Physics-Enhanced Machine Learning
School of Computation, Information and Technology
Technical University of Munich

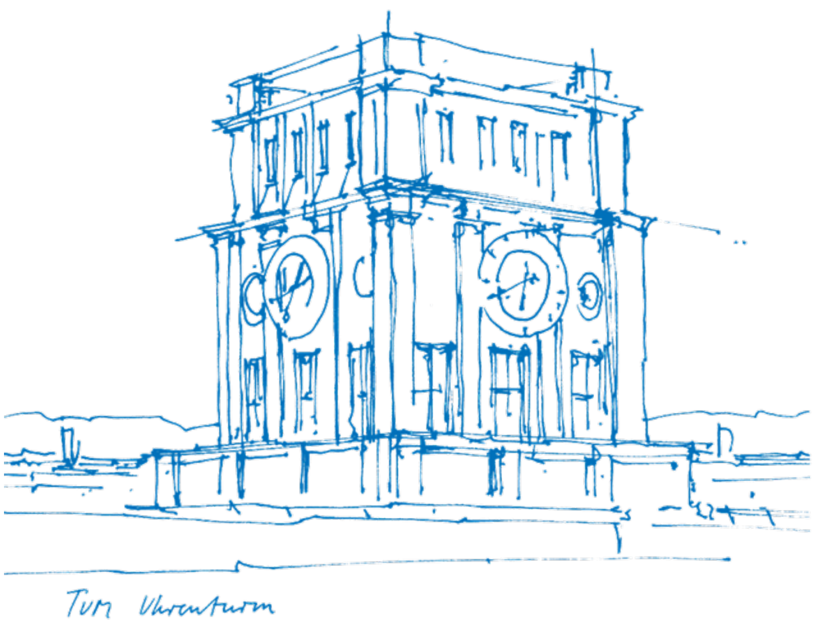
Woudschoten conference, talk #1

Learning dynamical systems from data

Felix Dietrich

Technical University of Munich

2025-09-25



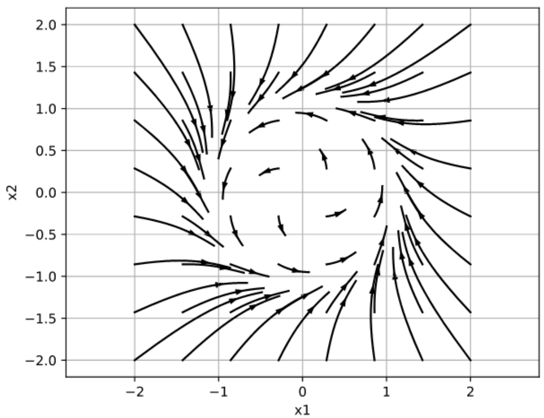
In collaboration with:

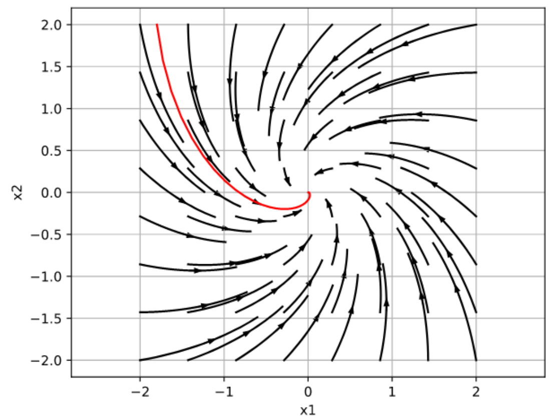
Yannis Kevrekidis
Igor Mezić
Qianxiao Li
Erik Bollt
Ronald Coifman
Hans Bungartz
Christian Mendl
Gerta Köster
Felix Kemeth
Tom Bertalan
Alexei Makeev †
Sebastian Reich
...and many more.

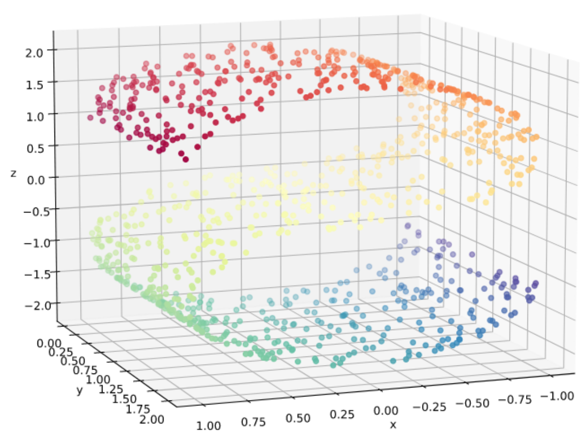
Outline

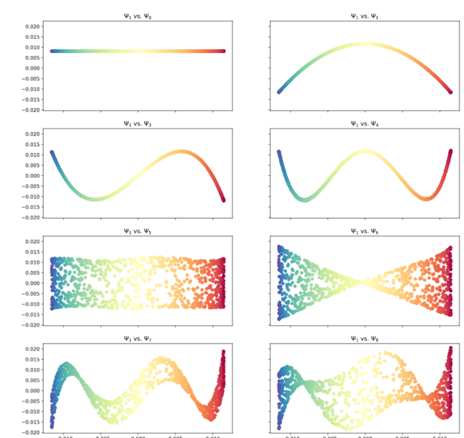
1. Constructing good base spaces
2. System identification with ordinary and partial differential equations
3. System identification with stochastic differential equations
4. Koopman operator framework
5. Open questions, outlook, conclusions

Constructing good base spaces









Laplace-Beltrami operators on manifolds

Geometry of data

$$[\Delta f](x) := \frac{1}{\sqrt{|G|}}\partial_i\left(\sqrt{|G|}G^{ij}\partial_j f\right)$$

The function $f \in C^2(M,\mathbb{R})$ is differentiated twice by $\nabla^2 = \Delta$.
Here: G is the metric tensor of the Riemannian manifold (M, G).

 Coifman & Lafon, 2005, 2006; Berry & Harlim, 2016

Laplace-Beltrami operators on manifolds

Geometry of data

$$[\Delta f](x) := \frac{1}{\sqrt{|G|}}\partial_i\left(\sqrt{|G|}G^{ij}\partial_j f\right)$$

The function $f \in C^2(M,\mathbb{R})$ is differentiated twice by $\nabla^2 = \Delta$.
Here: G is the metric tensor of the Riemannian manifold (M, G).
Benefits:

1. Extension of Laplace operator on Euclidean space: $\Delta f = \sum \partial_i \partial_i f$.
2. Definition is independent of the ambient space and invariant to isometries.
3. Eigenfunctions can be used as low-dimensional embedding coordinates: **Diffusion maps**.
4. Δ captures the entire geometry of (M, G).
5. Direct connection to Diffusion/Heat PDE, $\partial_t f = \Delta f$.
6. Higher-order operators also tractable (deRahm/Hodge): numerical exterior calculus.
7. Locality can be turned into efficient numerical scheme.

Coifman & Lafon, 2005, 2006; Berry & Harlim, 2016

Heat kernel approximation: $\exp(\varepsilon\Delta)$

Diffusion Maps [Coifman and Lafon, 2006]

with several adaptations from Berry, Sauer, Maggioni, ...

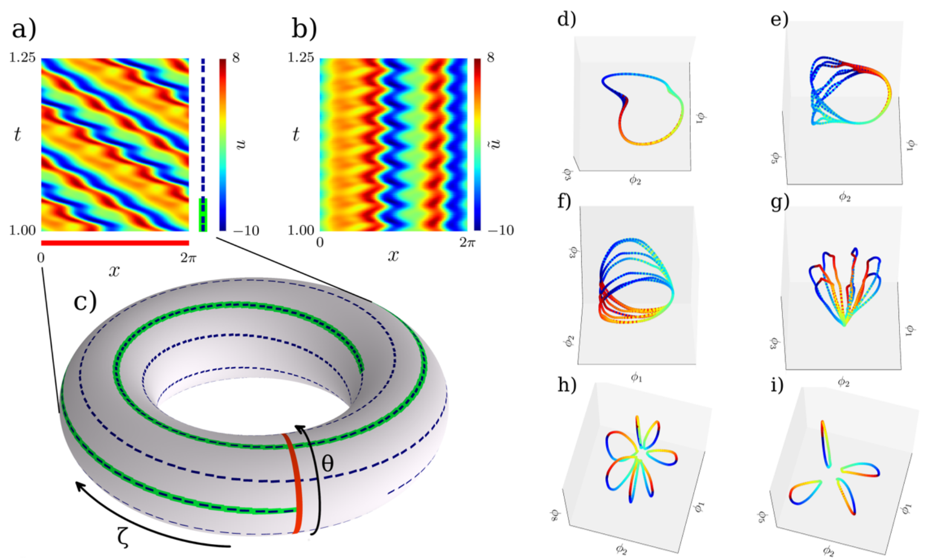
Given a data set $\{y_i \in \mathbb{R}^n\}_{i=1}^N$:

1. Form a distance matrix D with entries
$$D_{ij} = \|y_i - y_j\|,$$
where $i = 1, \ldots, N$ are the rows, $j = 1, \ldots, N$ are the columns, and y_i, y_j are the data points.
2. Form the kernel matrix W with $W_{ij} = \exp\left(-D_{ij}^2/\varepsilon\right)$.
3. Form the diagonal normalization matrix $P_{ii} = \sum_{j=1}^N W_{ij}$.
4. Normalize to form the kernel matrix $K = P^{-1}WP^{-1}$.
5. Form the diagonal normalization matrix $Q_{ii} = \sum_{j=1}^N K_{ij}$.
6. Form the symmetric matrix $\hat{T} = Q^{-1/2}KQ^{-1/2}$.
7. Find the $L+1$ largest eigenvalues a_l and associated eigenvectors v_l of $\hat{T}$.
8. Compute the eigenvalues of $\hat{T}^{1/\varepsilon}$ by $\lambda_l^2 = a_l^{1/\varepsilon}$.
9. Compute the eigenvectors of the matrix $T = Q^{-1}K$ by $\phi_l = Q^{-1/2}v_l$.

