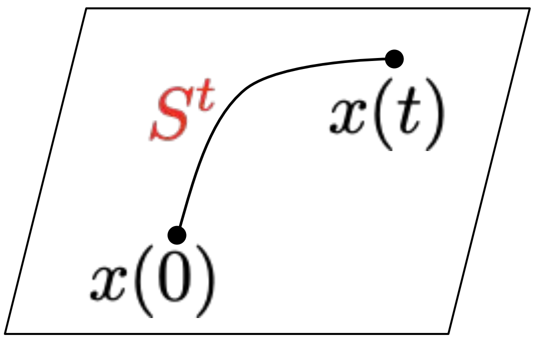
Learning Koopman eigenfunctions for prediction and control

Presenter: Poorva Shukla
(University of California, Santa Barbara)

Work by: Milan Korda
(LAAS-CNRS)
and
Igor Mezić
(University of California, Santa Barbara)

Linear predictor



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

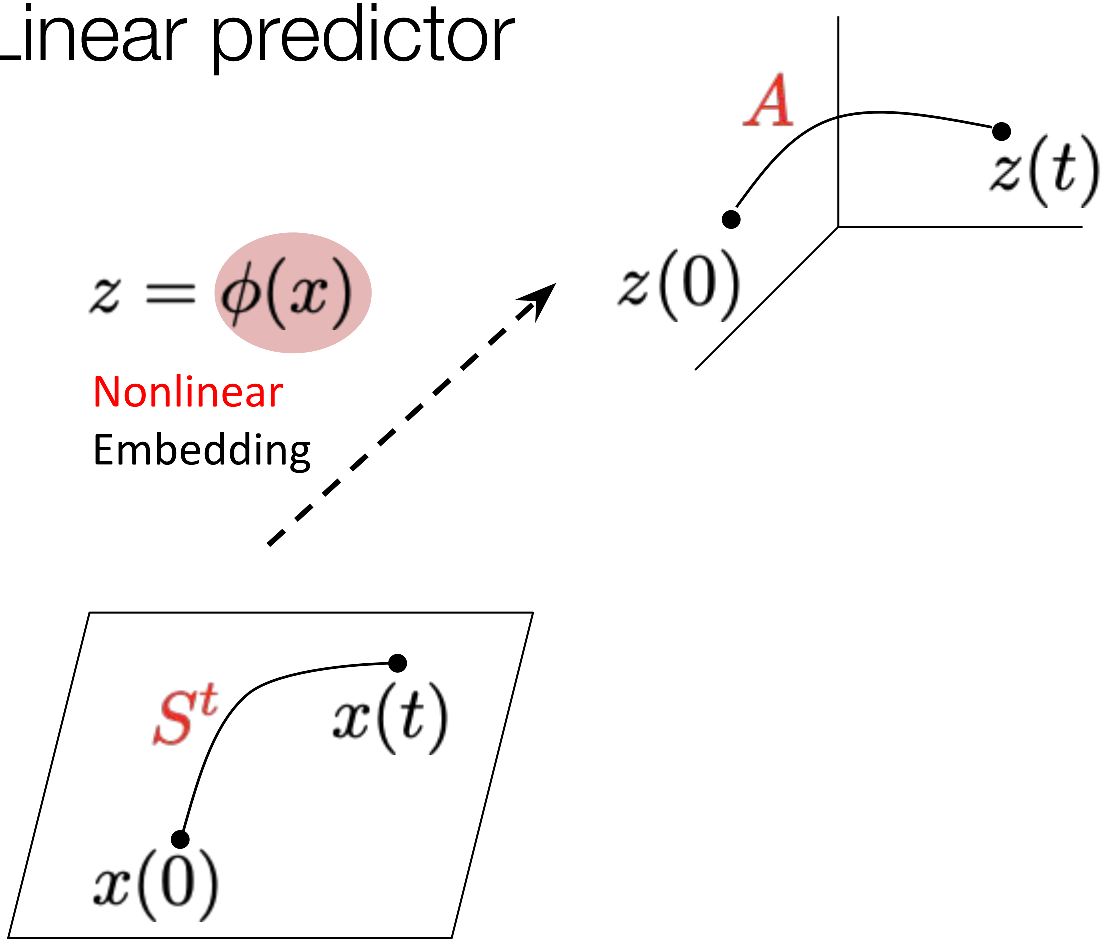
Nonlinear
Dynamics

$$\xi(x)$$

Vector of Observables

$$(e.g.\ \xi(x) = x)$$

Linear predictor



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

Nonlinear Dynamics

Linear Dynamics

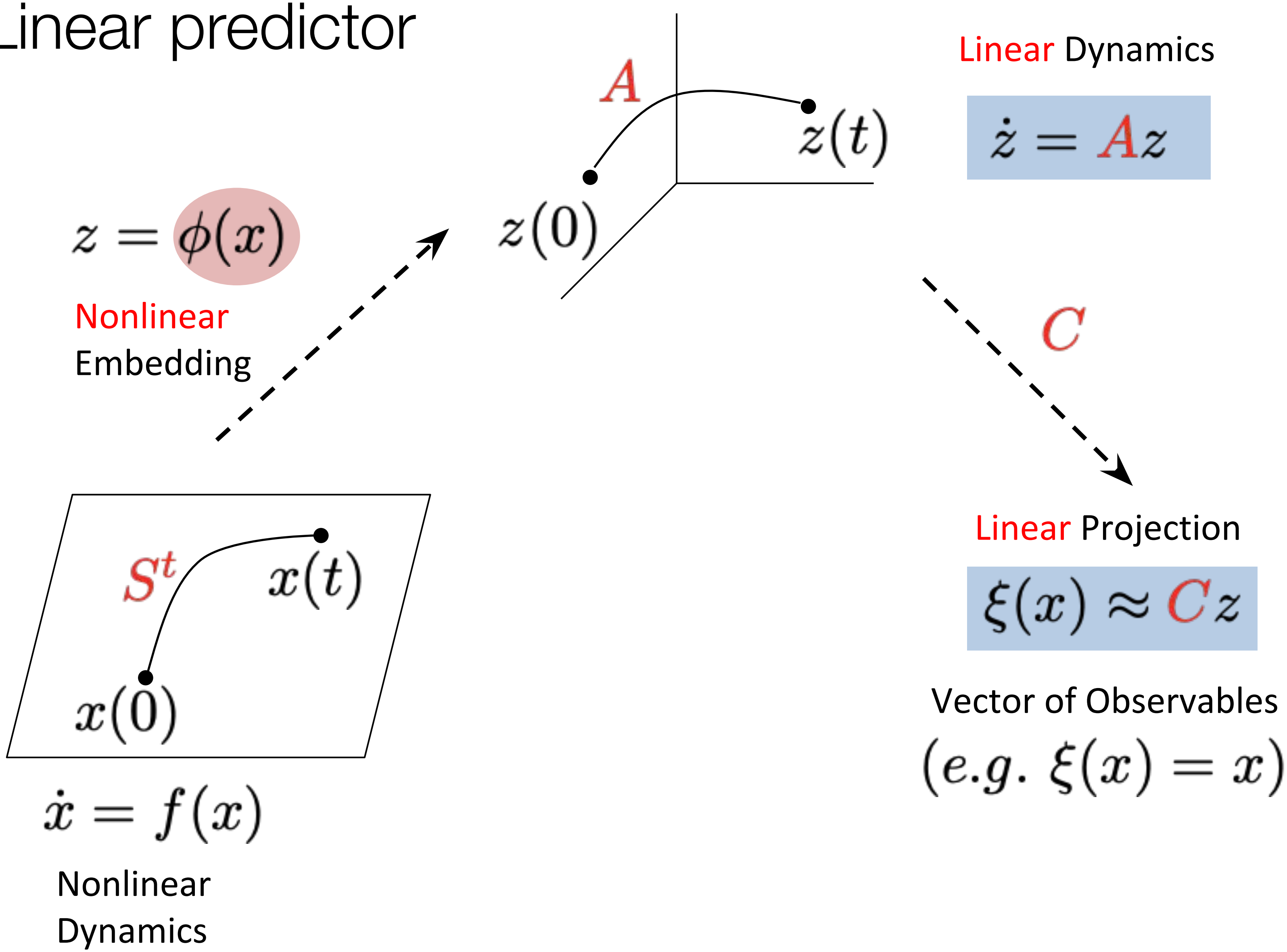
$$\dot{z} = Az$$

$$\xi(x)$$

Vector of Observables

$$(e.g.\ \xi(x) = x)$$

Linear predictor



Why linear predictors?

$$\dot{z} = A z$$
$$z(0) = \phi(x(0))$$
$$\hat{y} = C z$$

$$\hat{y} \approx \xi(x)$$

Why linear predictors?

$$\dot{z} = Az$$
$$z(0) = \phi(x(0))$$
$$\hat{y} = Cz$$

$$\hat{y} \approx \xi(x)$$

Nonlinear feedback control & estimation using linear techniques

Mature & well understood
Fast computation (linear algebra/ convex optimization
Rapid deployment in applications

Why linear predictors?

$$\dot{z} = Az$$
$$z(0) = \phi(x(0))$$
$$\hat{y} = Cz$$

$$\hat{y} \approx \xi(x)$$

Nonlinear feedback control & estimation using linear techniques

