Towards a Data-Driven, Model-Free Nonlinear Process Control Theory

PSE Seminar @ Purdue University
March 1, 2024

Wentao Tang

Assistant Professor, Chemical & Biomolecular Engineering

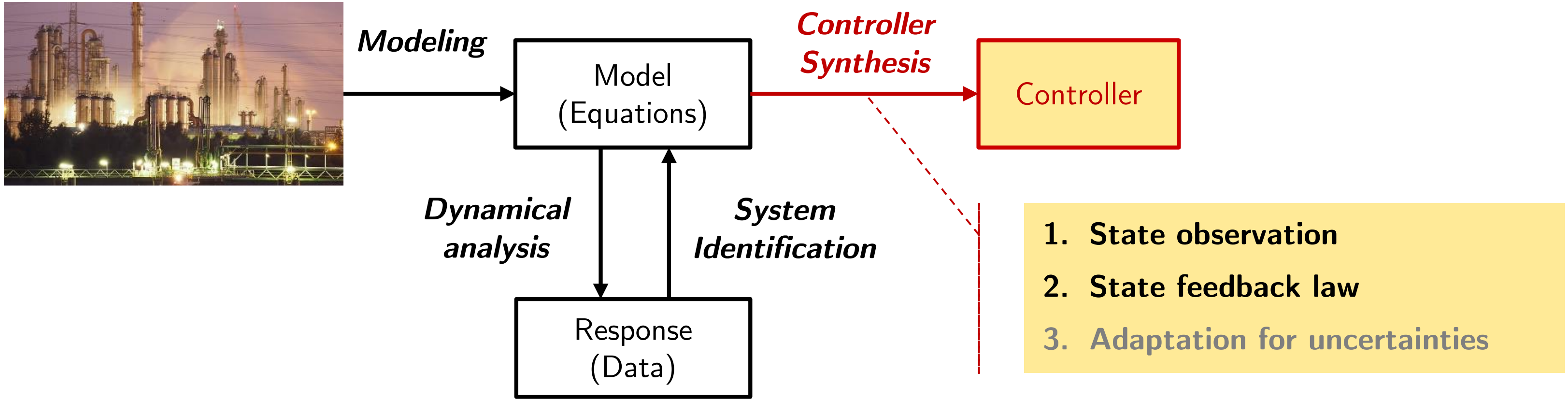
NC STATE UNIVERSITY

A Primer on Nonlinear Process Control

- Standard language: State-space form

$$\dot{x} = f(x, u) \qquad x(t) \in \mathbb{R}^n, \quad u(t) \in \mathbb{R}^m, \quad y(t) \in \mathbb{R}^p$$
$$y = h(x, u)$$

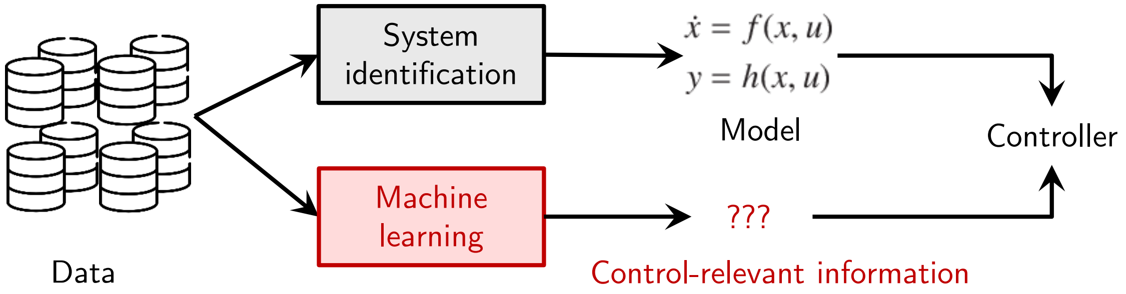
- Problems in a workflow



Isidori, A. (1985). *Nonlinear control systems*. Springer.
Sontag, E. D. (2013). *Mathematical control theory: deterministic finite dimensional systems*. Springer.