Challenges: Steps 1 (distances), 2 (kernel with parameters), 7 (eigenproblem).
[Optional] Other kernels, other distance functions.

Laplace Operator: case study

Constructing good base spaces for dynamical systems (here: PDE)



(a,b) Solutions of Kuramoto-Sivashinsky PDE in quasiperiodic domain (Torus (c) is conceptual).
(d-i) Multi-scale torus reconstruction in eigenfunction space.

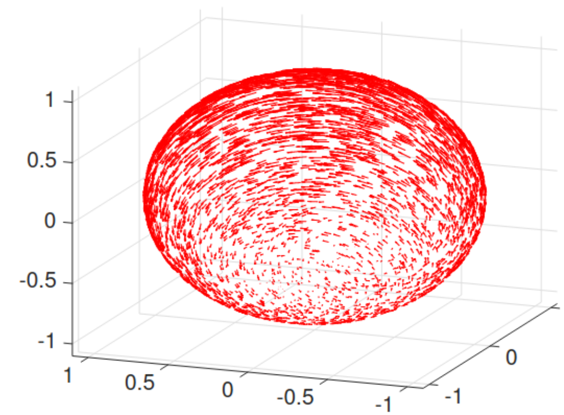
Kemeth, Haugland, D., Bertalan, Hohlein,
Li, Bollt, Talmon, Krischer, Kevrekidis, 2018

Kemeth, Bertalan, Thiem, D., Moon, Laing, Kevrekidis, 2020

Laplace Operator: Spectral exterior calculus

Object	Symbolic	Spectral
Function	f	$\hat{f}_i = \langle \phi_i, f \rangle_{L^2}$
Laplacian	Δf	$\langle \phi_k, \Delta f \rangle_{L^2} = \lambda_k \hat{f}_k$
L^2 Inner Product	$\langle f, h \rangle_{L^2}$	$\sum_i \hat{f}_i^* \hat{h}_i$
Dirichlet Energy	$\langle f, \Delta f \rangle_{L^2} = \int_{\mathcal{M}} \|\mathrm{grad}\, f\|^2 \, d\mu$	$\sum_i \lambda_i \lvert\hat{f}_i\rvert^2$
Multiplication	$\phi_i \phi_j$	$c_{ijk} = \langle \phi_i \phi_j, \phi_k \rangle_{L^2}$
Function Product	fh	$\sum_{ij} c_{kij} \hat{f}_i \hat{h}_j$
Riemannian Metric	$\mathrm{grad}\, \phi_i \cdot \mathrm{grad}\, \phi_j$	$g_{kij} \equiv \langle \mathrm{grad}\, \phi_i \cdot \mathrm{grad}\, \phi_j, \phi_k \rangle_{L^2}$ $= (\lambda_i + \lambda_j - \lambda_k) c_{kij}/2$
Gradient Field	$\mathrm{grad}\, f(h) = \mathrm{grad}\, f^* \cdot \mathrm{grad}\, h$	$\langle \phi_k, \mathrm{grad}\, f(h) \rangle_{L^2} = \sum_{ij} g_{kij} \hat{f}_i \hat{h}_j$
Exterior Derivative	$df(\mathrm{grad}\, h) = df^* \cdot dh$	$\sum_{ij} g_{kij} \hat{f}_i \hat{h}_j$
Vector Field (basis)	$v(f) = v^* \cdot \mathrm{grad}\, f$	$\sum_j v_{ij} \hat{f}_j$
Divergence	$\mathrm{div}\, v$	$\langle \phi_i, \mathrm{div}\, v \rangle_{L^2} = -v_{0i}$
Frame Elements	$b_{ij}(\phi_l) = \phi_i \, \mathrm{grad}\, \phi_j(\phi_l)$	$G_{ijkl} \equiv \langle b_{ij}(\phi_l), \phi_k \rangle_{L^2}$ $= \sum_m c_{mik} g_{mjl}$
Vector Field (frame)	$v(f) = \sum_{ij} v^{ij} b_{ij}(f)$	$\langle \phi_k, v(f) \rangle_{L^2} = \sum_{ijl} G_{ijkl} v^{ij} \hat{f}_l$
Frame Elements	$b^{ij}(v) = b^i \, db^j(v)$	$\langle \phi_k, b^{ij}(v) \rangle_{L^2} = \sum_{nlm} c_{kmi} G_{nlmj} v^{nl}$
1-Forms (frame)	$\omega = \sum_{ij} \omega_{ij} b^{ij}$	$\langle \phi_k, \omega(v) \rangle_{L^2} = \sum_{ij} \omega_{ij} \langle \phi_k, b^{ij}(v) \rangle_{L^2}$

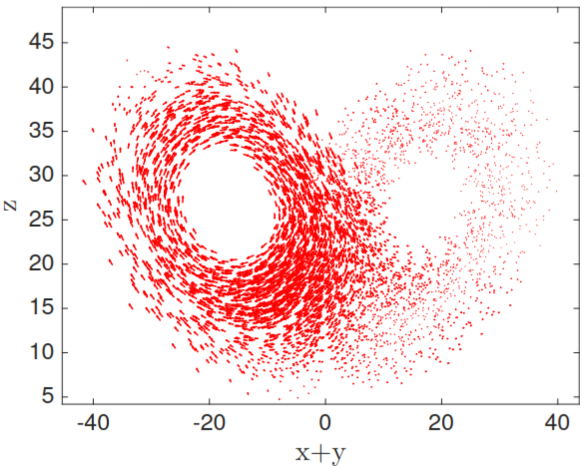
Berry, Tyrus, and Dimitrios Giannakis. 2020. “Spectral Exterior Calculus.” Communications on Pure and Applied Mathematics 73 (4): 689-770. https://doi.org/10.1002/cpa.21885.



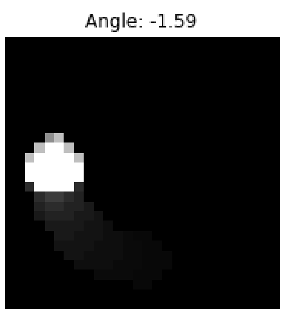
Laplace Operator: Spectral exterior calculus