Mature & well understood
Fast computation (linear algebra/ convex optimization
Rapid deployment in applications

- Model Predictive control (Korda and Mezic, 2018)
- State Estimation (Surana, Banazuk, 2016)

Choosing the embedding

$$\dot{z} = Az$$
$$z(0) = \phi(x(0))$$
$$\hat{y} = Cz$$

When can we predict exactly?

$$\hat{y} = \xi(x)$$

Choosing the embedding

$$\dot{z} = Az$$
$$z(0) = \phi(x(0))$$
$$\hat{y} = Cz$$

$$\hat{y} = \xi(x)$$

if

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

Choosing the embedding

$$\dot{z} = Az$$
$$z(0) = \phi(x(0))$$
$$\hat{y} = Cz$$

$$\hat{y} = \xi(x)$$

if

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

Choosing the embedding

$$\dot{z} = Az$$
$$z(0) = \phi(x(0))$$
$$\hat{y} = Cz$$

$$\hat{y} = \xi(x)$$

if

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

Learn **rich** set of **eigenfunctions** from data

Eigenfunction construction

Eigenfunction construction

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

Eigenfunction

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

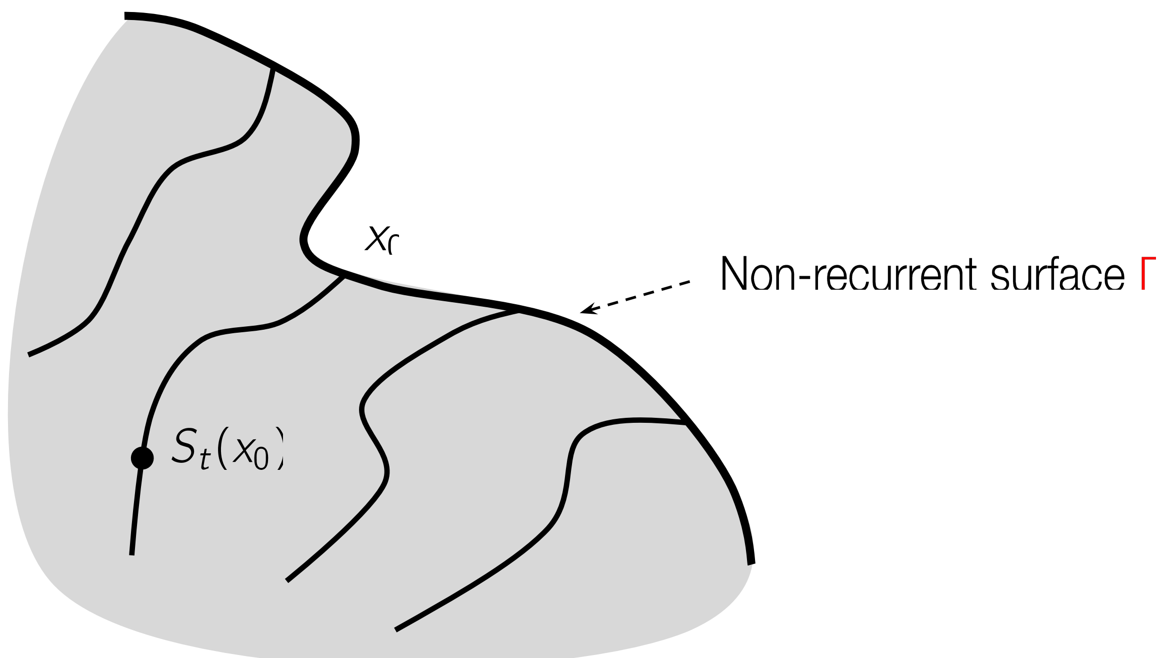
Eigenfunction construction

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

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 ⇒ uncountably many eigenfunctions

