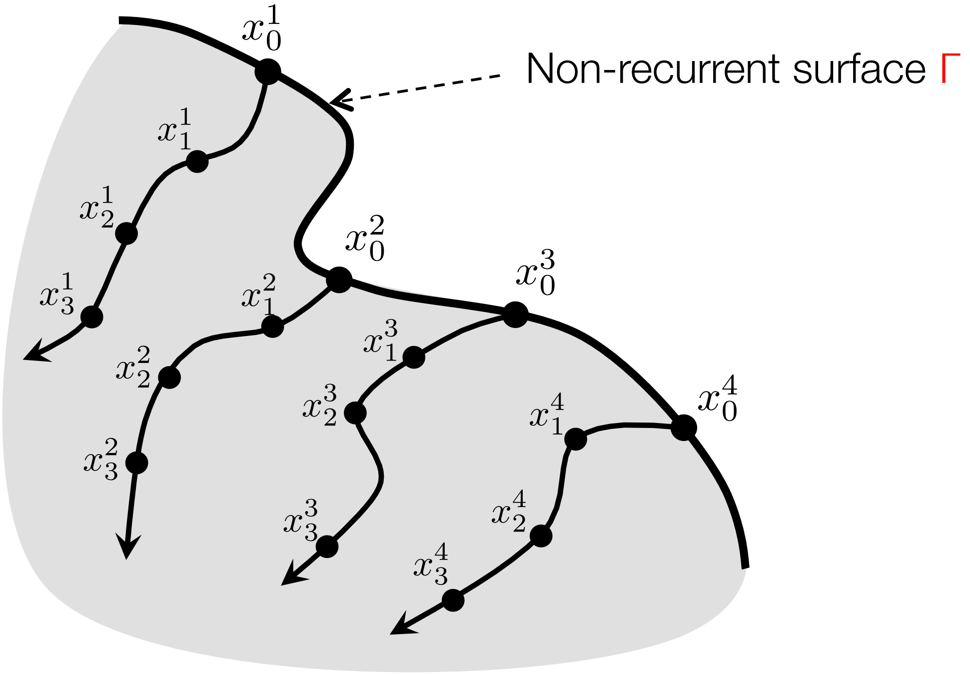


Data-driven construction

$g =$ arbitrary continuous function
$\lambda =$ arbitrary complex number

eigenfunction $\phi_{\lambda,g}$ defined on data

$$\phi_{\lambda,g}(x_k^j) := e^{\lambda k T_s} g(x_0^j)$$



Lemma: Flow rectifiable & initial conditions on distinct trajectories

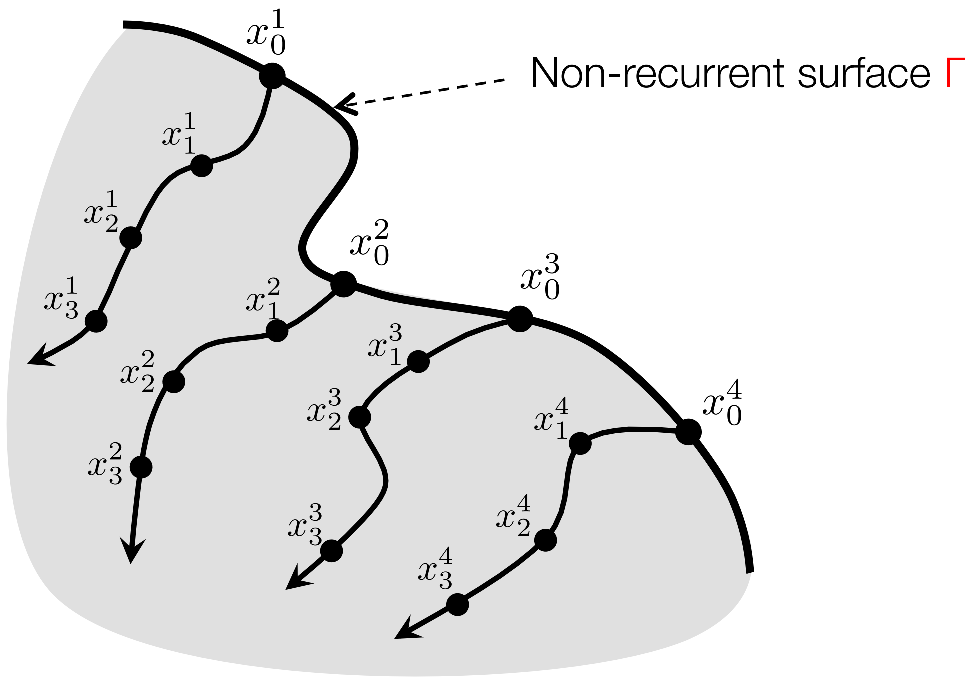
$\Rightarrow \exists$ non-recurrent surface Γ passing through initial conditions

Data-driven construction

$g =$ arbitrary continuous function
$\lambda =$ arbitrary complex number

eigenfunction $\phi_{\lambda,g}$ defined on data

$$\phi_{\lambda,g}(x_k^j) := e^{\lambda k T_s} g(x_0^j)$$



Lemma: Flow rectifiable & initial conditions on distinct trajectories
$\Rightarrow \exists$ non-recurrent surface Γ passing through initial conditions

$\Rightarrow \{\phi_{\lambda,g}(x_k^j)\}_{j,k}$ samples of a **continuous** eigenfunction $\Rightarrow$ can **interpolate**

Algorithm summary

Eigenfunction construction

Given trajectory data $(x_k^j)_{j,k}$

Choose $\lambda_1, \ldots, \lambda_{N_\lambda}$ complex numbers

Choose $g_1, \ldots, g_{N_g}$ continuous functions

Construct $N := N_\lambda N_g$ eigenfunctions by

Set $\phi_{\lambda,g}(x_k^j) := e^{\lambda k T_s} g(x_0^j)$ for each λ and g

Interpolate $\phi_{\lambda,g}(x_k^j)$ to get $\hat{\phi}_{\lambda,g}$

Output $\hat{\phi} = [\hat{\phi}_1, \ldots, \hat{\phi}_N]$

Algorithm summary

Eigenfunction construction

Given trajectory data $(x_k^j)_{j,k}$

Choose $\lambda_1, \ldots, \lambda_{N_\lambda}$ complex numbers

Choose $g_1, \ldots, g_{N_g}$ continuous functions

Construct $N := N_\lambda N_g$ eigenfunctions by

Set $\phi_{\lambda,g}(x_k^j) := e^{\lambda k T_s} g(x_0^j)$ for each λ and g

Interpolate $\phi_{\lambda,g}(x_k^j)$ to get $\hat{\phi}_{\lambda,g}$

Output $\hat{\phi} = [\hat{\phi}_1, \ldots, \hat{\phi}_N]$

Predictor matrices

Set $A = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$

Get C by minimizing $\sum_{i=1}^{M} \|\boldsymbol{\xi}(\bar{x}_i) - C\hat{\phi}(\bar{x}_i)\|^2$