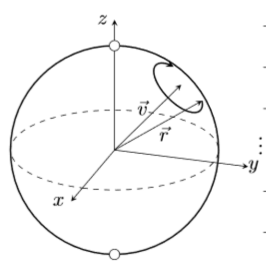
Object	Symbolic	Spectral
Function	f	$\hat{f}_i = \langle \phi_i, f \rangle_{L^2}$
Laplacian	Δf	$\langle \phi_k, \Delta f \rangle_{L^2} = \lambda_k \hat{f}_k$
L^2 Inner Product	$\langle f, h \rangle_{L^2}$	$\sum_i \hat{f}_i^* \hat{h}_i$
Dirichlet Energy	$\langle f, \Delta f \rangle_{L^2} = \int_{\mathscr{M}} \lVert \operatorname{grad} f \rVert^2 \, d\mu$	$\sum_i \lambda_i \lvert \hat{f}_i \rvert^2$
Multiplication	$\phi_i \phi_j$	$c_{ijk} = \langle \phi_i \phi_j, \phi_k \rangle_{L^2}$
Function Product	fh	$\sum_{ij} c_{kij} \hat{f}_i \hat{h}_j$
Riemannian Metric	$\operatorname{grad} \phi_i \cdot \operatorname{grad} \phi_j$	$g_{kij} \equiv \langle \operatorname{grad} \phi_i \cdot \operatorname{grad} \phi_j, \phi_k \rangle_{L^2}$ $= (\lambda_i + \lambda_j - \lambda_k) c_{kij}/2$
Gradient Field	$\operatorname{grad} f(h) = \operatorname{grad} f^* \cdot \operatorname{grad} h$	$\langle \phi_k, \operatorname{grad} f(h) \rangle_{L^2} = \sum_{ij} g_{kij} \hat{f}_i \hat{h}_j$
Exterior Derivative	$df(\operatorname{grad} h) = df^* \cdot dh$	$\sum_{ij} g_{kij} \hat{f}_i \hat{h}_j$
Vector Field (basis)	$v(f) = v^* \cdot \operatorname{grad} f$	$\sum_j v_{ij} \hat{f}_j$
Divergence	$\operatorname{div} v$	$\langle \phi_i, \operatorname{div} v \rangle_{L^2} = -v_{0i}$
Frame Elements	$b_{ij}(\phi_l) = \phi_i \operatorname{grad} \phi_j(\phi_l)$	$G_{ijkl} \equiv \langle b_{ij}(\phi_l), \phi_k \rangle_{L^2}$ $= \sum_m c_{mik} g_{mjl}$
Vector Field (frame)	$v(f) = \sum_{ij} v^{ij} b_{ij}(f)$	$\langle \phi_k, v(f) \rangle_{L^2} = \sum_{ijl} G_{ijkl} v^{ij} \hat{f}_l$
Frame Elements	$b^{ij}(v) = b^i \, db^j(v)$	$\langle \phi_k, b^{ij}(v) \rangle_{L^2} = \sum_{nlm} c_{kmi} G_{nlmj} v^{nl}$
1-Forms (frame)	$\omega = \sum_{ij} \omega_{ij} b^{ij}$	$\langle \phi_k, \omega(v) \rangle_{L^2} = \sum_{ij} \omega_{ij} \langle \phi_k, b^{ij}(v) \rangle_{L^2}$

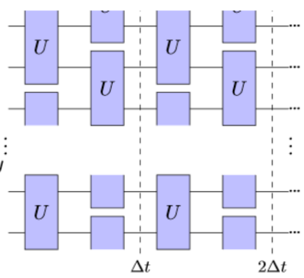
Berry, Tyrus, and Dimitrios Giannakis. 2020. "Spectral Exterior Calculus." Communications on Pure and Applied Mathematics 73 (4): 689-770. https://doi.org/10.1002/cpa.21885.

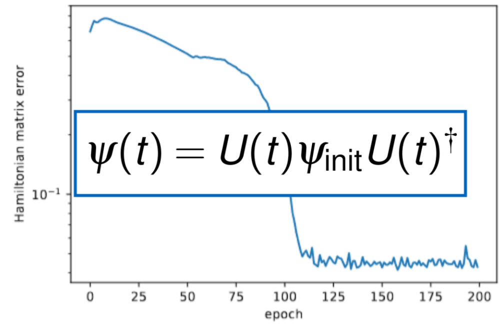


System identification with ordinary and partial differential equations









$$\psi(t) = U(t)\psi_{\text{init}}U(t)^\dagger$$

System identification with ODE

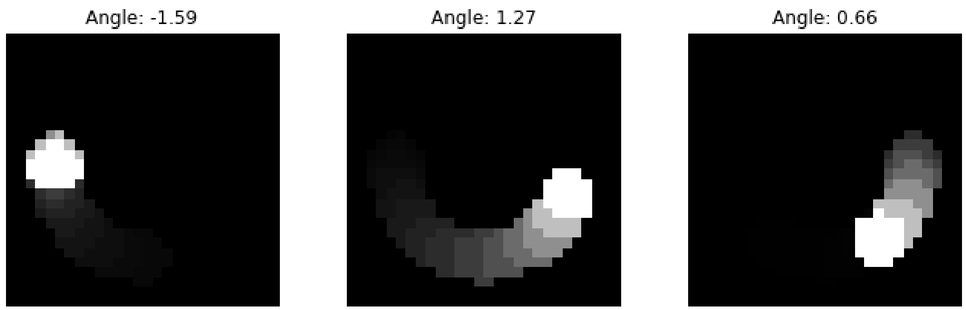
Equations with one derivative of the unknown

$$\frac{d}{dt}x(t) = f(x(t)).$$

Problem statement

Given: time series $\{x(t)\}$ of observations.

Goal: identify the function f.



System identification with ODE

Equations with one derivative of the unknown

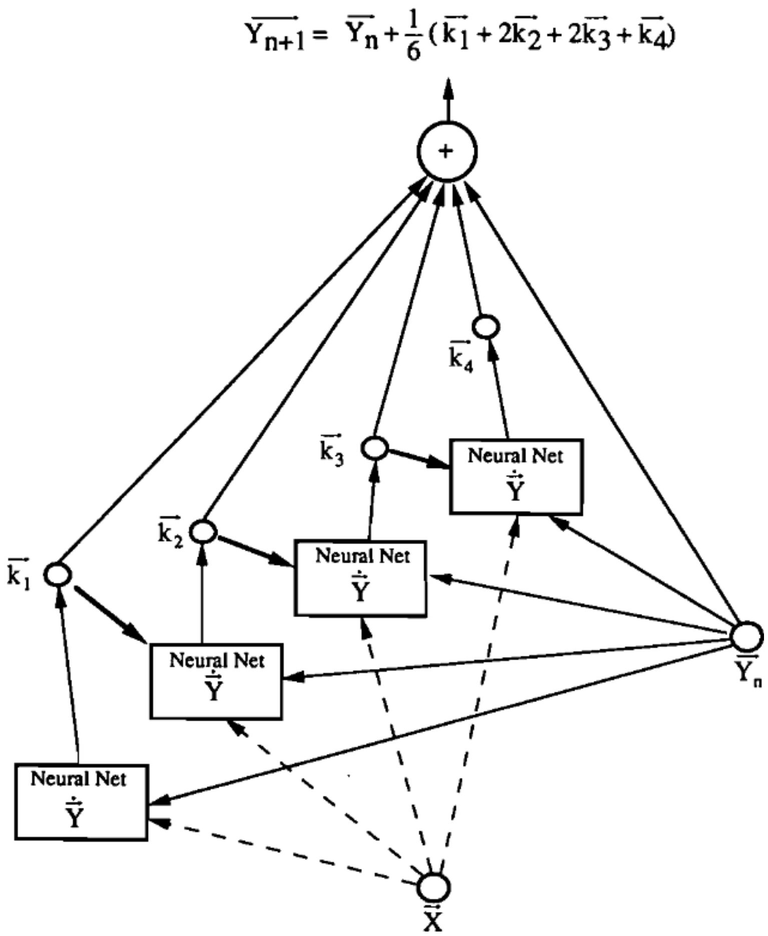
$\frac{d}{dt}x(t) = f(x(t)) \approx \sigma(Wx(t)+b)$.

Work before the last AI winter [1,2,3]

[1] González-García, Rico-Martínez, and Kevrekidis. 1998. "Identification of Distributed Parameter Systems: A Neural Net Based Approach." Computers & Chemical Engineering. https://doi.org/10.1016/s0098-1354(98)00191-4.

[2] Rico-Martinez, Anderson, and Kevrekidis. 1994. "Continuous-Time Nonlinear Signal Processing: A Neural Network Based Approach for Gray Box Identification." Proc. of IEEE Workshop on Neural Networks for Signal Processing. https://doi.org/10.1109/nnsp.1994.366006.

[3] Rico-Martínez, Krischer, Kevrekidis, Kube, and Hudson. 1992. "Discrete- vs. Continuous-Time Nonlinear Signal Processing of Cu Electrodissolution Data." Chemical Engineering Communications. https://doi.org/10.1080/00986449208936084.



Identification of Hamiltonian dynamics

Problem statement

Given: time series $\{p(t), q(t)\}$ of observations of a Hamiltonian system.

Goal: identify the Hamiltonian function $H(p, q)$, so that

$$\frac{d}{dt}(p,q) = J \cdot \nabla H(p,q),\ J := \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

Identification of Hamiltonian dynamics

Problem statement

Given: time series $\{p(t), q(t)\}$ of observations of a Hamiltonian system.
Goal: identify the Hamiltonian function $H(p,q)$, so that

$$\frac{d}{dt}(p,q) = J \cdot \nabla H(p,q),\ J := \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

Main idea, neural networks

1. Parameterize H as neural network $\hat{H}$.
2. Approximate time derivatives from time series with finite-differences.
3. **Train network** to minimize $\|\frac{d}{dt}(p,q) - J \cdot \nabla \hat{H}(p,q)\|^2$.

Identification of Hamiltonian dynamics

Problem statement

Given: time series $\{p(t), q(t)\}$ of observations of a Hamiltonian system.
Goal: identify the Hamiltonian function $H(p,q)$, so that

$$\frac{d}{dt}(p,q) = J \cdot \nabla H(p,q),\ J := \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

Main idea, neural networks

1. Parameterize H as neural network $\hat{H}$.
2. Approximate time derivatives from time series with finite-differences.
3. **Train network** to minimize $\|\frac{d}{dt}(p,q) - J \cdot \nabla \hat{H}(p,q)\|^2$.