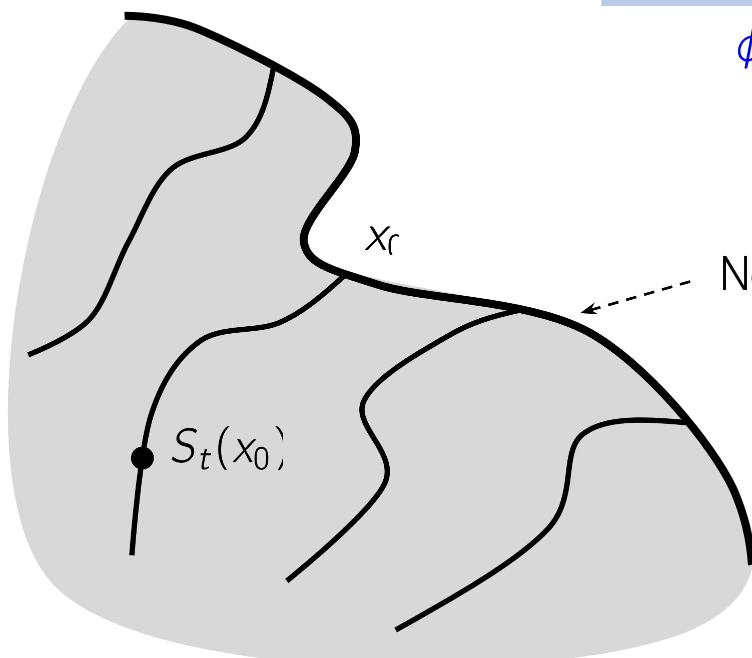
g = arbitrary continuous function

λ = arbitrary complex number

eigenfunction $\phi_{\lambda,g}$

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

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



Eigenfunction construction

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

Eigenfunction

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

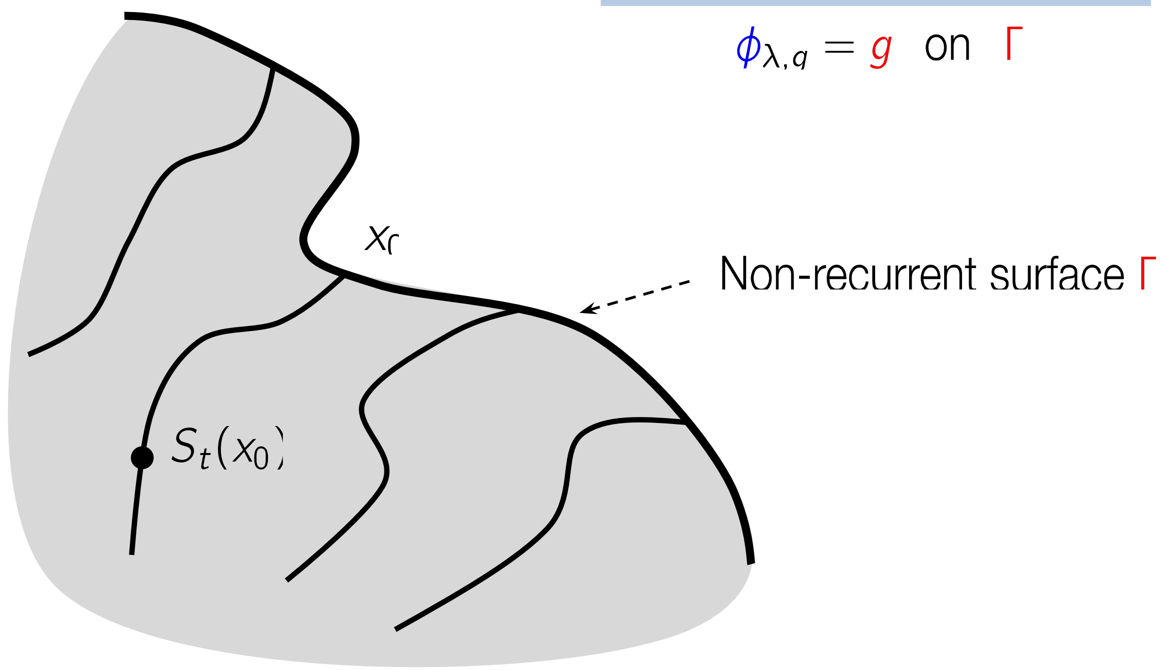
Key observation: Non-recurrent surface $\Rightarrow$ uncountably many eigenfunctions

g = arbitrary continuous function
λ = arbitrary complex number

eigenfunction $\phi_{\lambda,g}$

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

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



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

Eigenfunction construction

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

Eigenfunction

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

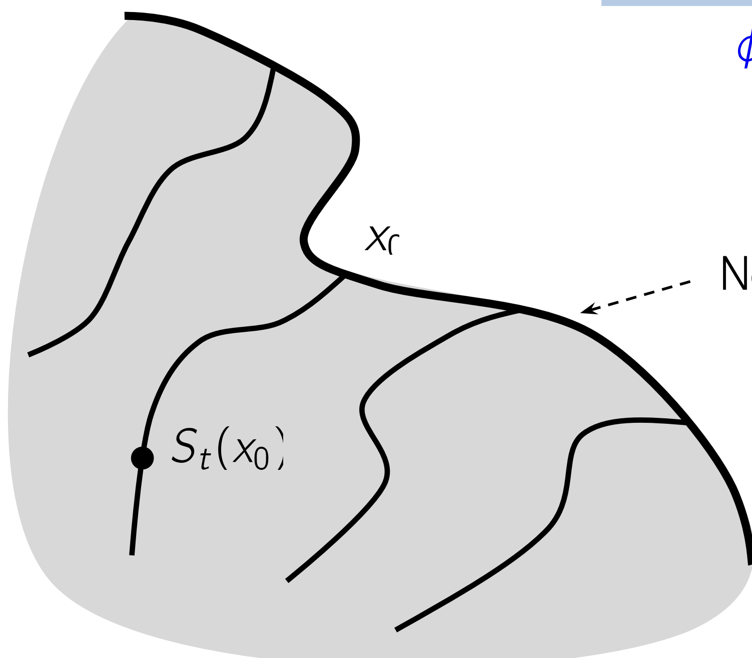
Key observation: Non-recurrent surface ⇒ uncountably many eigenfunctions

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

eigenfunction $\phi_{\lambda,g}$

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

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



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?

$\Lambda =$ 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?

Λ = 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 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 $\mathcal{C}(X_T)$