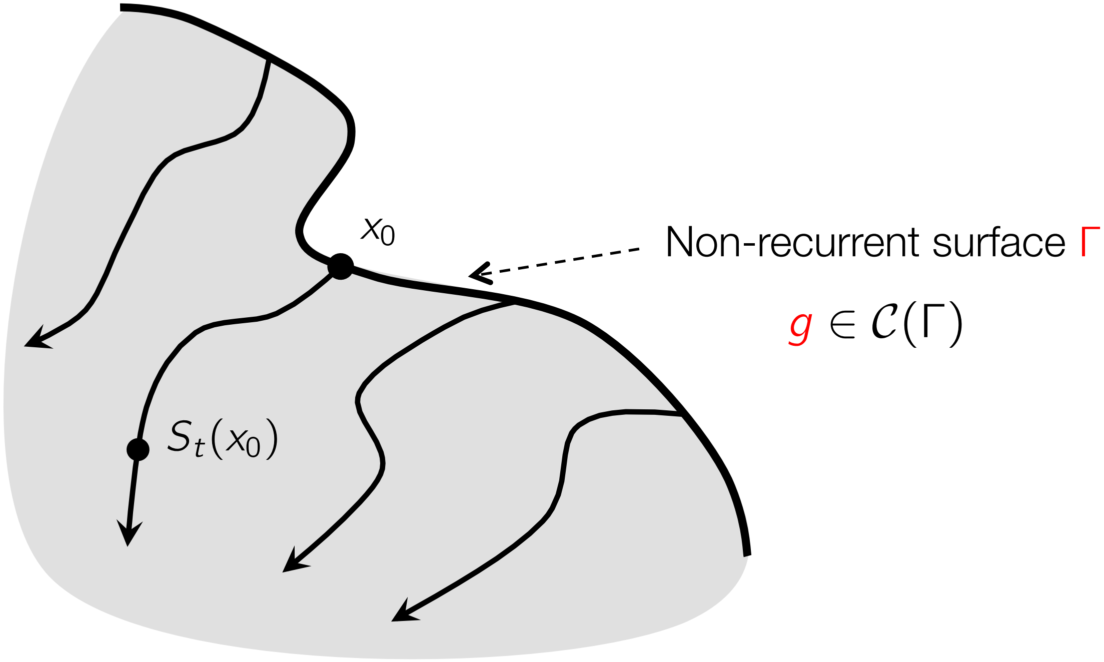
(Linear least-squares)

$$
\begin{aligned}
z_{k+1} &= A z_k \\
z_0 &= \hat{\phi}(x_0) \\
\hat{y}_k &= C z_k
\end{aligned}
$$

Optimal choice of boundary functions

Choice of boundary functions

$$\phi_{\lambda,g}(S_t(x_0)) = e^{\lambda t} g(x_0)$$



Observation: $\phi_{\lambda,g}$ depends linearly on g $\Rightarrow$ maybe can choose g using convex optimization

Choice of boundary functions

$$\phi_{\lambda,g}(S_t(x_0)) = e^{\lambda t} g(x_0)$$

Given $\lambda_1, \ldots, \lambda_{N_\lambda}$ there exist **linear** operators $\mathcal{L}_{\lambda_i}$ such that

$$\mathcal{L}_{\lambda_i} g = \phi_{\lambda,g}$$

Choice of boundary functions

$$\phi_{\lambda,g}(S_t(x_0)) = e^{\lambda t} g(x_0)$$

Given $\lambda_1, \ldots, \lambda_{N_\lambda}$ there exist **linear** operators $\mathcal{L}_{\lambda_i}$ such that

$$\mathcal{L}_{\lambda_i} g = \phi_{\lambda,g}$$

Given $\boldsymbol{\xi}$ we want to find $g_1, \ldots, g_{N_\lambda}$ such that

$$\|\boldsymbol{\xi} - \mathrm{Proj}_{\mathrm{span}\{\mathcal{L}_{\lambda_1} g_1, \ldots, \mathcal{L}_{\lambda_{N_\lambda}} g_{N_\lambda}\}} \boldsymbol{\xi}\| \quad \text{is minimized}$$

Is this convex in g's?

Choice of boundary functions

$$\phi_{\lambda,g}(S_t(x_0)) = e^{\lambda t} g(x_0)$$

Given $\lambda_1, \ldots, \lambda_{N_\lambda}$ there exist **linear** operators $\mathcal{L}_{\lambda_i}$ such that

$$\mathcal{L}_{\lambda_i} g = \phi_{\lambda,g}$$

Given $\boldsymbol{\xi}$ we want to find $g_1, \ldots, g_{N_\lambda}$ such that

$$\|\boldsymbol{\xi} - \mathrm{Proj}_{\mathrm{span}\{\mathcal{L}_{\lambda_1} g_1, \ldots, \mathcal{L}_{\lambda_{N_\lambda}} g_{N_\lambda}\}} \boldsymbol{\xi}\| \quad \text{is minimized}$$

Is this convex in g's?

Answer: probably not, but a convex reformulation exists if each component of $\boldsymbol{\xi}$ is considered separately

Convex reformulation

$$\underset{g_i \in \mathcal{C}(\Gamma)}{\text{minimize}} \, \|\boldsymbol{\xi} - \text{Proj}_{\text{span}\{\mathcal{L}_{\lambda_1} g_1, \ldots, \mathcal{L}_{\lambda_{N_\lambda}} g_{N_\lambda}\}} \boldsymbol{\xi}\|$$

$$\Updownarrow$$

$$\underset{g_i \in \mathcal{C}(\Gamma),\; c_i \in \mathbb{C}^{N_{\boldsymbol{\xi}}}}{\text{minimize}} \left\| \boldsymbol{\xi} - \sum_{i=1}^{N_\lambda} c_i \mathcal{L}_{\lambda_i} g_i \right\|$$

$\Updownarrow$ $\boldsymbol{\xi}$ scalar & substitution $\tilde{g}_i = c_i g_i$

$$\underset{\tilde{g}_i \in \mathcal{C}(\Gamma)}{\text{minimize}} \left\| \boldsymbol{\xi} - \sum_{i=1}^{N_\lambda} \mathcal{L}_{\lambda_i} \tilde{g}_i \right\|$$

Convex

Regularization

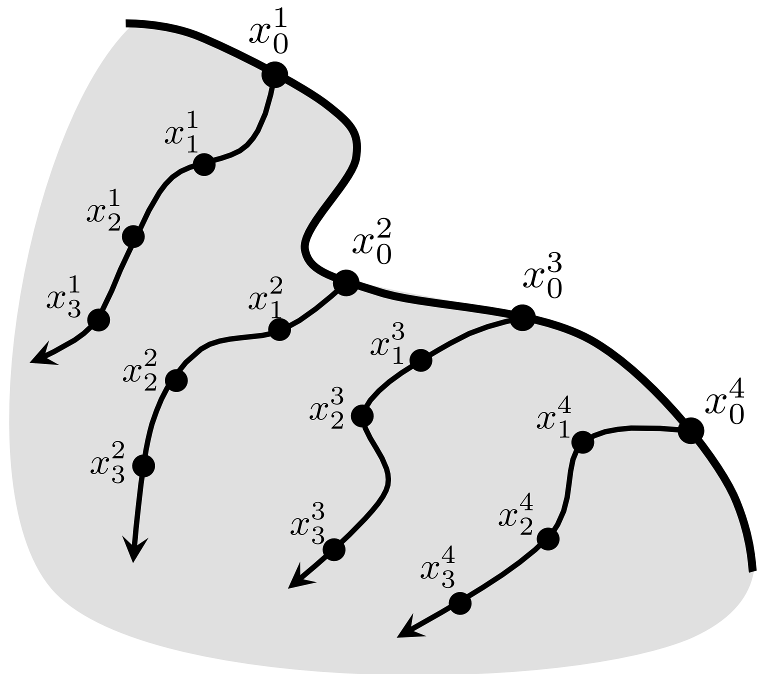
$$\underset{g_1,\ldots,g_{N_\lambda}}{\text{minimize}} \left\| \boldsymbol{\xi} - \sum_{i=1}^{N_\lambda} \mathcal{L}_{\lambda_i} g_i \right\| \quad + \quad \text{regularizer}\,(g_1,\ldots,g_{N_\lambda})$$

Examples

$$\text{regularizer} = \sum_i \text{Lipschitz}(\mathcal{L}_{\lambda_i} g_i)$$

$$\text{regularizer} = \sum_i \int \|\nabla \mathcal{L}_{\lambda_i} g_i\|^2$$

Data-driven construction



When restricted to the available data set, the operators $\mathcal{L}_i$ become matrices

the functions g become vectors

$\Rightarrow$ Finite-dimensional convex optimization problem

l_2 norm squared + quadratic regularization $\Rightarrow$ **least-squares**

Adding control

Adding control

$$
\begin{aligned}
z_{k+1} &= A z_k + B u_k \\
z_0 &= \hat{\phi}(x_0) \\
\hat{y}_k &= C z_k
\end{aligned}
$$