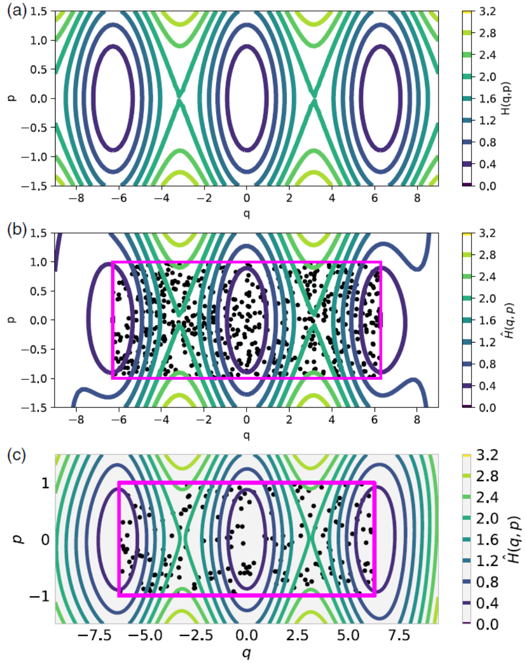
Main idea, Gaussian processes

1. Use Gaussian process $\hat{H}$ as ansatz for unknown Hamiltonian: $\hat{H} = k^* K^{-1} H$.
2. Approximate time derivatives from time series with finite-differences.
3. **Solve linear PDE** through least squares: $J \cdot \nabla \hat{H} = \underbrace{J \cdot \nabla k^* K^{-1}}_{A} \underbrace{H}_{x} = \underbrace{\frac{d}{dt}(p,q)}_{b}$.

Identification of Hamiltonian dynamics

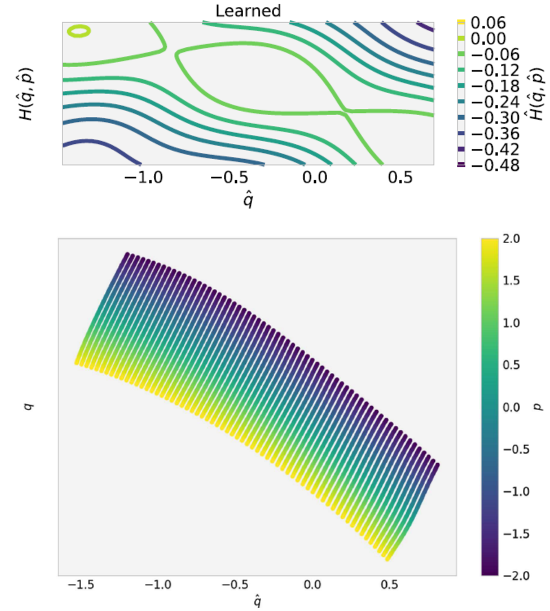
Examples

Non-linear pendulum

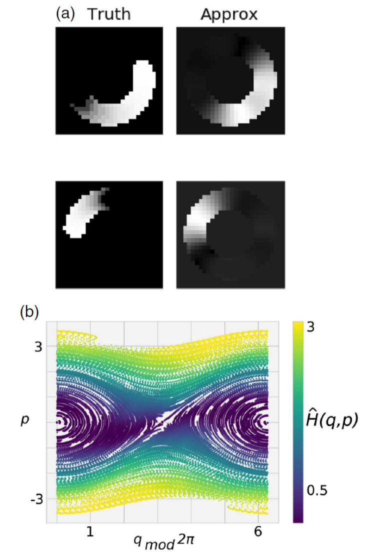


Data points: ANN 20,000; GP 625

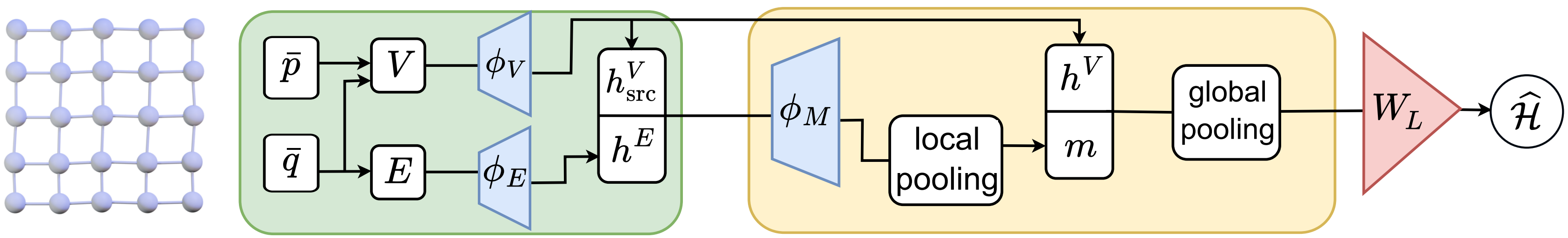
Hamiltonian systems in latent spaces



Identification in high-dimensional ambient spaces (image data)



Learning Hamiltonian dynamics on graphs



Hamiltonian **graph neural network** constructed with random feature layers. The network solves a linear PDE defined on a high-dimensional base space (all nodes of the graph combined) for H, so that

$$\dot{q} = \frac{\partial H}{\partial p}, \; \dot{p} = -\frac{\partial H}{\partial q}.$$

Rahma, Datar, Cukarska, and D. “Rapid Training of Hamiltonian Graph Networks without Gradient Descent.” Preprint, `http://arxiv.org/abs/2506.06558`.

More on this in my talk tomorrow (Friday, 9:00)!

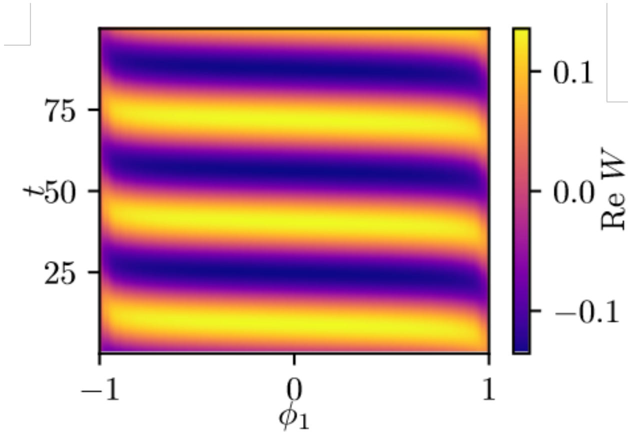
System identification with PDE

Equations with more than one derivative of the unknown

$$\frac{d}{dt}x(t,s) = f(x(t,s), \frac{d}{ds}x(t,s)).$$

Problem statement

Given: time series $\{x(t,s)\}$ of observations.
Goal: identify the function f.



Identification of PDE for oscillator systems

Problem statement

Given: time series $\{w(t)\}$ of observations of a system of oscillators.

Goal: identify a good base space **and** the PDE on it.

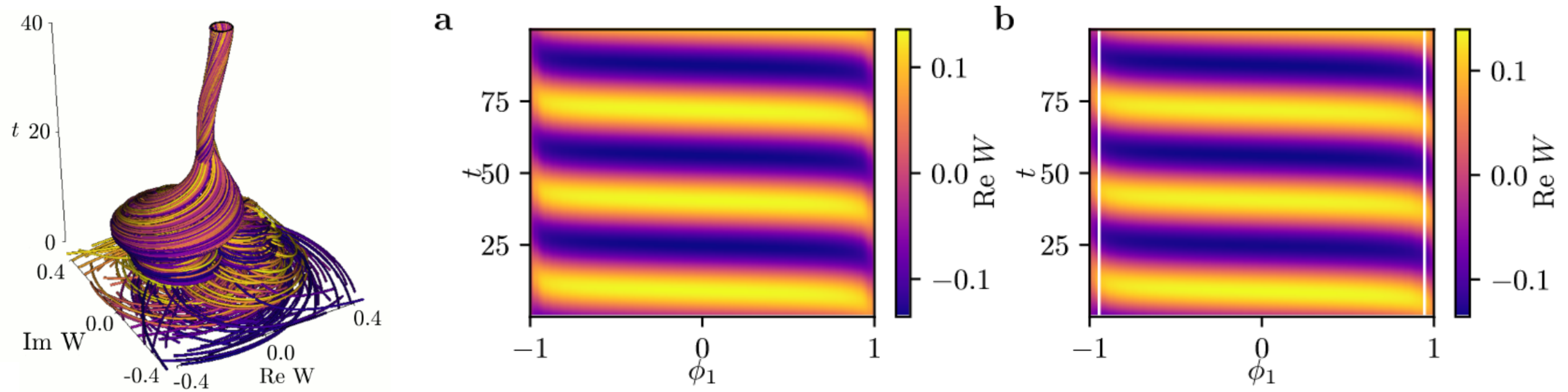
Identification of PDE for oscillator systems

Problem statement

Given: time series $\{w(t)\}$ of observations of a system of oscillators.
Goal: identify a good base space **and** the PDE on it.