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 $c_1, \ldots, c_N$

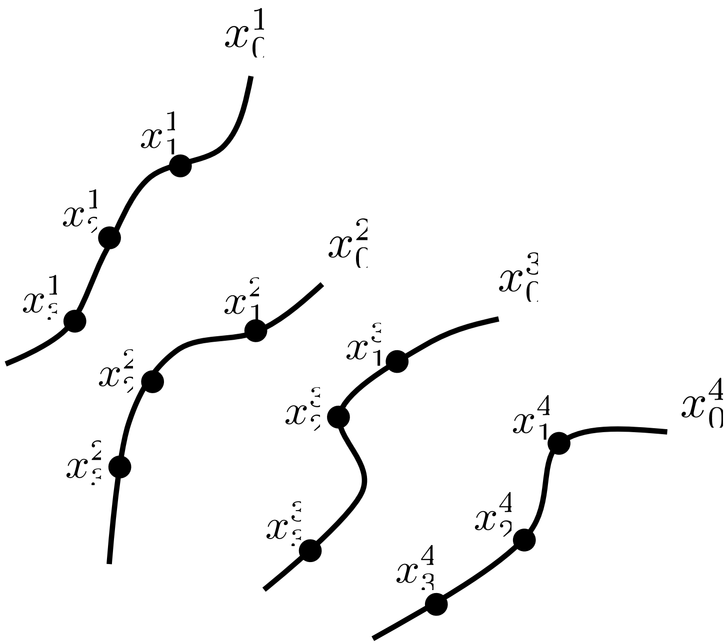
Data-driven construction

Data-driven construction

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

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

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

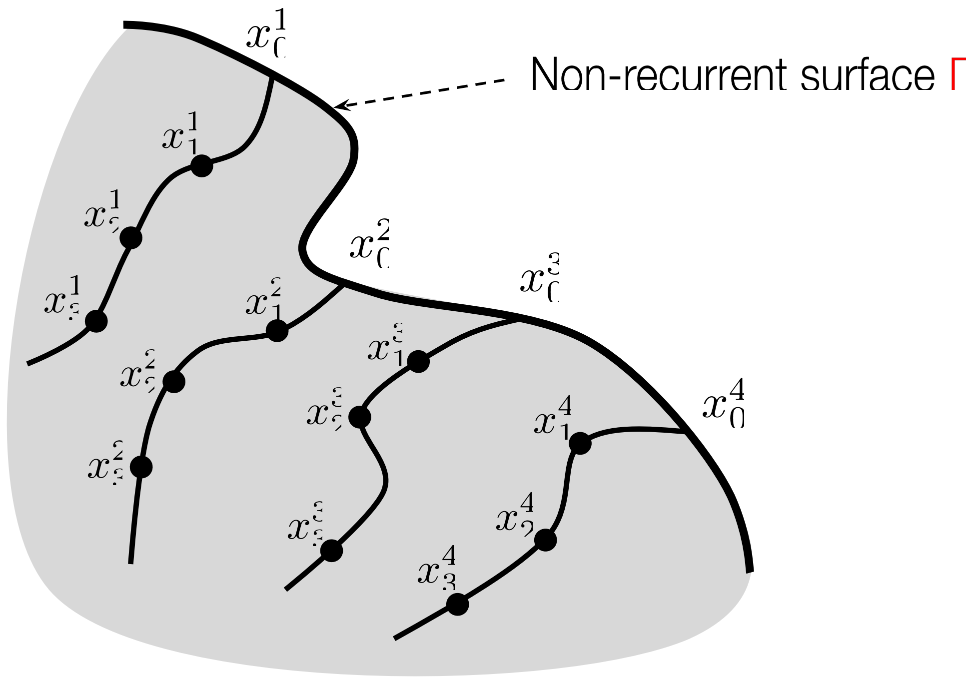


Data-driven construction

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

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

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



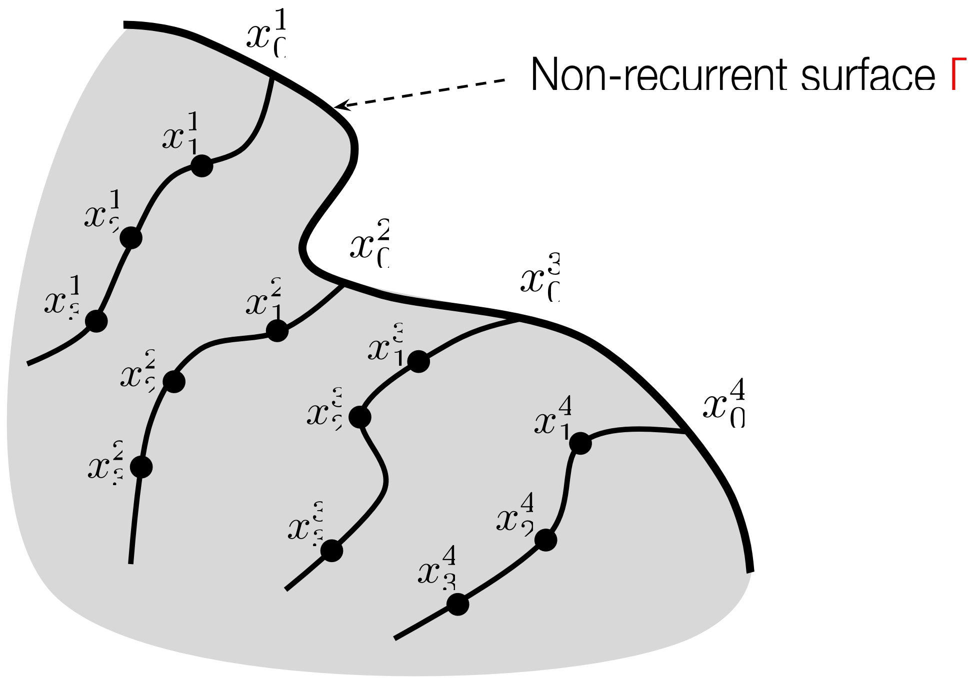
Lemma: Flow rectifiable & initial conditions on distinct trajectories
$\Rightarrow \exists$ non-recurrent surface Γ passing through initial conditions

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 constructior

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)

$$\dot{z} = Az$$
$$z(0) = \phi(x(0))$$
$$\hat{y} = Cz$$

Linear predictor

Goal: Given a data-set (the value of a vector of observables on a training data), predict the value of the vector of observables at a future time.

Given:

$$D = x^j(kT_s) \text{ for } k = \{0, \ldots, M_s\},\ j = \{1, \ldots, M_t\}$$

Predict:

$$\xi(x^j(kT_s)) \text{ for } k > M_s,\ j = \{1, \ldots, M_t\}$$

Construct:

$$z = \phi(x) \qquad \dot{z} = Az \qquad \xi(x) \approx \hat{y}$$

$$z(0) = \phi(x(0))$$

$$\hat{y} = Cz$$

Linear predictor with control

Goal: Given a data-set (the value of a vector of observables on a training data for a controlled system), predict the value of the vector of observables at a future time.

Given:

$$D = [x^j(kT_s), u^j(kT_s)] \text{ for } k = \{0, \ldots, M_s\},\ j = \{1, \ldots, M_t\}$$

Predict:

$$\xi(x^j(kT_s)) \text{ for } k > M_s,\ j = \{1, \ldots, M_t\}$$

Construct:

$$z = \phi(x)$$

$$\dot{z} = Az + Bu$$
$$z(0) = \phi(x(0))$$
$$\hat{y} = Cz$$

$$\xi(x) \approx \hat{y}$$

Adding control

Adding control

$$\dot{z} = Az + Bu$$
$$z(0) = \hat{\phi}(x(0))$$
$$\hat{y} = Cz$$

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

Minimize multi-step prediction error

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

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

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

Adding control

$$\dot{z} = Az + Bu$$
$$z(0) = \hat{\phi}(x(0))$$
$$\hat{y} = Cz$$

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

Minimize multi-step prediction error

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

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

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

&

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

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

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

x

Koopman MPC *[Korda, Mezić 2018]*

Koopman MPC

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

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

x

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

Numerical examples

Numerical examples – Van der Pol

Dynamics

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

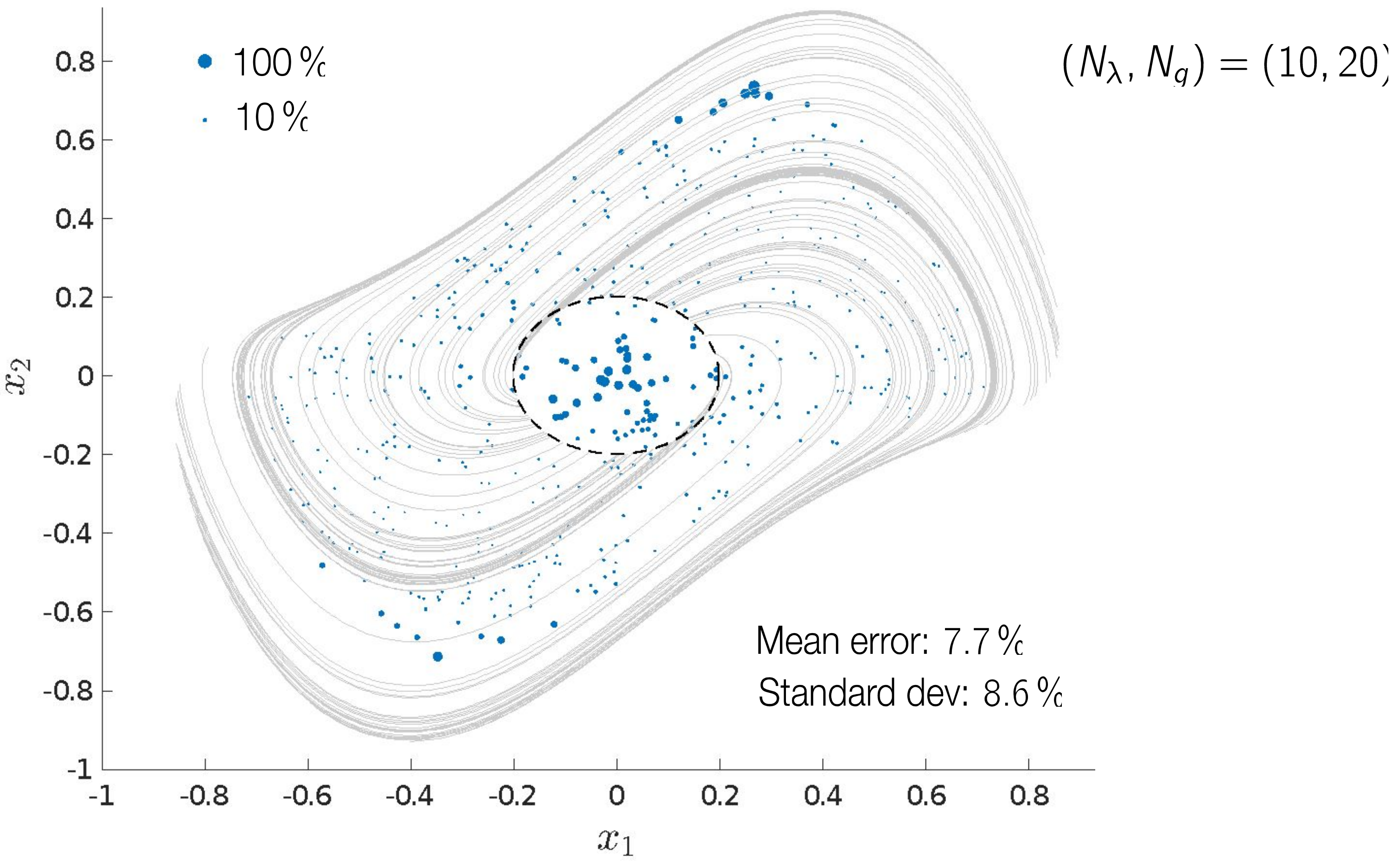
Data: 100 trajectories, 3 second long

Eigenvalues: Mesh from DMD eigenvalues

Boundary functions: Thin plate spline RBFs

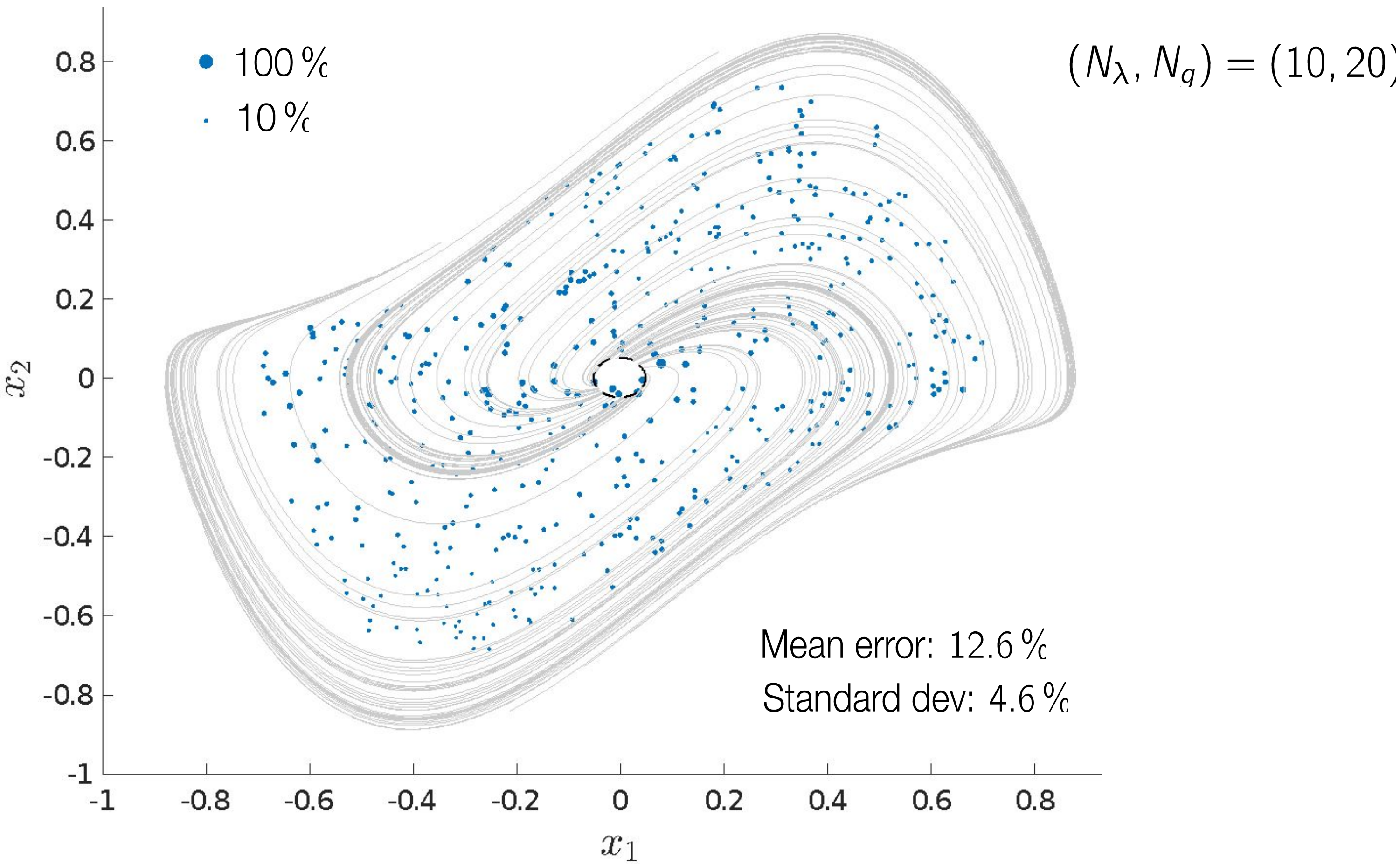
Numerical examples – Van der Pol

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

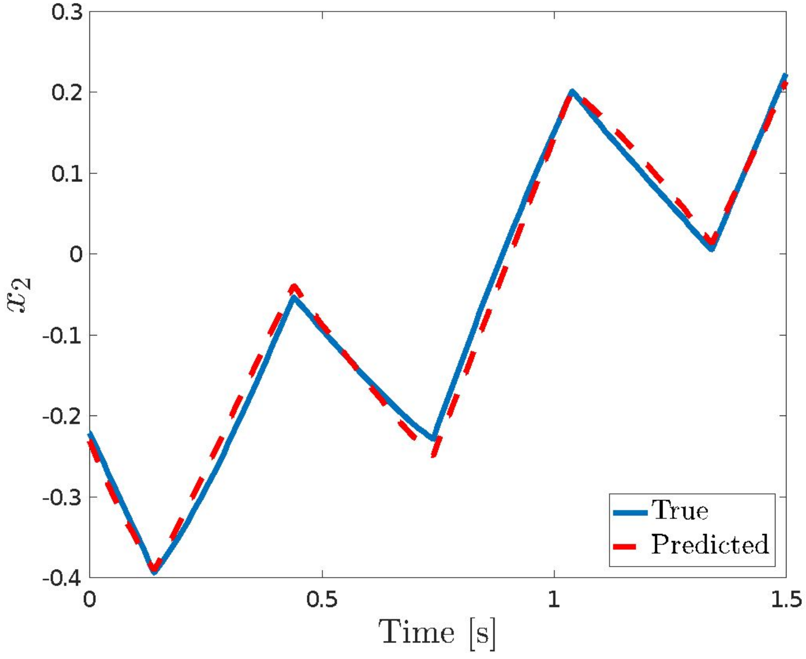


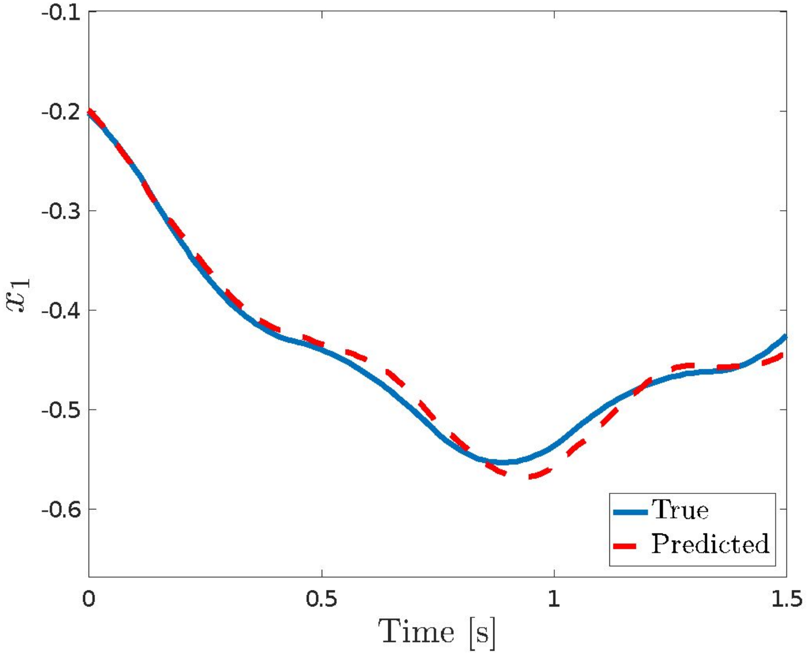
Numerical examples – Van der Pol

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]	$5.0\,\%$	$12.1\,\%$	$9.6\,\%$	$24.9\,\%$	$61.5\,\%$
Mean error [controlled]	$7.7\,\%$	$13.2\,\%$	$12.2\,\%$	$28.4\,\%$	$60.1\,\%$