A, C, $\hat{\phi}$ known

Minimize multi-step prediction error

$$\underset{B \in \mathbb{R}^{N \times m}}{\text{minimize}} \sum_{j=1}^{\#\mathrm{traj}} \sum_{k=1}^{\mathrm{trajLen}} \|\xi(x_k^j) - \hat{y}_k(x_0^j)\|_2^2,$$

$\hat{y}_k$ is **linear** in B

$$\hat{y}_k(x_0^j) = C A^k z_0^j + \sum_{i=0}^{k-1} C A^{k-i-1} B u_i^j$$

Adding control

$$z_{k+1} = Az_k + Bu_k$$
$$z_0 = \hat{\phi}(x_0)$$
$$\hat{y}_k = Cz_k$$

A, C, $\hat{\phi}$ known

Minimize multi-step prediction error

$$\underset{B\in\mathbb{R}^{N\times m}}{\text{minimize}} \sum_{j=1}^{\#\text{traj}} \sum_{k=1}^{\text{trajLen}} \|\xi(x_k^j) - \hat{y}_k(x_0^j)\|_2^2,$$

$\hat{y}_k$ is **linear** in B $\qquad \hat{y}_k(x_0^j) = CA^k z_0^j + \sum_{i=0}^{k-1} CA^{k-i-1} B u_i^j$

&

A and C **known** $\Rightarrow$ $\underset{b\in\mathbb{R}^{Nm}}{\text{minimize}} \ \|\Theta b - \theta\|^2$ where $b = \text{vec}(B)$

Linear least-squares problem $\Rightarrow$ $B = \text{vec}^{-1}(\Theta^\dagger \theta)$

Koopman MPC *[Korda, Mezić 2018]*

Nonlinear MPC

$$
\begin{array}{lll}
\underset{u_i, x_i}{\text{minimize}} & \sum_{i=0}^{N_p-1} l_x(x_i) + u_i^\top R u_i + r^\top u_i & \\
\text{subject to} & {\color{red} x_{i+1} = f(x_i, u_i)}, & i = 0, \ldots, N_p - 1 \\
& c_x(x_i) + C_u u_i \le b, & i = 0, \ldots, N_p - 1 \\
\text{parameter} & {\color{blue} x_0 = x} &
\end{array}
$$

x

$$\kappa(x) = \{u_0^\star, u_1^\star, \ldots, u_{N_p-1}^\star\} \longrightarrow$$

$$x^+ = f(x, u)$$

Koopman MPC *[Korda, Mezić 2018]*

Koopman MPC

$$
\begin{array}{lll}
\underset{u_i, z_i, \hat{y}_i}{\text{minimize}} & \sum_{i=0}^{N_p-1} \hat{y}_i^\top Q \hat{y}_i + u_i^\top R u_i + q^\top \hat{y}_i + r^\top u_i & \\
\text{subject to} & z_{i+1} = {\color{red}A} z_i + {\color{red}B} u_i, & i = 0, \ldots, N_p - 1 \\
& \hat{y}_i = {\color{red}C} z_i & i = 0, \ldots, N_p - 1 \\
& E z_i + F u_i \le b, & i = 0, \ldots, N_p - 1 \\
\text{parameter} & {\color{blue} z_0 = \hat{\phi}(x)} &
\end{array}
$$

$$\kappa(x) = \{u_0^\star, u_1^\star, \ldots, u_{N_p-1}^\star\} \longrightarrow \quad x^+ = f(x, u) \quad \uparrow x$$

Can handle **nonlinear constraints** and **costs** in a linear fashion

Koopman MPC *[Korda, Mezić 2018]*

Dense-form Koopman MPC

$$\begin{aligned}
\underset{\mathbf{u}\in\mathbb{R}^{mN_p}}{\text{minimize}} \quad & \mathbf{u}^\top H \mathbf{u}^\top + h^\top \mathbf{u} + z_0^\top G \mathbf{u} \\
\text{subject to} \quad & L\mathbf{u} + M z_0 \le c \\
\text{parameter} \quad & z_0 = \hat{\phi}(x)
\end{aligned}$$

Convex QP!

$$\kappa(x) = \begin{bmatrix} u_0^\star \\ \vdots \\ u_{N_p-1}^\star \end{bmatrix} \longrightarrow \quad x^+ = f(x,u)$$

x

Computation cost **independent** of the size of the lift!

Koopman MPC summary

At each step of closed-loop operation

- Set $z_0 = \hat{\phi}(x_{\text{current}})$
- Solve

$$\begin{array}{ll} \underset{\mathbf{u}\in\mathbb{R}^{mN_p}}{\text{minimize}} & \mathbf{u}^\top H \mathbf{u}^\top + h^\top \mathbf{u} + z_0^\top G \mathbf{u} \\ \text{subject to} & L\mathbf{u} + M z_0 \le c \end{array} \quad \Rightarrow \quad \mathbf{u}^\star = \begin{bmatrix} u_0^\star \\ \vdots \\ u_{N_p-1}^\star \end{bmatrix}$$

- Apply $u_0^\star$ to the system

Main benefits

Data-driven: No model required

Fast & **simple**: only small **convex quadratic program** solved online

Nonlinear constraints and **costs** handled in a linear fashion

Numerical examples

Numerical examples – damped Duffing

Dynamics

$$\dot{x}_1 = x_2$$

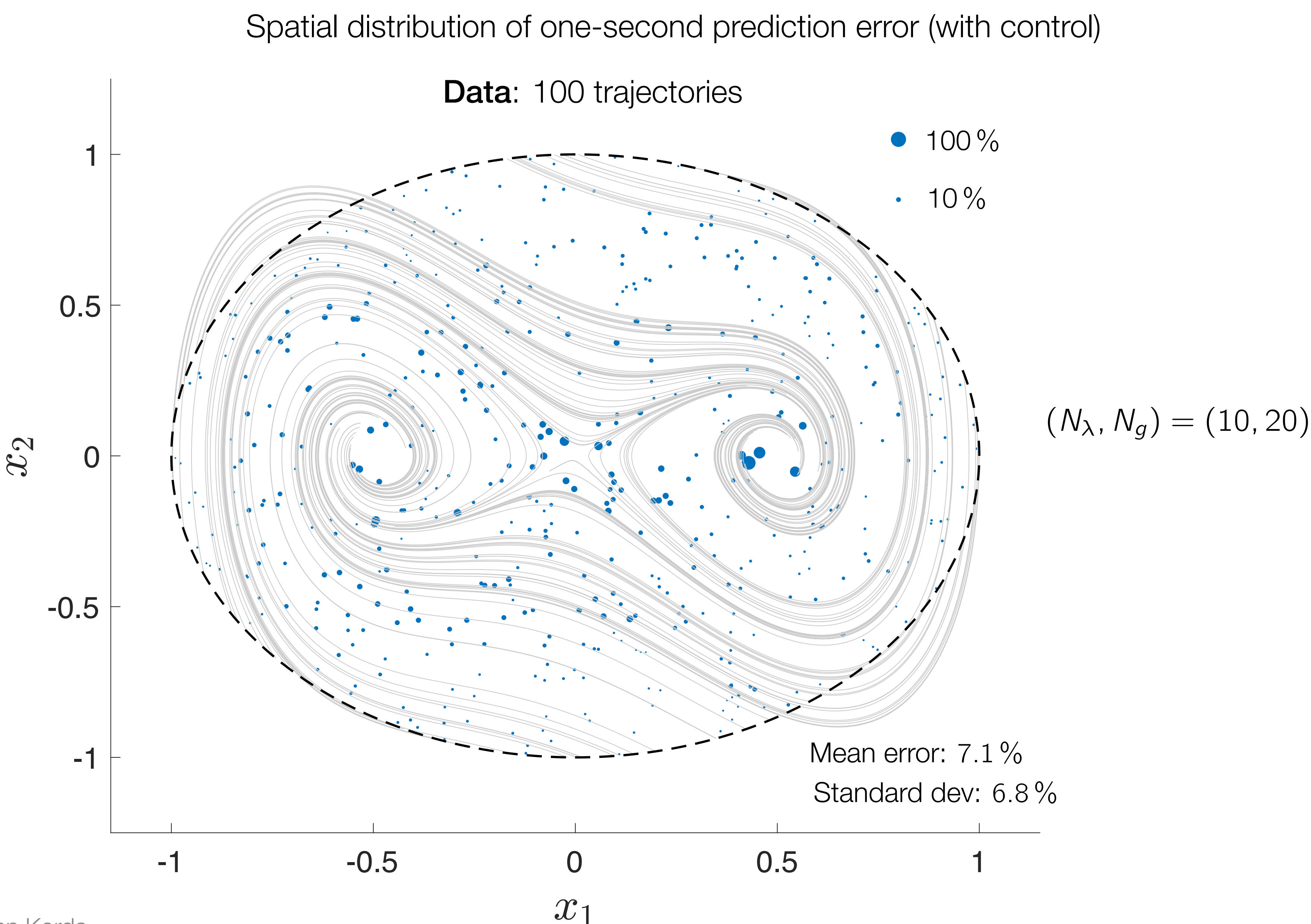
$$\dot{x}_2 = -0.5x_2 - x_1(4x_1^2 - 1) + 0.5u$$