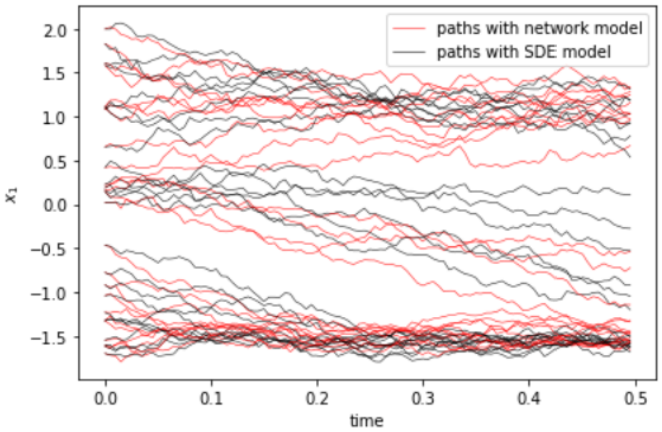
Main idea

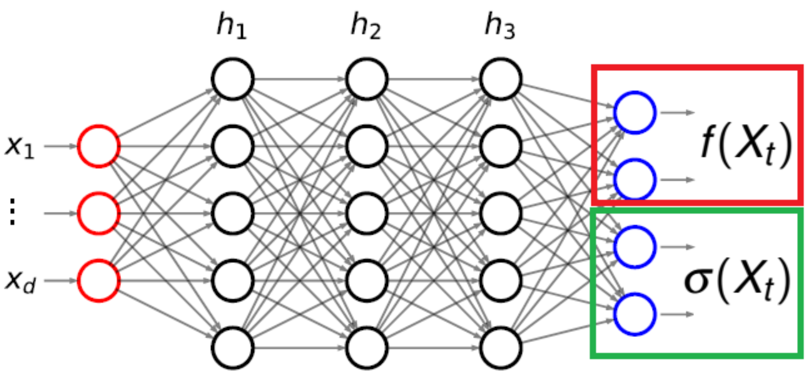
1. Parameterize short, individual oscillator **trajectories** with ϕ.
2. Observe how the oscillators "move" on the new space: $\frac{d}{dt}w(t,\phi) = f(w, \frac{d}{d\phi}w, \frac{d^2}{d\phi^2}w, \frac{d^3}{d\phi^3}w)$.
3. Estimate the function f.



Left: observations $w(t)$. Center: Re$[w(t)]$ on new coordinate ϕ. Right: integrated, learned PDE.

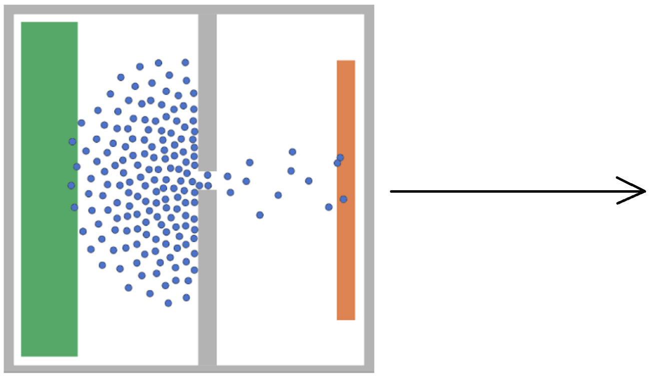
System identification with stochastic differential equations

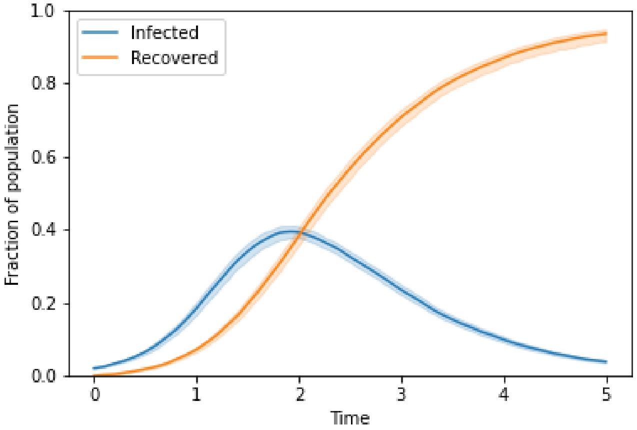




Identify SDE to coarse-grain particle dynamics

1. Coarse-graining particle dynamics
2. Identifying SDE for coarse observables
3. Here: Coarse-graining agent-based models: local infection models





From particle dynamics to coarse-grained SDE.

Introduction: SDE

Stochastic ordinary differential equations (SDE)?

$$dX_t = f(X_t)dt + \sigma(X_t)dW_t$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$, $\sigma : \mathbb{R}^n \to \mathrm{SPD}(n) \subset \mathbb{R}^{n \times n}$ are **drift** and **diffusivity**, respectively, W_t are n independent Wiener processs, with $W_t - W_s \sim \mathcal{N}(0, t-s)$.

Example: double-well potential

Let $f(x) = -(4x^3 - 8x + 3)/2$, and $\sigma(x) = (0.1x + 1)/2$.

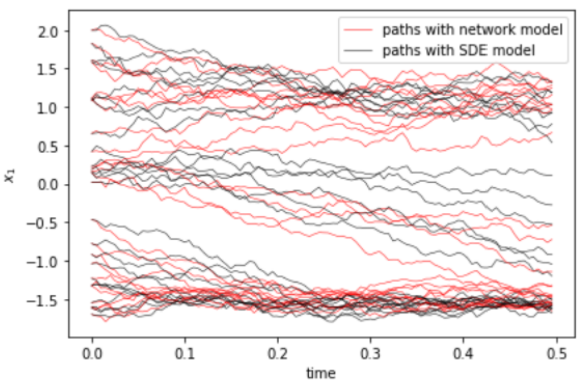


Figure: Sample paths from the SDE and the network.

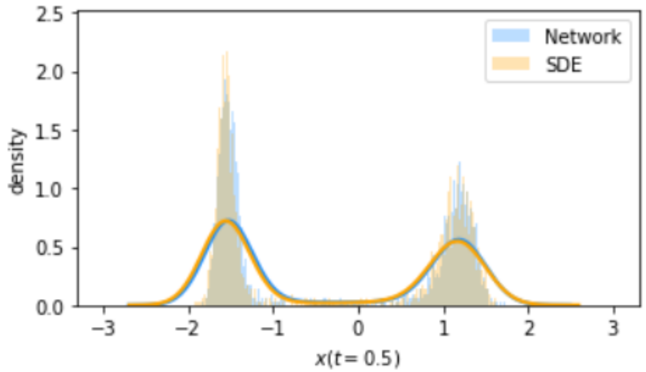


Figure: Density at $t = 0.5$.

Our approach to identify SDE

Problem statement

Data: snapshots $(x(t), x(t+\Delta t))$ all over the state space, possibly with different Δt for each snapshot.
Goal: approximate drift f and diffusivity σ.
Challenge: We **do not have** (a) long time series or (b) constant time steps.
Assumptions:
(a) diffusivity is an SPD matrix everywhere;
(b) drift and diffusivity are continuous w.r.t. the input;
(c) our dataset samples the state space well.

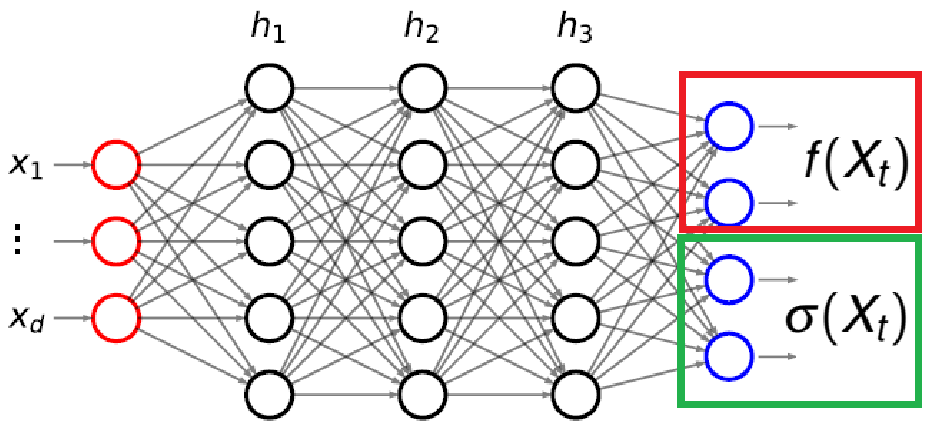


Figure: Network architecture to learn drift and (diagonal) diffusivity.