EDMD error (200 RBF basis functions) = $22.1\,\%$

Numerical examples – damped Duffing

Dynamics

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

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

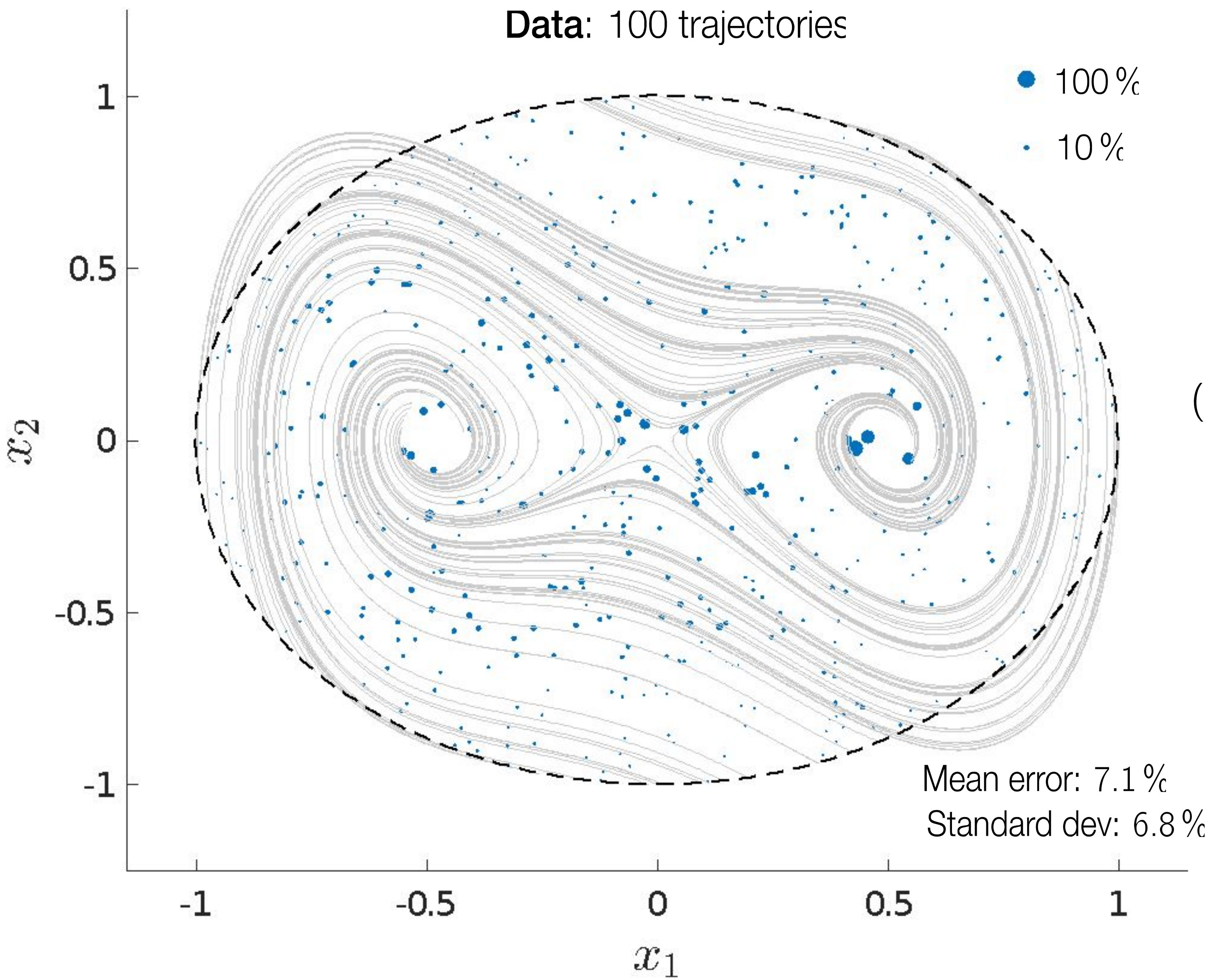
Data: 100 trajectories, 8 second long

Eigenvalues: Mesh from DMD eigenvalues

Boundary functions: Thin plate spline RBFs

Numerical examples – damped Duffing

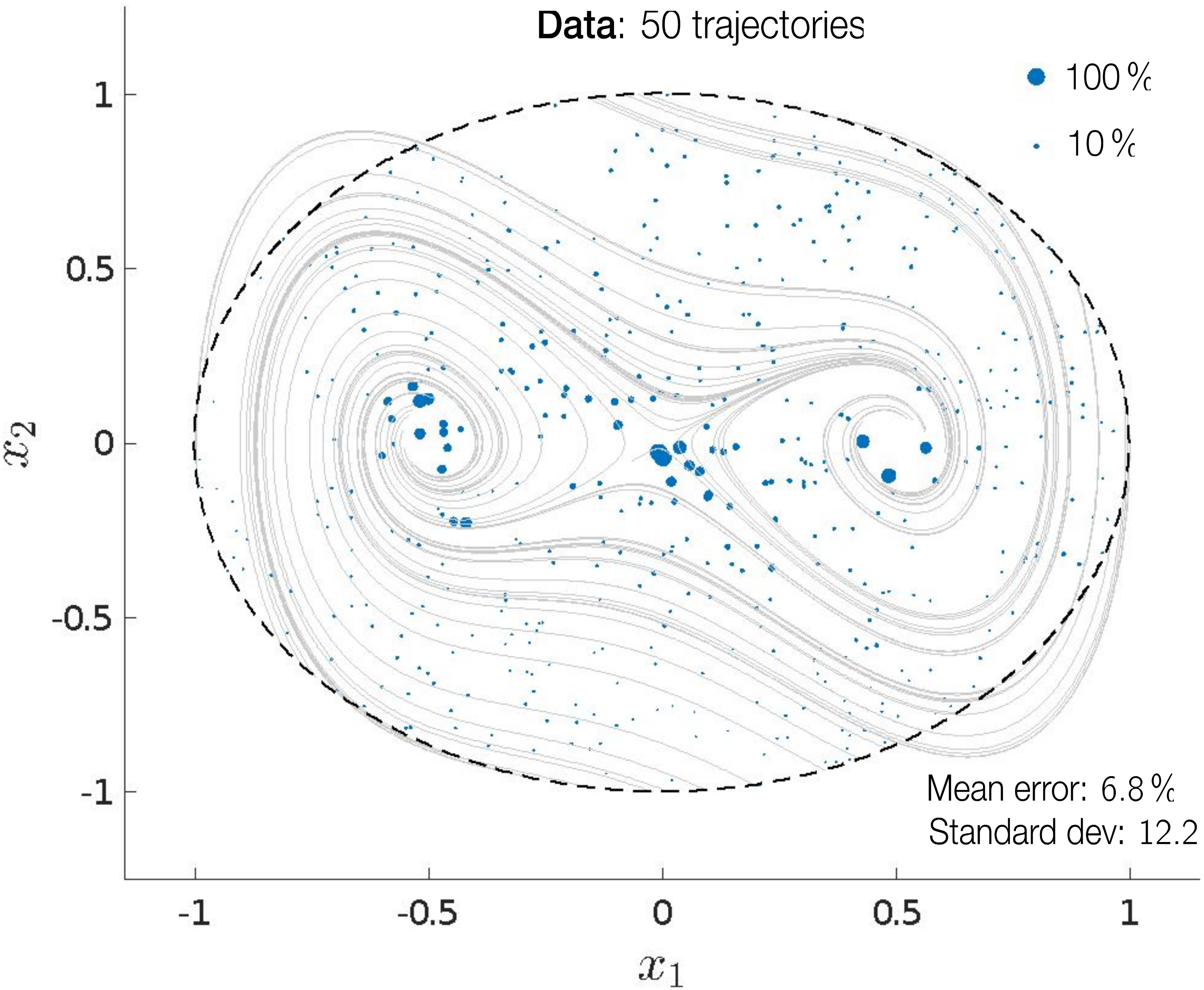
Spatial distribution of one-second prediction error (with control)