Data: 100 trajectories, 8 second long

Eigenvalues: Lattice from DMD eigenvalues

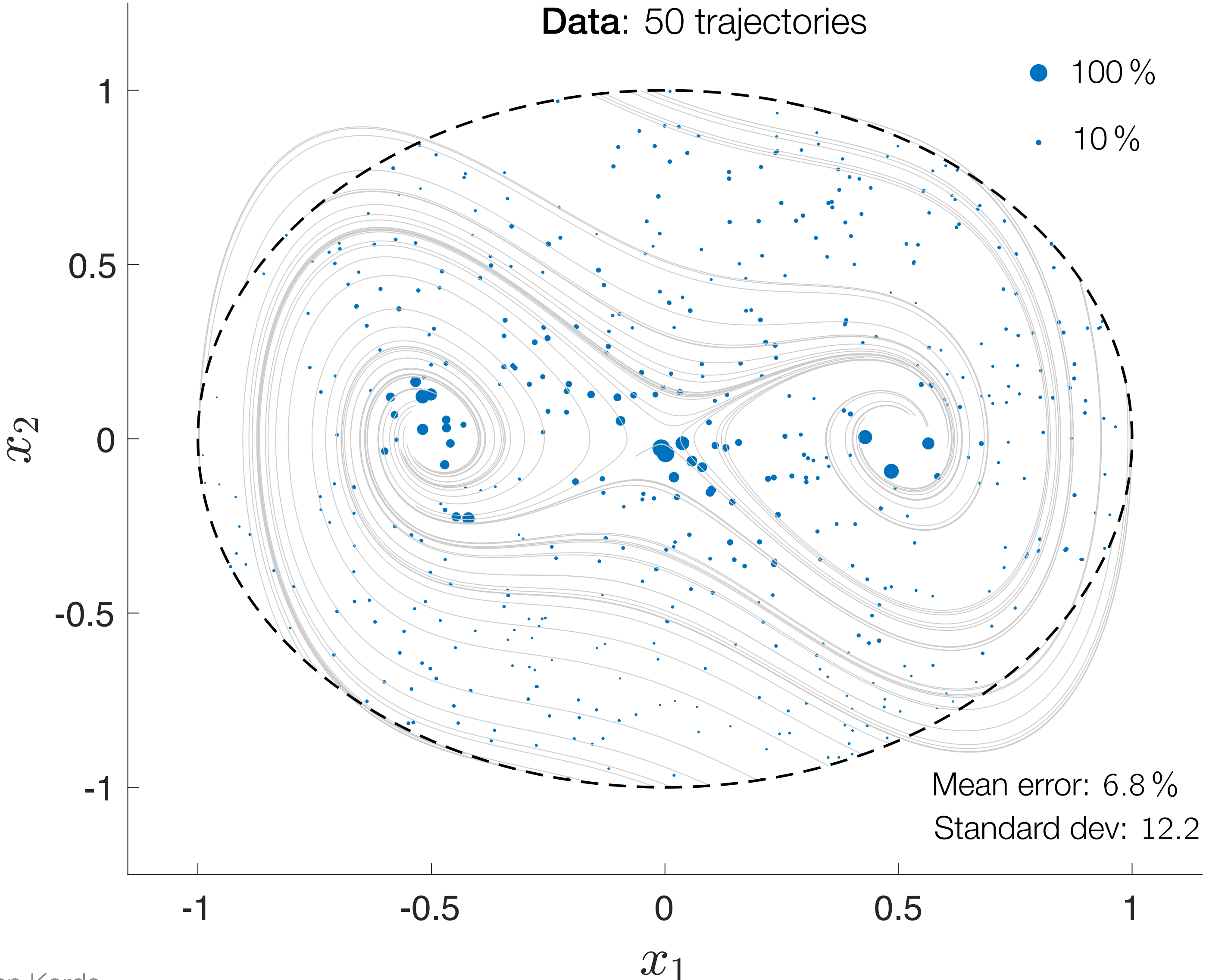
Boundary functions: Thin plate spline RBFs

Numerical examples – damped Duffing



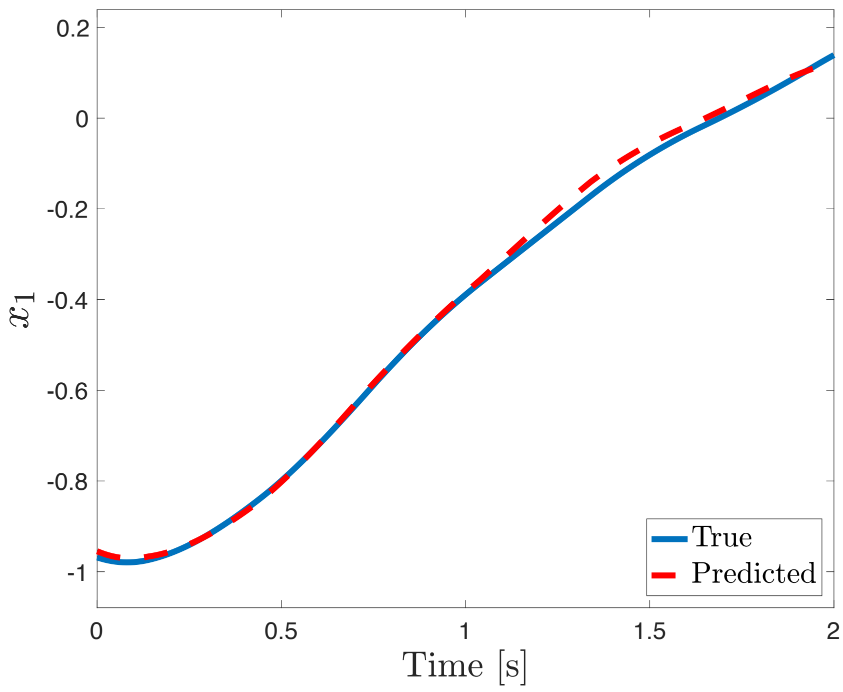
Numerical examples – damped Duffing

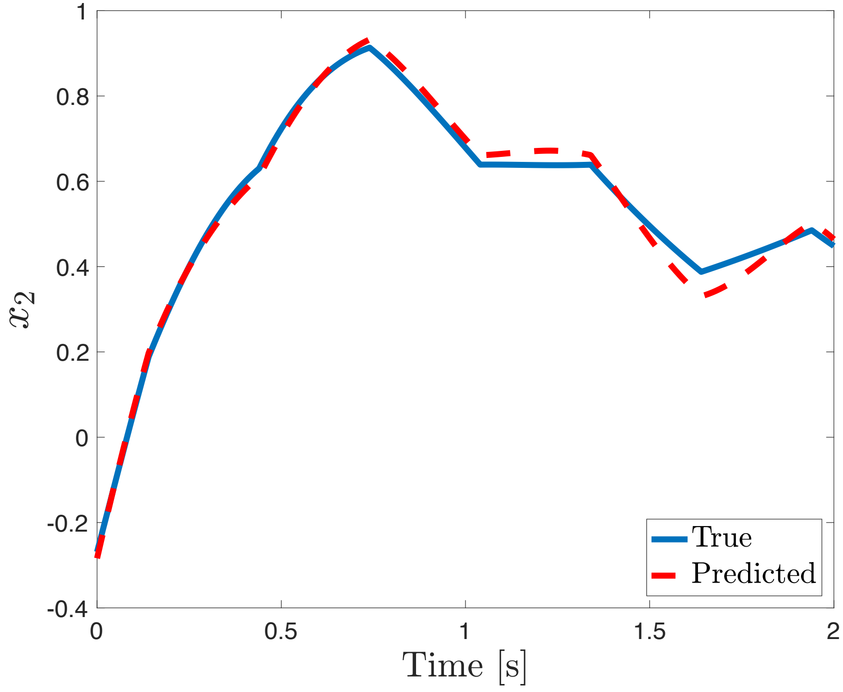
Spatial distribution of one-second prediction error (with control)