D., Makeev, Kevrekidis, Evangelou, Bertalan,
Reich, Kevrekidis, Chaos, 2022; pre-print: arXiv:2106.09004

Our approach to identify SDE

Main idea

We assume the data is generated by the SDE:

$$dX_t = f(X_t)dt + \sigma(X_t)dW_t$$

This SDE can be approximated with the Euler-Maruyama scheme:

$$X(t+\Delta t) - X(t) = f(X(t))\Delta t + \sigma(X(t))\xi,\ \xi \sim \mathcal{N}(0, \Delta t)$$

D., Makeev, Kevrekidis, Evangelou, Bertalan, Reich, Kevrekidis, Chaos, 2022; pre-print: arXiv:2106.09004

Our approach to identify SDE

Main idea

We assume the data is generated by the SDE:

$$dX_t = f(X_t)dt + \sigma(X_t)dW_t$$

This SDE can be approximated with the Euler-Maruyama scheme:

$$X(t+\Delta t) - X(t) = f(X(t))\Delta t + \sigma(X(t))\xi,\ \xi \sim \mathcal{N}(0, \Delta t)$$

Therefore: We can assume that

$$X(t+\Delta t) \sim \mathcal{N}\left(X(t) + f(X(t))\Delta t, \sigma(X(t))^2 \Delta t\right)$$

If $p : \mathbb{R}^n \to \mathbb{R}^+$ is the probability density of this normal distribution, and we set f_θ, σ_θ to be our neural network, then

$$\theta := \arg\max_{\xi} \mathbb{E}\left[\log p_\xi\left(X_{k+1}|X_k\right)\right] \approx \arg\max_{\xi}\left[\frac{1}{N}\sum_{i=1}^{N} \log p_\xi\left(X_{k+1}^{(i)}|X_k^{(i)}\right)\right].$$

Since we assume normality, $\log p_\xi$ has a simple formula.

D., Makeev, Kevrekidis, Evangelou, Bertalan,
Reich, Kevrekidis, Chaos, 2022; pre-print: arXiv:2106.09004

Our approach to identify SDE

Main idea

We assume the data is generated by the SDE:

$$dX_t = f(X_t)dt + \sigma(X_t)dW_t$$

This SDE can be approximated with the Euler-Maruyama scheme:

$$X(t+\Delta t) - X(t) = f(X(t))\Delta t + \sigma(X(t))\xi,\ \xi \sim \mathcal{N}(0, \Delta t)$$

Therefore: We can assume that

$$X(t+\Delta t) \sim \mathcal{N}\left(X(t) + f(X(t))\Delta t, \sigma(X(t))^2 \Delta t\right)$$

If $p : \mathbb{R}^n \to \mathbb{R}^+$ is the probability density of this normal distribution, and we set f_θ, σ_θ to be our neural network, then

$$\theta := \arg\max_\xi \mathbb{E}\left[\log p_\xi\left(X_{k+1}|X_k\right)\right] \approx \arg\max_\xi \left[\frac{1}{N}\sum_{i=1}^{N} \log p_\xi\left(X_{k+1}^{(i)}|X_k^{(i)}\right)\right].$$

Since we assume normality, $\log p_\xi$ has a simple formula.

This is just log marginal likelihood maximization!

D., Makeev, Kevrekidis, Evangelou, Bertalan, Reich, Kevrekidis, Chaos, 2022; pre-print: arXiv:2106.09004

Results in paper

What about other integrators?

The SDE can be approximated with the Euler-Maruyama scheme:

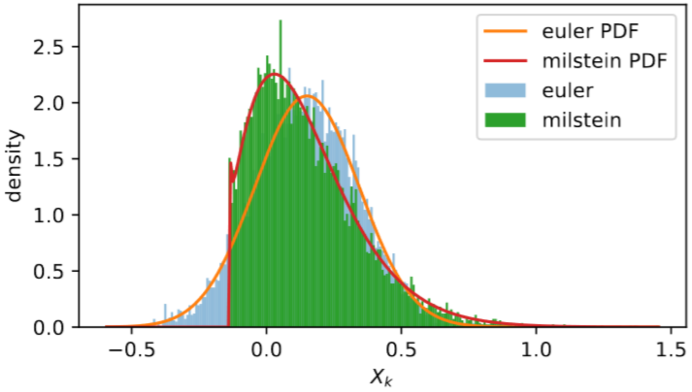
$$X(t+\Delta t) = X + f(X)\Delta t + \sigma(X)\xi,\ \xi \sim \mathcal{N}(0,\Delta t),$$

...or with the Milstein scheme:

$$X(t+\Delta t) = X + f(X)\Delta t + \sigma(X)\xi + \frac{1}{2}\sigma(X)\frac{d}{dx}\sigma(X)\left(\xi^2 - \Delta t\right),\ \xi \sim \mathcal{N}(0,\Delta t)$$

...or any other (good) numerical integration scheme:

$$X(t+\Delta t) = \phi\left(X; \Delta t, \xi\right),\ \xi \sim \mathcal{N}(0,\Delta t)\ \text{(or other noise)}$$



Which probability densities for $p(X(t+\Delta t|X(t))$ arise?
Can we use them for training? Yes!

See our paper, also for Langevins + SPDE.

D., Makeev, Kevrekidis, Evangelou, Bertalan, Reich, Kevrekidis, Chaos, 2022; pre-print: arXiv:2106.09004

Our approach to estimating SDE

Results (kMC lattice model)

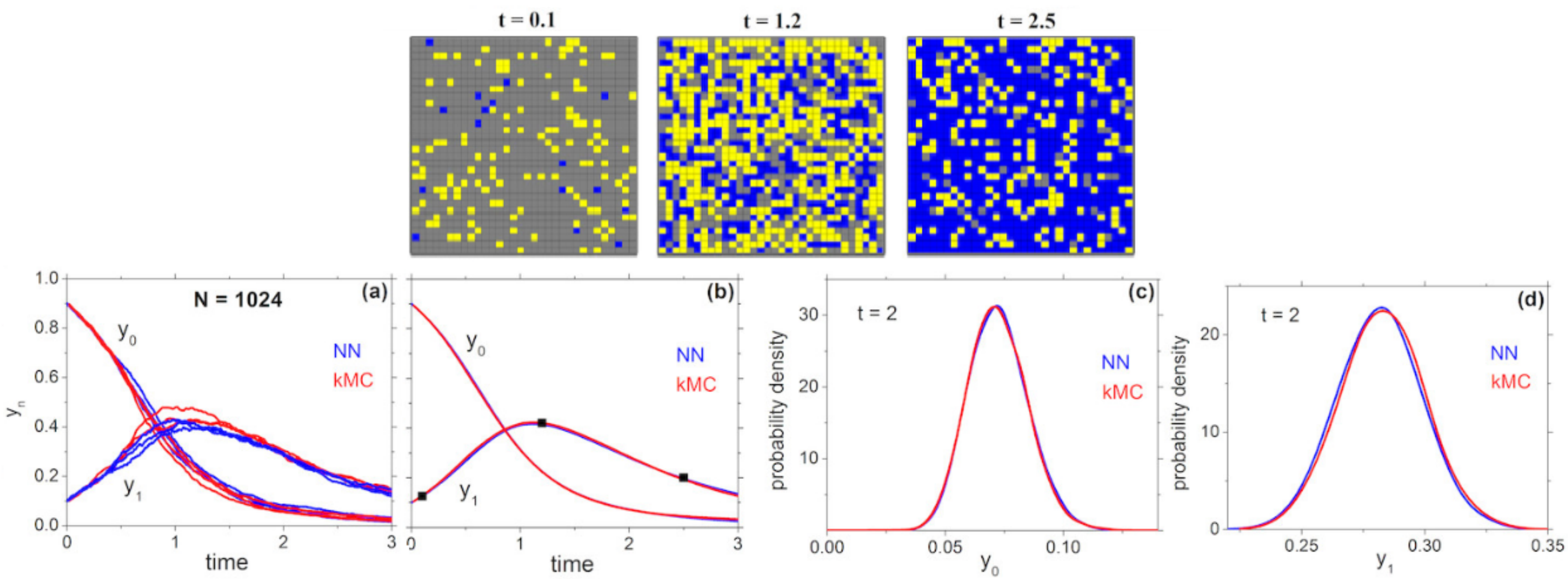
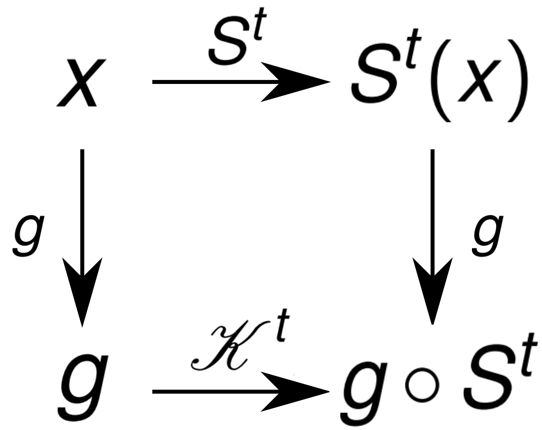


Figure: Ilustration of the kMC lattice (top row), physically relevant values measured over time resp. simulated with the identified SDE from the network (a), averaged paths over 200 simulations (b), and propagated densities from the initial condition until $t = 2$ (c,d). Lattice snapshots show S, I and R-type species as grey, yellow and blue squares, respectively.