Data-Driven Control: Use of Machine Learning

- Different ideas of using ML in control
  - Modeling – sparse, kernel, neural methods
  - Monitoring – fault detection and performance maintenance
  - Model-free control

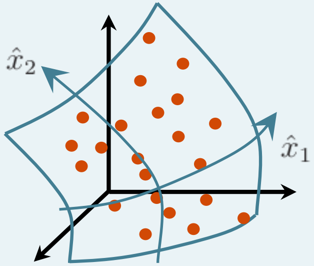


- Why model-free control?
  1. Technical factors – faster workflow, utilization of simulated/operational data
  2. Human factors – loss of workforce, need for time flexibility, accessibility to advanced control system
  3. Personal perception – model-based control is error-prone (not “fool-proof”)

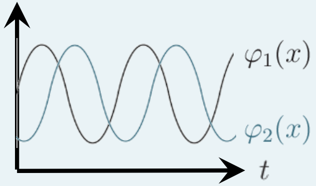
Tang, W., & Daoutidis, P. (2022). Data-driven control: Overview and perspectives. In *2022 American Control Conference (ACC)* (pp. 1048-1064).
Soudbakhsh, D., et al. (2023). Data-driven control: Theory and applications. In *2023 American Control Conference* (ACC) (pp. 1922-1939).

Towards Data-Driven Nonlinear Control

Observation / Estimation

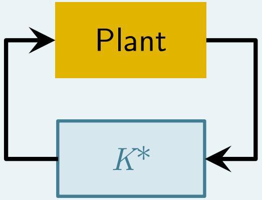


Latent states

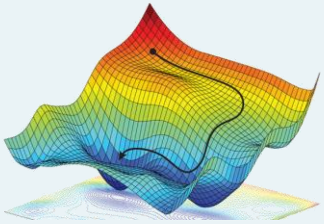


Dynamic modes

Decision Making



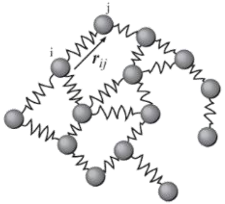
Controller synthesis



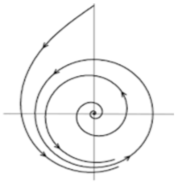
Optimization

Nonlinear Behaviors

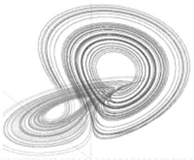
Conservation



Bifurcation



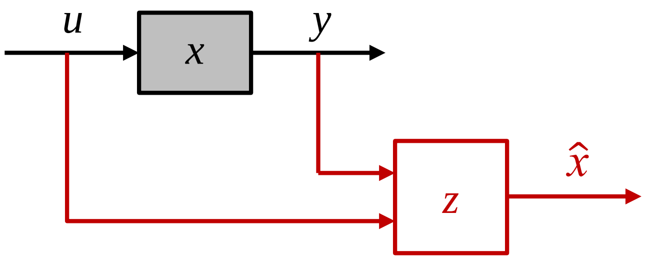
Chaos



Ramifications

- Molecular dynamics
- Multiphase flow
- Dynamical analysis of optimization algorithms

I. Data-Driven Nonlinear State Observation



$$\dot{z} = \varphi(z, y, u)$$
$$\hat{x} = \psi(z, y, u)$$

Papers:

Tang, W. (2023). Data-driven state observation for nonlinear systems based on online learning. *AIChE Journal*, e18224.
Tang, W. (2024). Synthesis of data-driven nonlinear state observers using Lipschitz-bounded neural networks. *To appear on ACC*. arXiv:2310.03187.
Weeks, C., & Tang, W. (2024). Data-driven nonlinear state observation using video measurements. *To appear on 12th ADCHEM*. arXiv:2311.14895.
Woelk, M., & Tang, W. (2024). *Manuscript in preparation.*

(Model-Based) State Observation: Classical Results

- Linear systems

$$\dot{x}(t) = Fx(t),\ y(t) = Hx(t)$$

  - Luenberger observer: LTI dynamics + linear output map

$$\boxed{\begin{aligned} \dot{z}(t) &= Az(t) + By(t), \\ \hat{x}(t) &= T^{\dagger} z(t). \end{aligned}}$$

where $T^{\dagger}$ is the left-pseudoinverse of T, determined by a Sylvester equation

$$TF - AT = BH$$

$\supseteq$

- Special case: "Kalman filter"
  - Let $A = F - BH$. Then $T = I$, and

$$\dot{\hat{x}}(t) = F\hat{x}(t) + B\left(y(t) - H\hat{x}(t)\right)$$

- Nonlinear systems

$$\dot{x}(t) = F(x(t)),\ y(t) = H(x(t))$$

  - **Kazantzis-Kravaris-Luenberger** observer: LTI dynamics + nonlinear output map