$(N_\lambda, N_g) = (10, 20)$

Numerical examples – damped Duffing





$$(N_\lambda, N_g) = (10, 20)$$

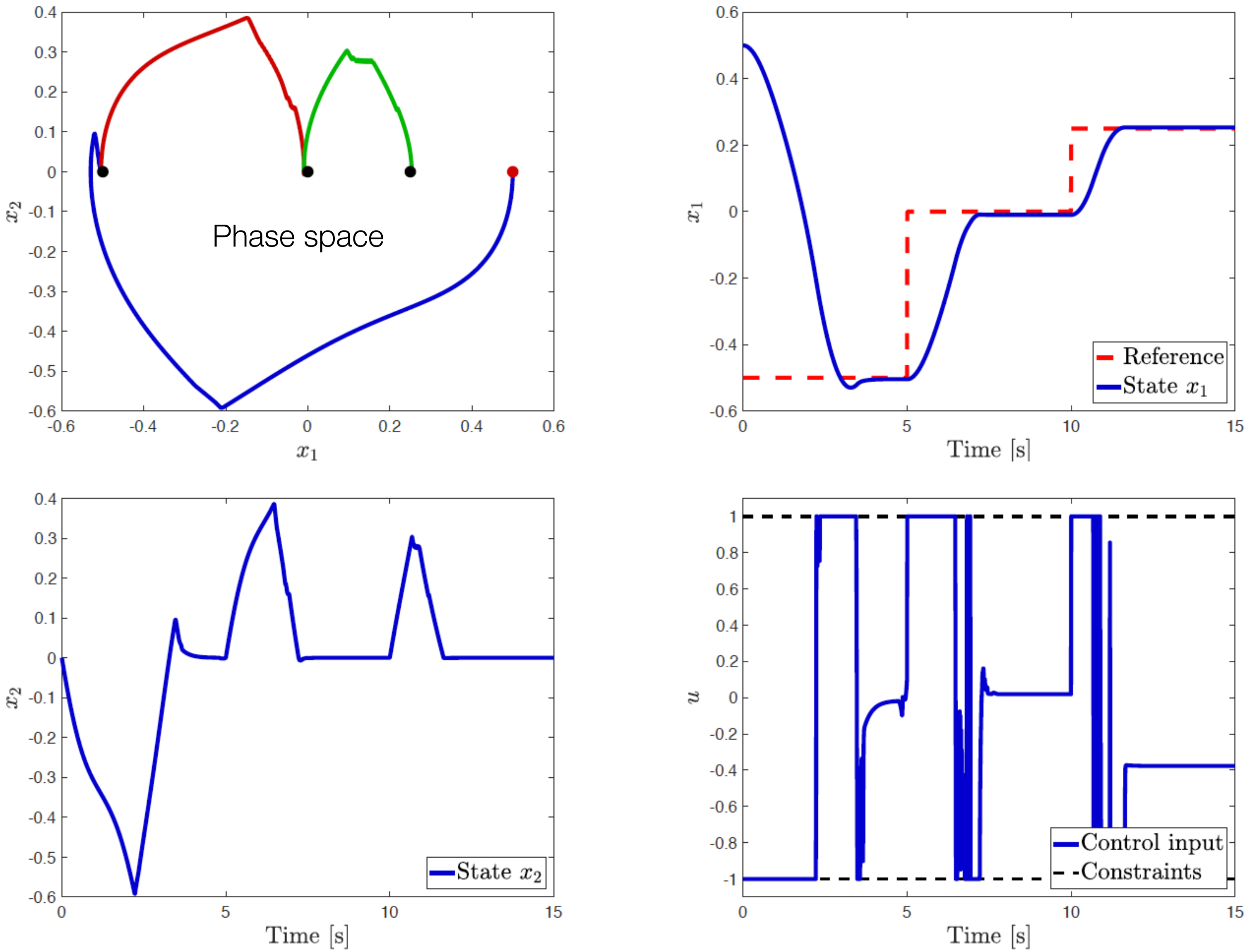
Numerical examples – damped Duffing

(N_Λ, N_G)	$(10,30)$	$(10,20)$	$(6,20)$	$(10,10)$	$(10,5)$	$(10,3)$
Mean error [uncontrolled]	6.9 %	8.9 %	17.4 %	19.9 %	38.8 %	56.2 %
Mean error [controlled]	4.6 %	6.7 %	15.8 %	15.7 %	35.6 %	53.5 %