D., Makeev, Kevrekidis, Evangelou, Bertalan, Reich, Kevrekidis, Chaos, 2022; pre-print: arXiv:2106.09004

Koopman operator framework



Koopman operator of dynamical systems

Main idea: “Dynamics of observables” instead of “dynamics of states”

S^t is the flow of a dynamical system, $S^t(x(0)) = x(t)$. The observable $g \in \mathscr{F}$ is transformed by $\mathscr{K}^t : \mathscr{F} \to \mathscr{F}$, which yields $(g \circ S^t) \in \mathscr{F}$.

$$[\mathscr{K}^t g](x) := g(S^t(x))$$

$$\begin{array}{ccc} x & \xrightarrow{S^t} & S^t(x) \\ \downarrow g & & \downarrow g \\ g & \xrightarrow{\mathscr{K}^t} & g \circ S^t \end{array}$$

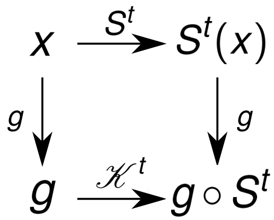
Li, D., Bollt, Kevrekidis, 2017
Bollt, Li, D., Kevrekidis, 2018
D., Thiem, Kevrekidis, 2020
Lehmberg, D., Köster, 2021
Gallos, Lehmberg, D., Siettos, 2024

Koopman operator of dynamical systems

Main idea: “Dynamics of observables” instead of “dynamics of states”

S^t is the flow of a dynamical system, $S^t(x(0)) = x(t)$. The observable $g \in \mathscr{F}$ is transformed by $\mathscr{K}^t : \mathscr{F} \to \mathscr{F}$, which yields $(g \circ S^t) \in \mathscr{F}$.

$$[\mathscr{K}^t g](x) := g(S^t(x))$$



Benefits

1. Each flow S^t has an associated operator (semi-group).
2. $\mathscr{K}^t$ captures a lot of information about S^t (also: spatial isomorphism $\Longrightarrow$ spectral isomorphism).
3. “Global linearization” of the dynamics.
4. Prediction is trivial in the eigenfunction space.
5. **Estimation from scattered data**, with approximation error estimates even for finite data.
6. **Data-driven design of linear controllers for non-linear systems**.

Li, D., Bollt, Kevrekidis, 2017
Bollt, Li, D., Kevrekidis, 2018
D., Thiem, Kevrekidis, 2020
Lehmberg, D., Köster, 2021
Gallos, Lehmberg, D., Siettos, 2024

Koopman operator approximation

Extended Dynamic Mode Decomposition: Algorithm

Input: N_X data pairs (X_n, X_{n+1}), $X_n, X_{n+1} \in M$, s.t. $X_{n+1} = S^t(X_n)$.

1. Define a dictionary with N_D observables, **e.g. with a neural network from M to $\mathbb{R}^{N_D}$,**
$$D = \{d_k : M \to \mathbb{R} | k = 1, \dots, N_D\} \subset \mathscr{F}.$$
2. Construct the matrix G,
$$G = D(X_n) = \left[d_1(X_n), d_2(X_n), \dots, d_{N_D}(X_n)\right] \in \mathbb{R}^{N_X \times N_D}.$$
3. Construct the matrix A,
$$A = [\mathscr{K}^t D](X_n) = \left[d_1(X_{n+1}), d_2(X_{n+1}), \dots, d_{N_D}(X_{n+1})\right] \in \mathbb{R}^{N_X \times N_D}.$$
4. Approximate the operator $\mathscr{K}^t$ through a matrix K, s.t. "$KG = A$", (e.g. using least-squares minimization):
$$\min_K \|KG^T G - A^T G\|^2,\ K \in \mathbb{R}^{N_D \times N_D}.$$

Williams, Kevrekidis, Rowley: "A Data-Driven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition", Journal of Nonlinear Science, 2015.
Li, D., Bollt, Kevrekidis: "Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the Koopman operator." Chaos, 2017. doi:10.1063/1.4993854

Koopman operator of dynamical systems

Data-driven design of optimal controllers

Given: data $(x(t), u(t))$ of a controlled dynamical system $\frac{d}{dt}x = f(x) + Gu$.
Goal: find optimal control inputs $u_{opt}(t)$ to minimize cost $c(x(t), x_{ref}(t))$.

Optimal control with the Koopman operator

1. The Koopman operator frameworks transforms the system to $\frac{d}{dt}z = Az + Bu$, $x = Cz$.
2. The control problem turns into minimization of $u^T Hu + Qu$.

Koopman operator of dynamical systems

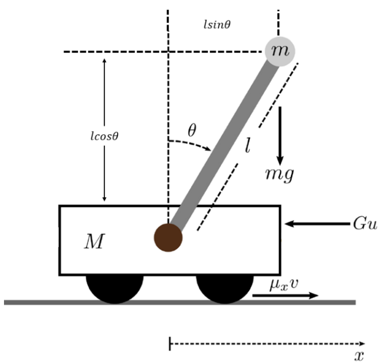
Data-driven design of optimal controllers

Given: data $(x(t), u(t))$ of a controlled dynamical system $\frac{d}{dt}x = f(x) + Gu$.
Goal: find optimal control inputs $u_{opt}(t)$ to minimize cost $c(x(t), x_{ref}(t))$.

Optimal control with the Koopman operator

1. The Koopman operator frameworks transforms the system to $\frac{d}{dt}z = Az + Bu$, $x = Cz$.
2. The control problem turns into minimization of $u^T Hu + Qu$.

Examples

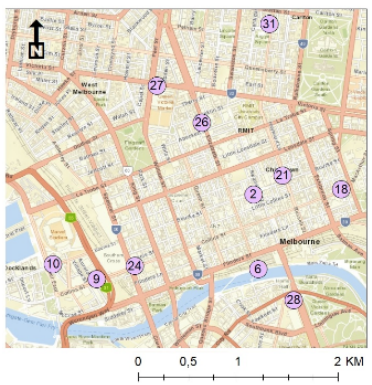


Sievers, Hartmann, D., 2025, to appear

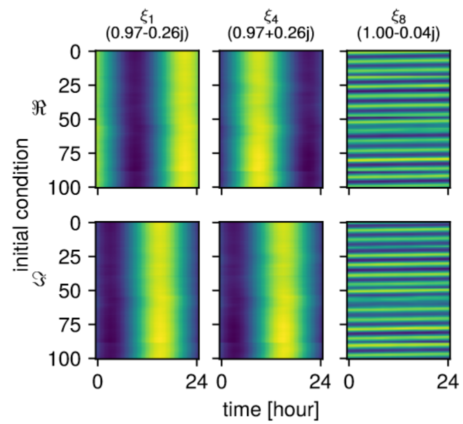
Lovelett, D., Lee, Kevrekidis, 2019

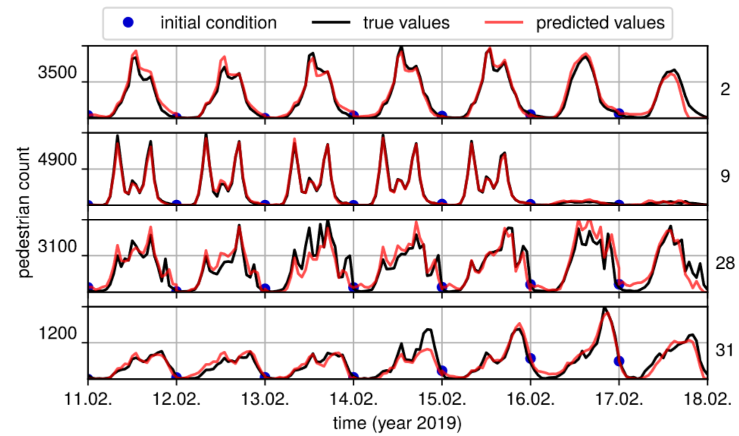
Koopman Operator case study (1/2)