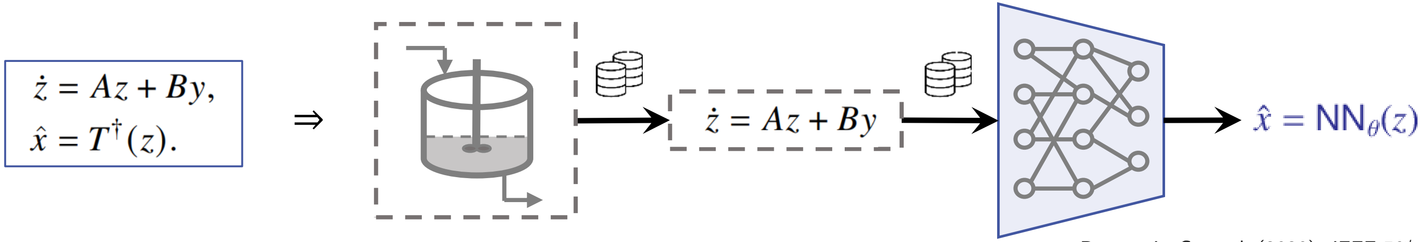
$$\boxed{\begin{aligned} \dot{z} &= Az + By, \\ \hat{x} &= T^{\dagger}(z). \end{aligned}} \qquad \frac{\partial T}{\partial x}(x)F(x) = AT(x) + BH(x), \quad \forall x \in \mathcal{X}.$$

where $T^{\dagger}$ is the left-pseudoinverse of a nonlinear transform T, determined by the PDE system [which can be solved (with some difficulties) if the model (F, H) is known.]

Luenberger, D. G. (1964). *IEEE Trans. Mil. Electr.*, 8(2), 74-80.
Kazantzis, N., & Kravaris, C. (1998). *Syst. Control Lett.*, 34, 241-247.

1. Lipschitz-Bounded Neural Observer

- Neural KKL observer: Assign the linear observer dynamics and train the static mapping



Ramos, L. C. et al. (2020). *IEEE 59th CDC*, 5435-5442.
Miao, K., & Gatsis, K. (2023). *5th L4DC*, 208-219.
Niazi, M. U. B., et al. (2023, May). *ACC*, 3048-3055.

  - Limitation: Overfitted neural network → generalization loss
  - Solution: Constraining the Lipschitz constant $\mathrm{Lip}(\mathrm{NN}_\theta) \le L$

- **Why?**

Theorem. Probabilistic guarantee on the mean squared state observation error:

$$R(\theta) \le \hat{R}(\theta) + C(\delta, \epsilon, h_{A,B}, \sigma) \cdot (1 + \mathrm{Lip}(\mathrm{NN}_\theta)\mathrm{Lip}(T))^2$$

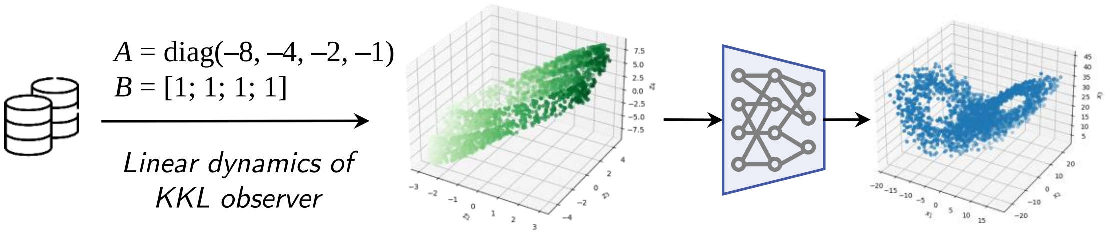
gen. loss — train. loss

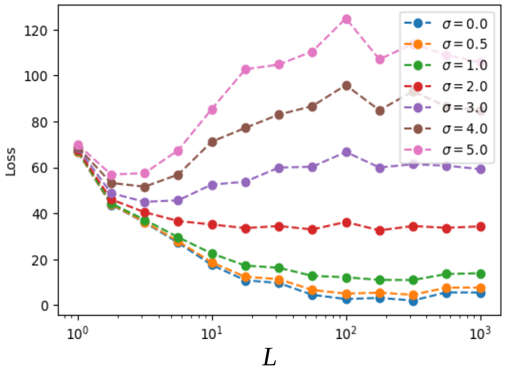
$1-\delta$: confidence
ε: initialization effect, practically 0
$h_{A,B}$: sensitivity to noise, σ: noise

- **How?** A special NN architecture, see **Wang, R., & Manchester, I.** (2023). ICML (pp. 36093-36110)
(Easy to implement with PyTorch.)