EDMD error (200 RBF basis functions) = 25.1 %

Numerical examples – damped Duffing

Feedback control – Koopman MPC



Numerical examples – Van der Pol

Dynamics

$$\dot{x}_1 = 2x_2$$
$$\dot{x}_2 = -0.8x_1 + 2x_2 - 10x_1^2 x_2 + u$$

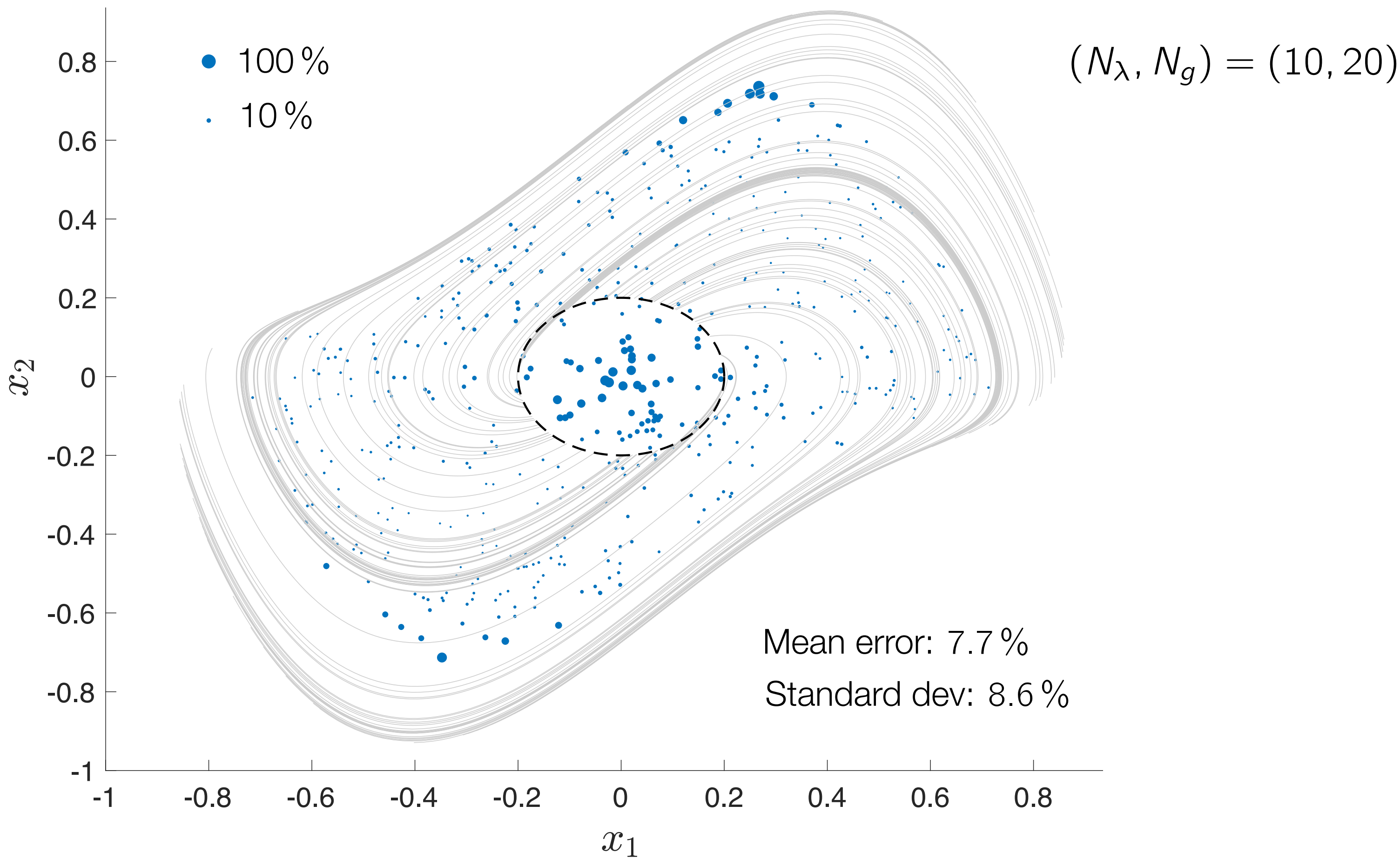
Data: 100 trajectories, 3 second long

Eigenvalues: Lattice from DMD eigenvalues

Boundary functions: Thin plate spline RBFs

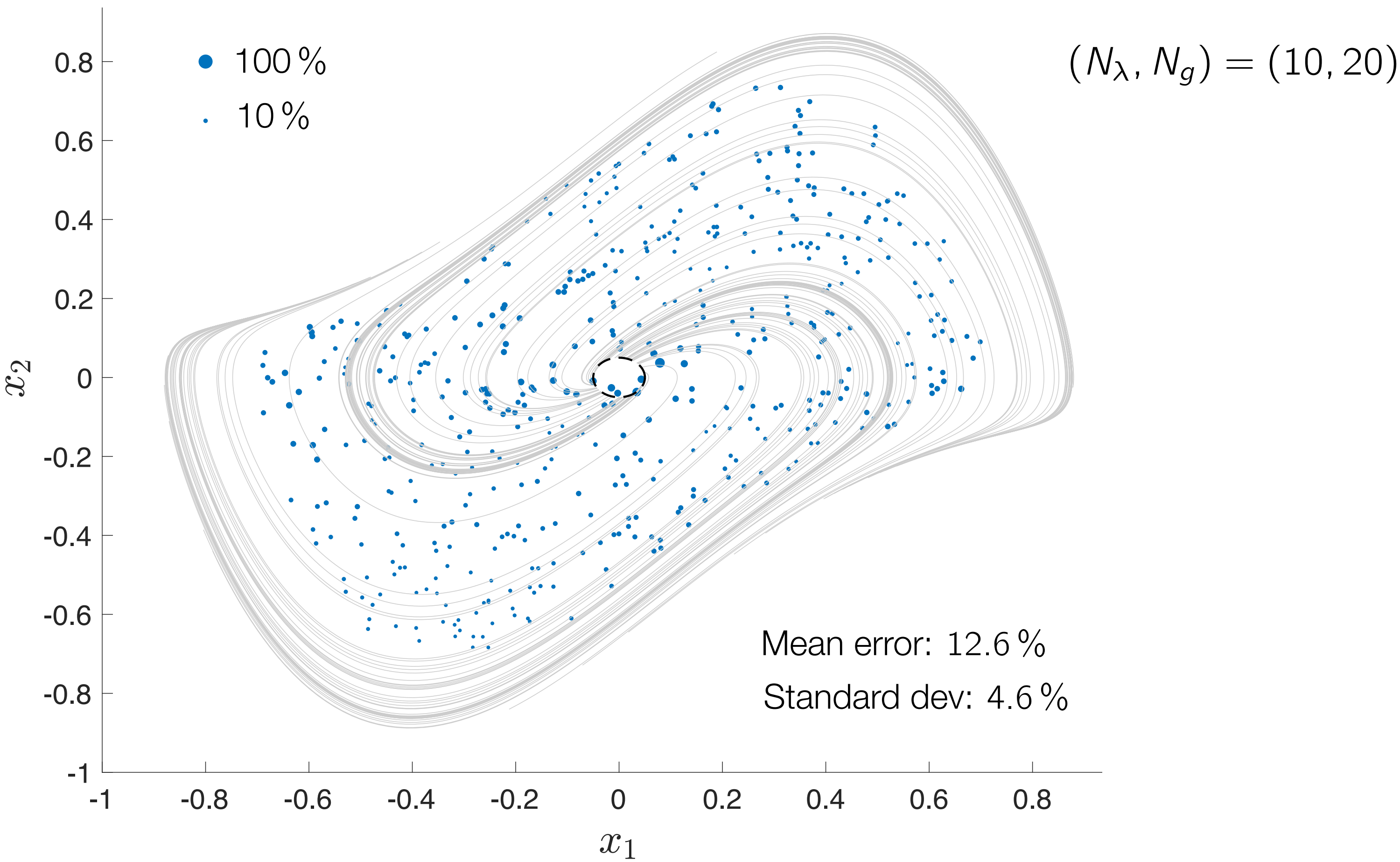
Numerical examples – Van der Pol

Spatial distribution of one-second prediction error (with control)

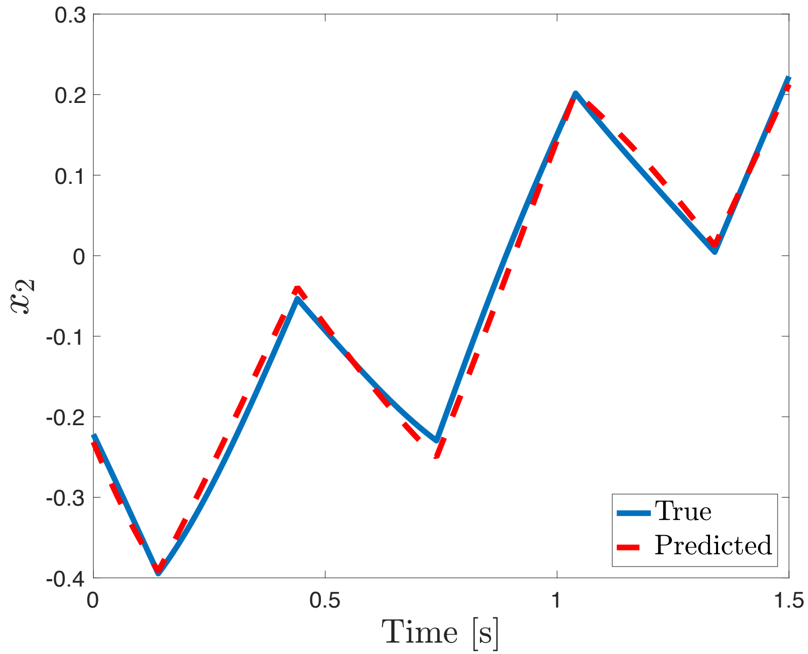


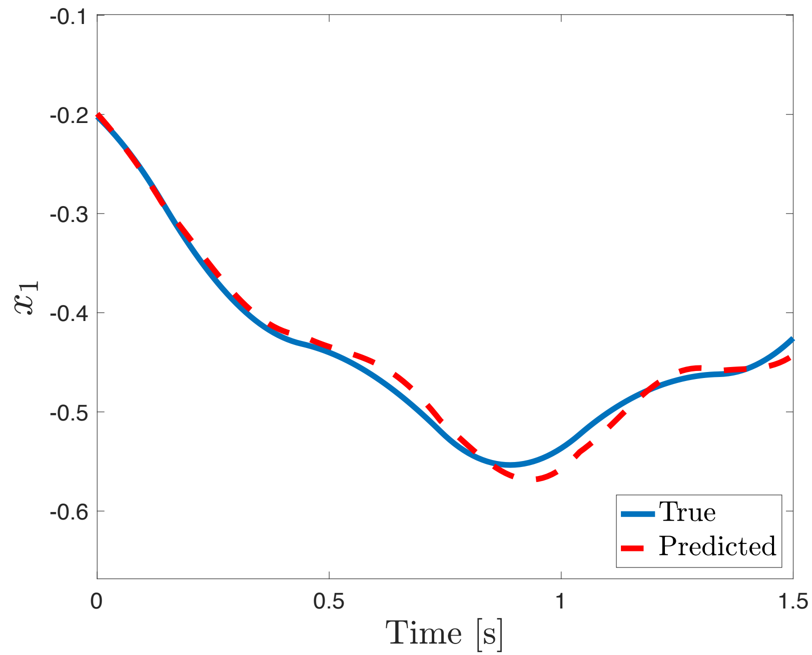
Numerical examples – Van der Pol

Spatial distribution of one-second prediction error (with control)