Pedestrian counting data from Melbourne



©Open Street Map contributors, and the GIS User Community



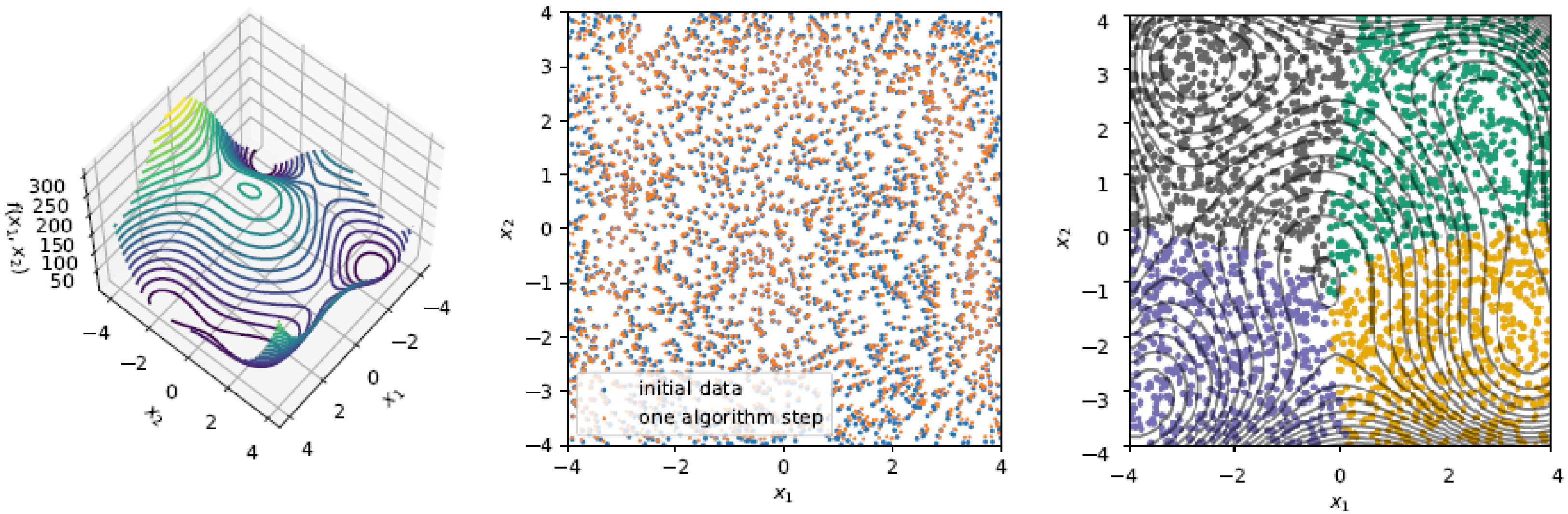


Left: Sensor positions in the city of Melbourne.
Center: Koopman operator eigenfunctions evaluated over the data.
Right: Surrogate model accuracy.

Lehmberg, D., Köster (2021)

Koopman Operator case study (2/2)

Studying algorithms as dynamical systems



Left: Himmelblau's function (optimization problem).
Center: Data (X_n, X_{n+1}) obtained from gradient descent: $X_{n+1} = X_n - \nabla f(X_n)$.
Right: Decomposition into basins of attraction, using Koopman operator eigenfunctions.

Paper: `https://doi.org/10.1137/19M1277059`; `https://arxiv.org/abs/1907.10807`

Open questions, outlook, conclusions

Open questions and future work

Learning invariances and uncertainty

- Symplectic forms, geometric symmetries, learning (Lie) group equivariances
- Uncertainty quantification for predictions and inverse problems
- Dynamics of higher-order, topological obstructions

“Learning dynamical systems” vs. ”dynamical systems for learning”

- Use iterative algorithms (SGD...) to identify systems
- Identify systems underlying the iterative systems (“Learning to optimize”)

... and much more!

- Numerical analysis, scientific machine learning, good software[1]
- Turbulence and chaos, system with continuous spectrum
- Causal relationships
- Discrete systems, non-smooth vector fields
- Learning on graphs
- ...

[1] datafold: https://datafold-dev.gitlab.io/datafold/

Outlook: Connection to next talk

Linear operator decomposition

Given $\mathscr{L} : A \to B$, assume we can perform a decomposition

$$\mathscr{L} = \sum_{k=1}^{\infty} \psi_k \lambda_k \phi_k =: \Psi \Lambda \Phi,$$

where $\psi_k \in B$, $\lambda_k \in \mathbb{C}$, $\phi_k \in A^*$.

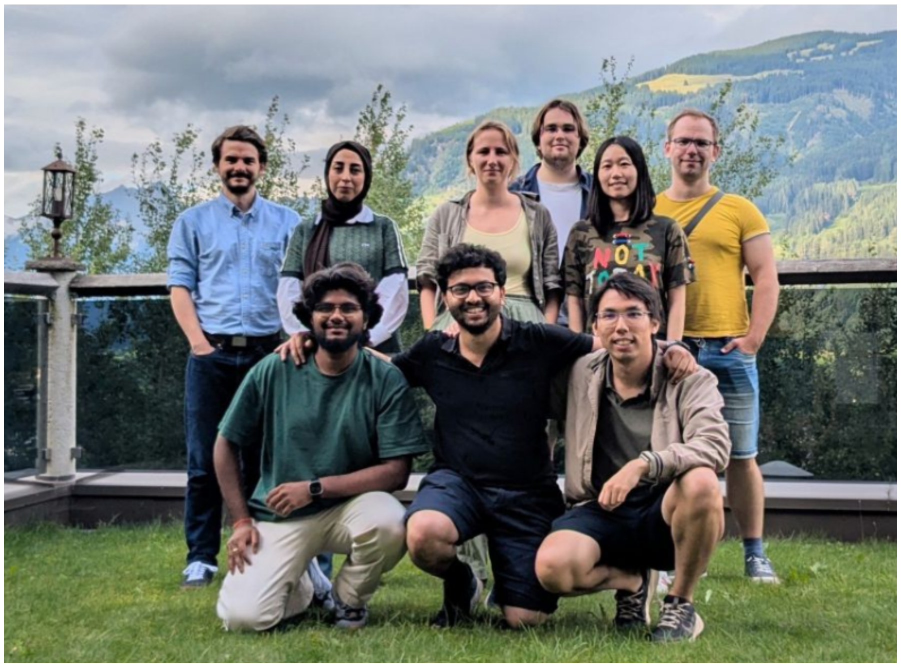
Connection to next talk (Friday, 9:00)

Given a sequence of neurons $\{g_k = \sigma(w_k \cdot + b_k)\}_k$, can we find a unitary transformation Q with

$$\mathscr{L} = \sum_{k=1}^{\infty} \psi_k \lambda_k \phi_k =: \Psi Q \Lambda^{1/2} \Lambda^{1/2} Q^* \Phi,$$

where $[\Psi Q \Lambda^{1/2}]_k = g_k$? Maybe we can even define Q as a random basis?

Discussion



Felix, Zahra, Iryna, Erik, Qing, Vladyslav, Shyam, Chinmay, Hessel. Not in the picture: Ana, Atamert, Berkay, Felix S., Maximilian, Nadiia, Rahul.

Contact

`felix.dietrich@tum.de`

`www.cs.cit.tum.de/en/scml`

Funding

DFG

SPONSORED BY THE Federal Ministry of Education and Research

MDSI

TUM Georg Nemetschek Institute

Literature I

Coifman, R. R. and Lafon, S. (2006).
Diffusion maps.
Applied and Computational Harmonic Analysis, 21(1):5–30.

D., Kooshkbaghi, M., Bollt, E. M., and Kevrekidis, I. G. (2020).
Manifold learning for organizing unstructured sets of process observations.
Chaos: An Interdisciplinary Journal of Nonlinear Science, 30(4):043108.

Kemeth, F. P., Haugland, S. W., D., Bertalan, T., Hohlein, K., Li, Q., Bollt, E. M., Talmon, R., Krischer, K., and Kevrekidis, I. G. (2018).
An emergent space for distributed data with hidden internal order through manifold learning.
IEEE Access.

Koopman, B. O. (1931).
Hamiltonian systems and transformation in Hilbert space.
Proceedings of the National Academy of Sciences of the United States of America, 17(5):315–318.

Lehmberg, D., D., Köster, G., and Bungartz, H.-J. (2020).
datafold: data-driven models for point clouds and time series on manifolds.
Journal of Open Source Software, 5(51):2283.

Li, Q., D., Bollt, E. M., and Kevrekidis, I. G. (2017).
Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the Koopman operator.
Chaos: An Interdisciplinary Journal of Nonlinear Science, 27(10):103111.

Mezić, I. (2005).
Spectral properties of dynamical systems, model reduction and decompositions.
Nonlinear Dynamics, 41(1):309–325.

Nüske, F., Peitz, S., Philipp, F., Schaller, M., and Worthmann, K. (2023).
Finite-Data Error Bounds for Koopman-Based Prediction and Control.
Journal of Nonlinear Science, 33(1):14.

Peterfreund, E., Lindenbaum, O., D., Bertalan, T., Gavish, M., Kevrekidis, I. G., and Coifman, R. R.
Loca: Local conformal autoencoder for standardized data coordinates.