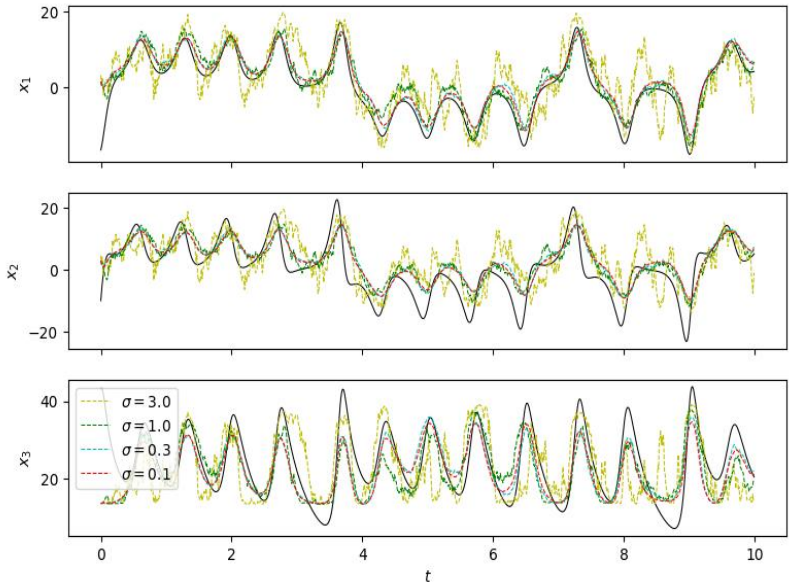
1. Lipschitz-Bounded Neural Observer

- Example: Lorenz system





L = 10



- When the environment is highly noisy, the Lipschitz bound has a severe effect on the generalization loss
- Increasing noise causes more noisy observations and sometimes incorrect directions of evolution

2. Online Least Squares for a Chen-Fliess Observer

- Neural networks – nonconvex and stochastic training, too many parameters
- **Query**: Linear parameterization of observer, amenable to convex optimization (least squares)
  - Much simpler, more efficient, and more reliable performance

- KKL observer as an input-affine system:

$$\dot{z} = g_0(z) + \sum_{i=1}^{m} g_i(z)y, \quad \hat{x} = h(z).$$

  - Lie derivatives

$$L_{g_{i_k}} \cdots L_{g_{i_2}} L_{g_{i_1}} h_j(z) = \frac{\partial}{\partial z}\left(\cdots \frac{\partial}{\partial z}\left(\frac{\partial h_j}{\partial z} g_{i_1} \right) g_{i_2} \cdots \right) g_{i_k}(z),$$

$$i_1, \ldots, i_k = 0, 1, \ldots, m, \quad j = 1, \ldots, n.$$

  - Recursive integrals

$$E_i(t_0, t_1) = \int_{t_0}^{t_1} y_i(\tau) d\tau, \quad i = 0, 1, \ldots, m, \quad t_0, t_1 \in \mathbb{R}, \quad t_0 \le t_1.$$

$$E_{i_1 i_2 \ldots i_k}(t_0, t_1) = \int_{t_0}^{t_1} E_{i_1 i_2 \ldots i_{k-1}}(t_0, \tau) y_{i_k}(\tau) d\tau, \quad k \ge 2.$$

- Chen-Fliess series: Within a time window $\Delta \in [0, \overline{\Delta}]$:

$\mu \in \mathbb{I}_m^k$: A multi-index of length k from $\{0, 1, 2, \ldots, m\}$

$$\hat{x}_j(t + \Delta) = \sum_{k=0}^{\infty} \sum_{\mu \in \mathbb{I}_m^k} L_\mu h_j(z(t)) E_\mu(t, t + \Delta).$$

Data labels for training — $\hat{x}_j(t + \Delta)$

Coefficients to be estimated — $L_\mu h_j(z(t))$

Input features of the data — $E_\mu(t, t + \Delta)$

Now amenable to linear regression!