Numerical examples – Van der Pol





$$(N_\lambda, N_g) = (10, 20)$$

Numerical examples – Van der Pol

Mean prediction error for different number of eigenfunctions

(N_λ, N_g)	$(10, 20)$	$(6, 20)$	$(10, 10)$	$(10, 5)$	$(10, 3)$
Mean error [uncontrolled]	10.4 %	18.5 %	14.0 %	26.4 %	33.4 %
Mean error [controlled]	12.6 %	18.3 %	16.0 %	26.5 %	34.2 %

Optimized x Not optimized

Total # of eigenfunctions

Without optimization of g

$$N = N_\lambda N_G$$

With optimization of g

$$N = N_\lambda N_{\boldsymbol{\xi}}$$

Optimized x Not optimized

Total # of eigenfunctions

Without optimization of g

$$N = N_\lambda N_G$$

With optimization of g

$$N = N_\lambda N_{\boldsymbol{\xi}}$$

Duffing

	Not optimized						Optimized	
N	300	200	120	100	50	30	20	12
Mean error [controlled]	4.6 %	6.7 %	15.8 %	15.7 %	35.6 %	53.5 %	5.03 %	15.4 %

Optimized x Not optimized

Total # of eigenfunctions

Without optimization of g

$$N = N_\lambda N_G$$

With optimization of g

$$N = N_\lambda N_{\boldsymbol{\xi}}$$

Duffing

	Not optimized						Optimized	
N	300	200	120	100	50	30	20	12
Mean error [controlled]	4.6 %	6.7 %	15.8 %	15.7 %	35.6 %	53.5 %	5.03 %	15.4 %

Van der Pol

	Not optimized						Optimized	
N	300	200	120	100	50	30	30	20
Mean error [controlled]	11.6 %	12.6 %	18.3 %	16.0 %	26.5 %	34.2 %	7.6 %	13.6 %

Open problems

Open problem 1:

Optimal choice of g jointly for all components of $\boldsymbol{\xi}$

$$\underset{x\in\mathbb{R}^n}{\text{maximize}} \quad x^\top A^\top \left(\sum_i B_i x x^\top B_i^\top\right)^{-1} A x$$

Remarks: Homogenous of degree zero

Similar to (generalized) eigenvalue problem

Open problem 2:

Optimal choice of eigenvalues

$$\phi_{\lambda,g}(S_t(x_0)) = e^{\lambda t} g(x_0)$$

Remarks: Log-linear in λ

Geometric programming?

Low-dimensional

Conclusion

- Data-driven construction of Koopman eigenfunctions

  Only linear algebra and/or convex optimization needed

  Readily applicable to control and estimation

  Seems very robust

Future work

- High dimensional interpolation / approximation

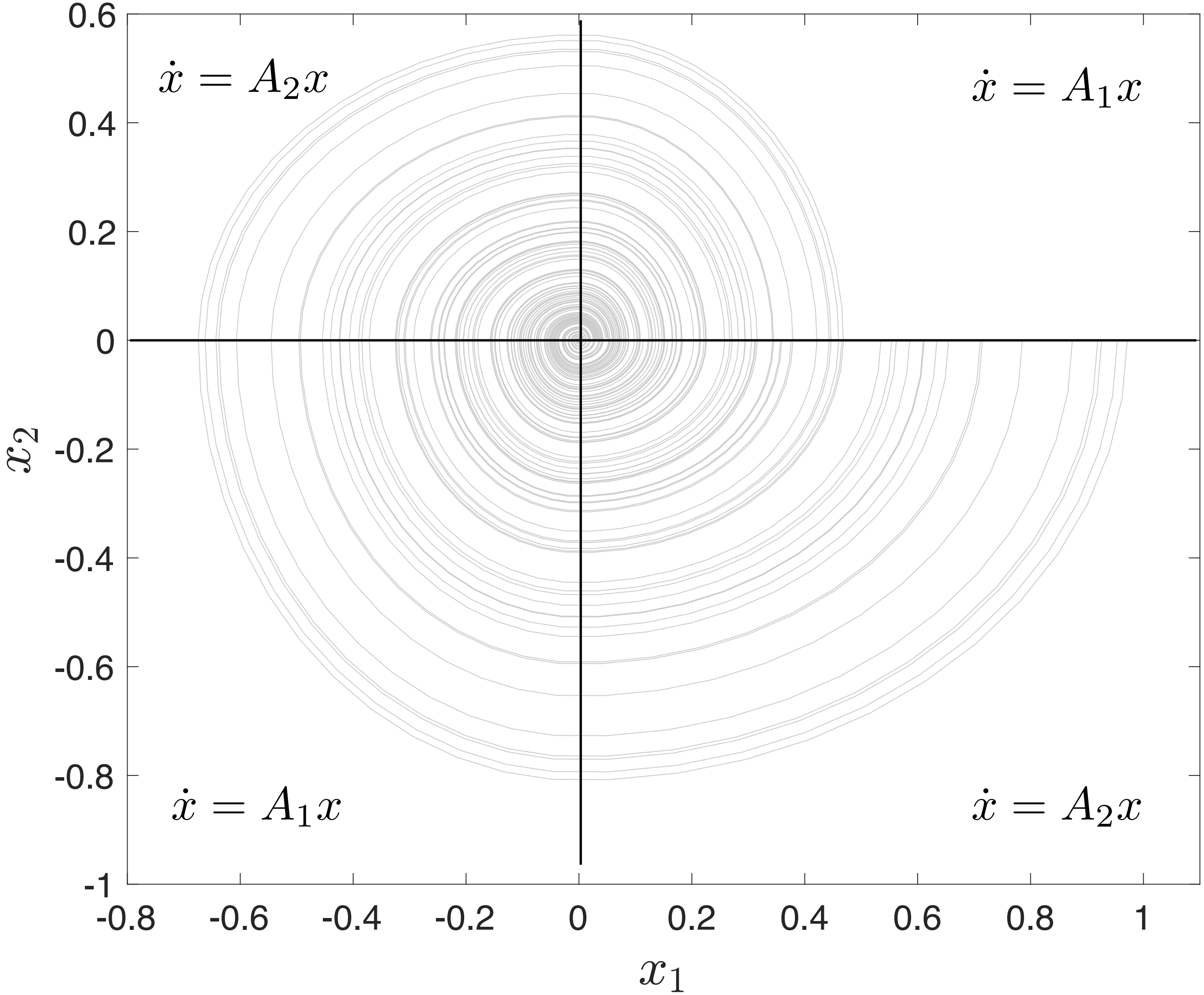
- Exploit **algebraic structure** (products of eigenfunctions)

  $\phi_1, \ldots, \phi_N$ eigenfunctions $\Rightarrow$ $\phi_1^{p_1} \cdot \ldots \cdot \phi_N^{p_N}$ also an eigenfunction

- Generalized eigenfunctions – Jordan blocks

$$\begin{bmatrix} \phi_1(x(t)) \\ \phi_2(x(t)) \end{bmatrix} := \exp\left(t \begin{bmatrix} \lambda & 1 \\ 0 & \lambda \end{bmatrix} \right) \begin{bmatrix} g_1(x_0) \\ g_2(x_0) \end{bmatrix} \quad \Rightarrow \quad \mathrm{span}\{\phi_1, \phi_2\} \text{ is invariant!}$$