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

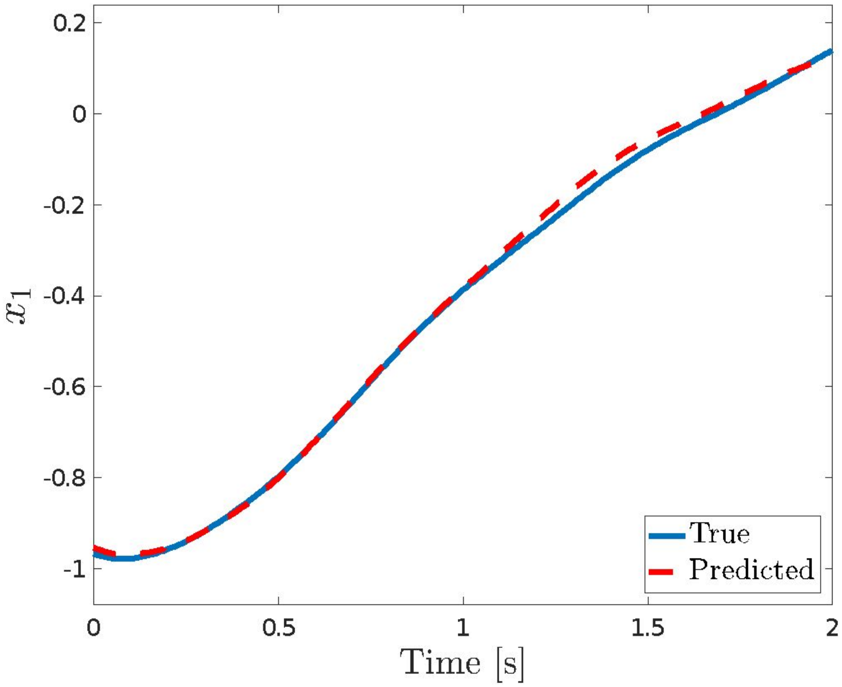
Numerical examples – damped Duffing

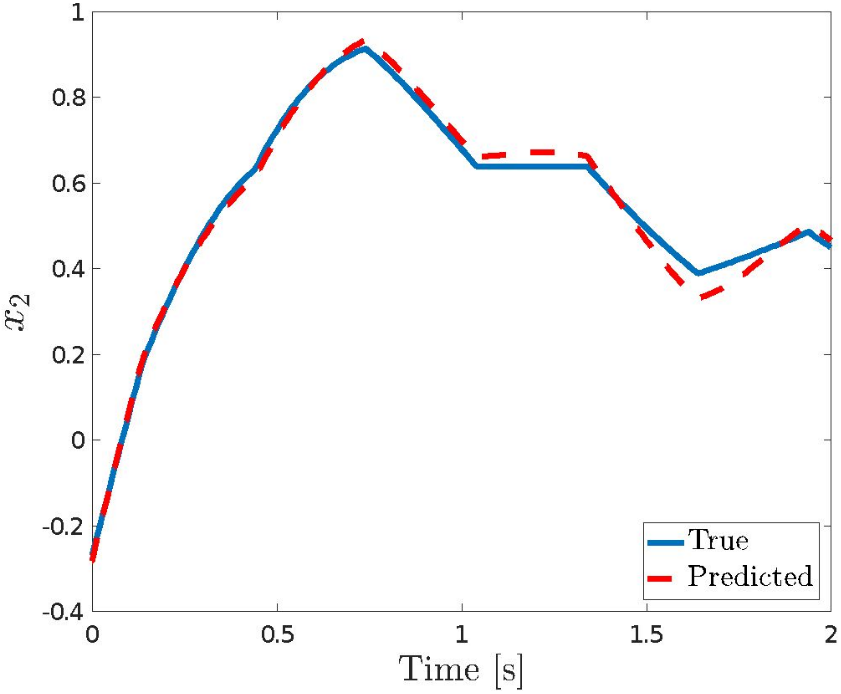
Spatial distribution of one-second prediction error (with control)



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

Numerical examples – damped Duffing





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

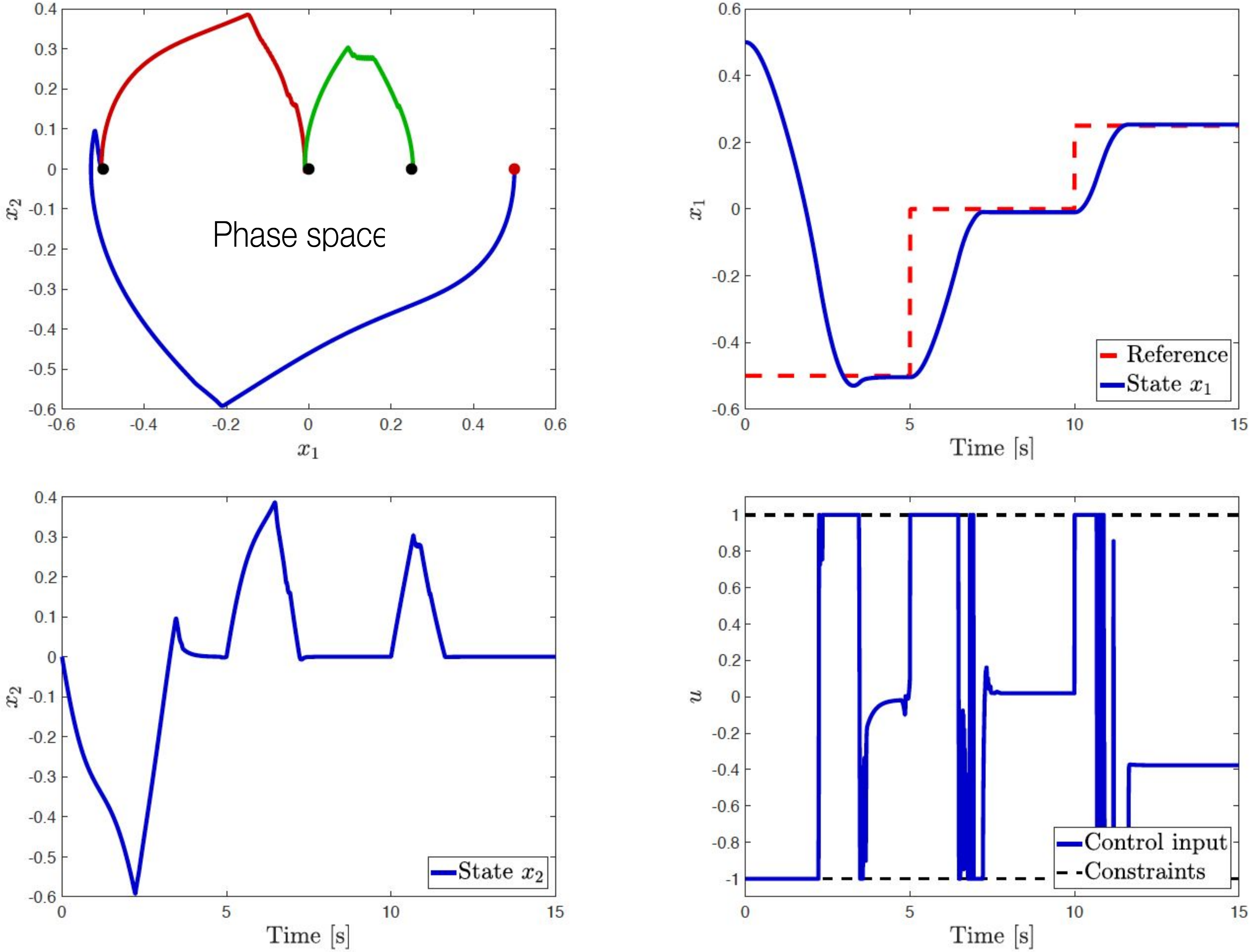
Numerical examples – damped Duffing

(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



Conclusion

Data-driven contruction of Koopman eigenfunctions

- Geared towards transient off-attractor dynamics
- Only linear algebra and/or convex optimization needed
- Readily applicable to control and estimaton
- Very robust
- Can optimally choose the boundary functions

Future work

- High dimensional interpolation/approximation
- Exploit the algebraic structure (products of eigenfunctions)

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

- Generalized eigenfunctions (Jordan blocks)

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