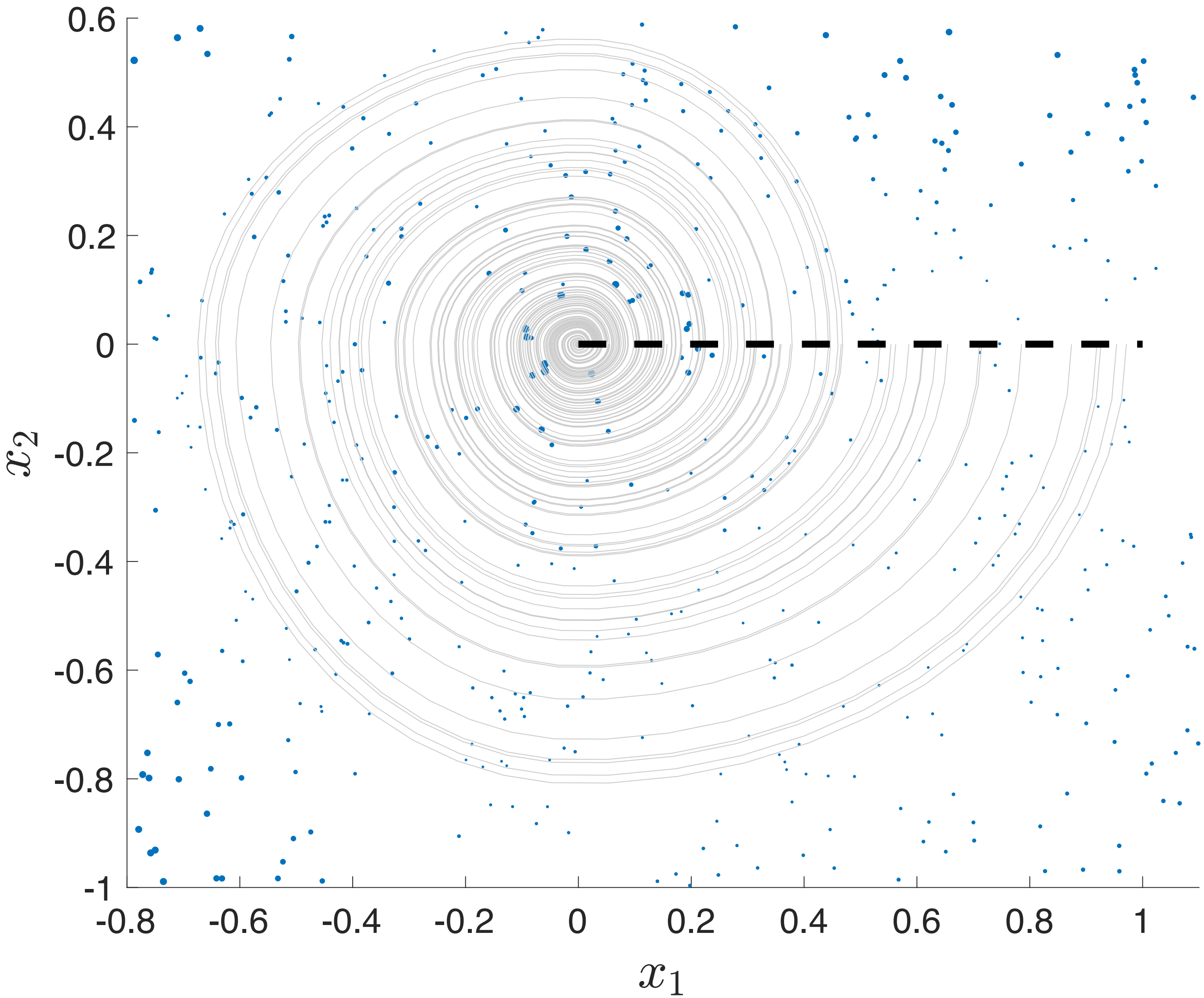
Switched linear system



Switched linear system

Mean error = 5.8 %

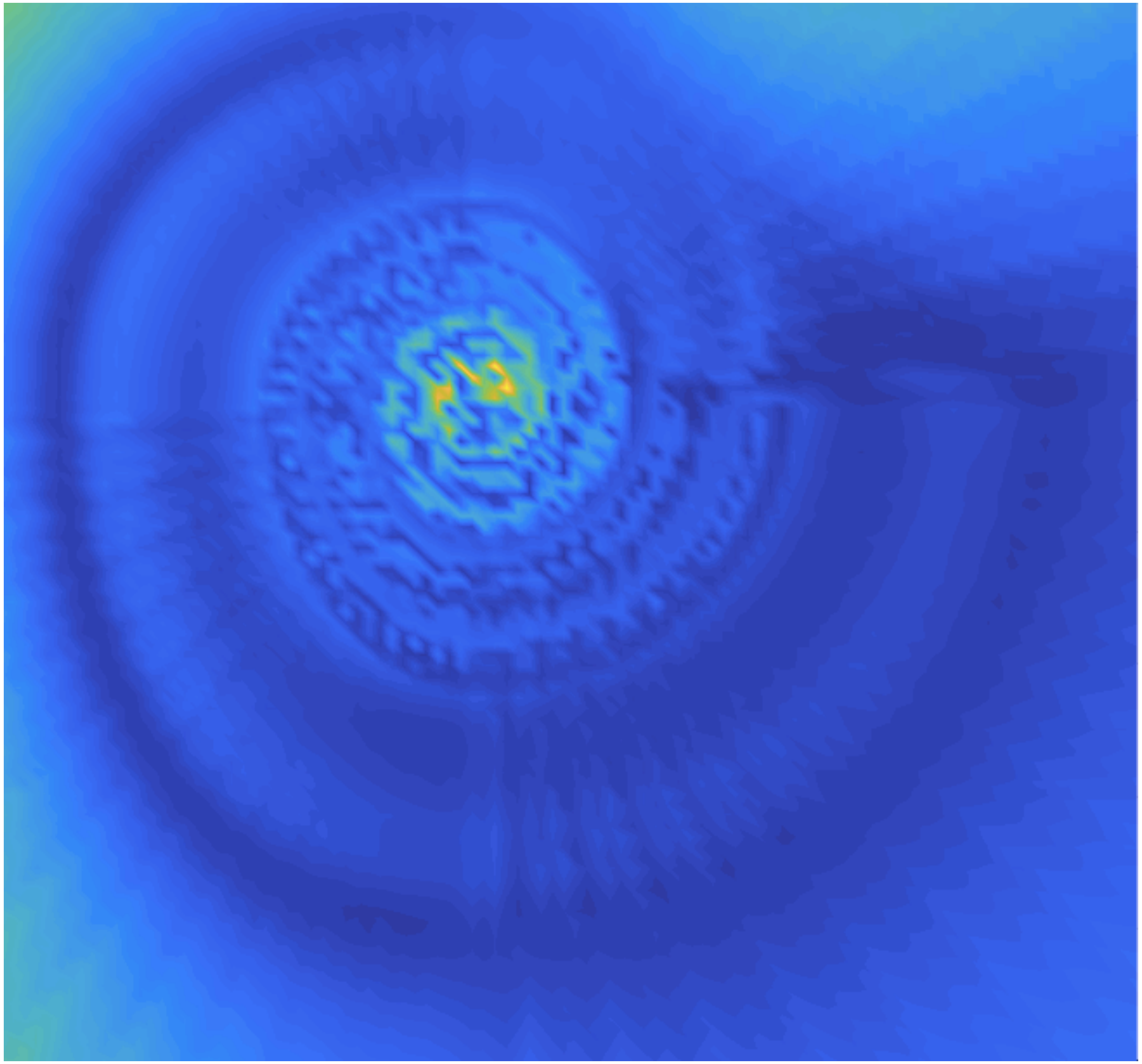
Standard dev. = 3.4 %



Switched linear system

Mean error = 5.8 %

Standard dev. = 3.4 %



Switched linear system

Mean error = 5.8 %

Standard dev. = 3.4 %