2. Online Least Squares for a Chen-Fliess Observer

- **Truncation** to a finite order K of terms

$$\theta_j(t) = \left[L_\mu h_j(z(t))\right]_{\mu \in \mathbb{I}_m^{\leq K}}, \quad \phi(t,\delta) = \left[E_\mu(t, t+\delta)\right]_{\mu \in \mathbb{I}_m^{\leq K}}$$

Coefficients to be estimated *Input features*

- Update the solution in continuous time using **online gradient descent**

$$\dot{\theta}_j(t) = -\eta \nabla J(\theta_j(t), t)$$

- A least squares problem: moving horizon with fixed length

$$\min_{\theta_j} \; J(\theta_j, t) := \frac{1}{2} \int_0^{\Delta} \left(\theta_j^\top \phi(t,\delta) - x_j(t+\delta)\right)^2 d\delta.$$

Theorem. Bound on mean squared observation error

$$\frac{1}{t}\int_0^t \|\hat{x}(\tau) - x(\tau)\|^2 d\tau \leq \frac{C}{t}\int_0^t \|\dot{x}(\tau)\|^2 d\tau + C' + \frac{C''}{t}.$$

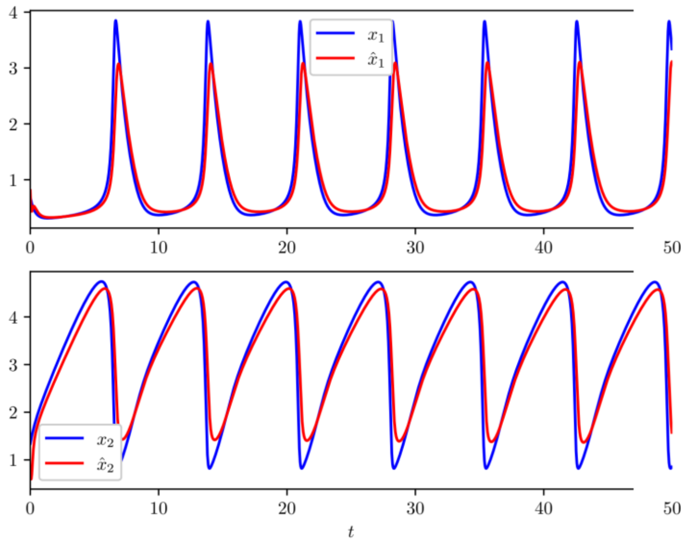
The bound depends on (i) truncation length, (ii) intensity of persistent excitation, and (iii) horizon length, in addition to (iv) variation rate of the true states.

2. Online Least Squares for a Chen-Fliess Observer

- Example 1: Brusselator

$$\dot{x}_1 = 1 + x_1^2 x_2 - 4x_1, \quad \dot{x}_2 = 3x_1 - x_1^2 x_2$$

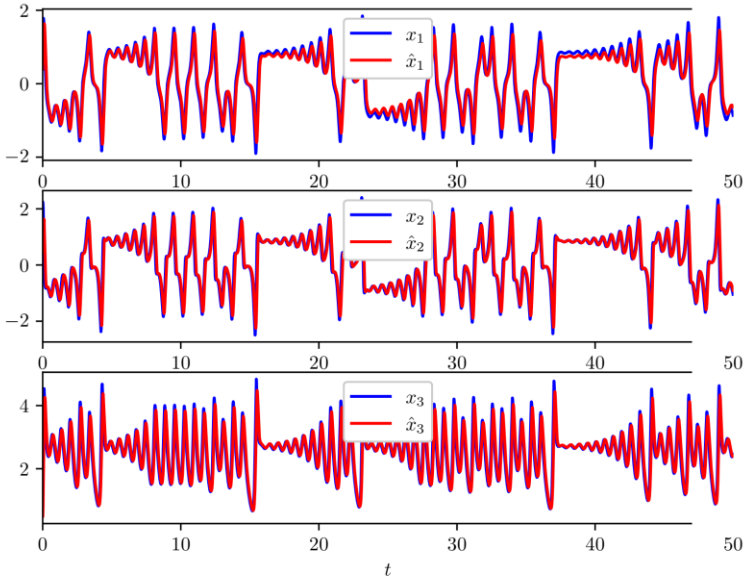
$$y = x_1 + x_2$$



- Example 2: Lorenz system

$$\dot{x}_1 = 10(x_2 - x_1), \quad \dot{x}_2 = x_1(28 - 10x_3) - x_2, \quad \dot{x}_3 = 10x_1x_2 - (8/3)x_3.$$

$$y = x_2$$

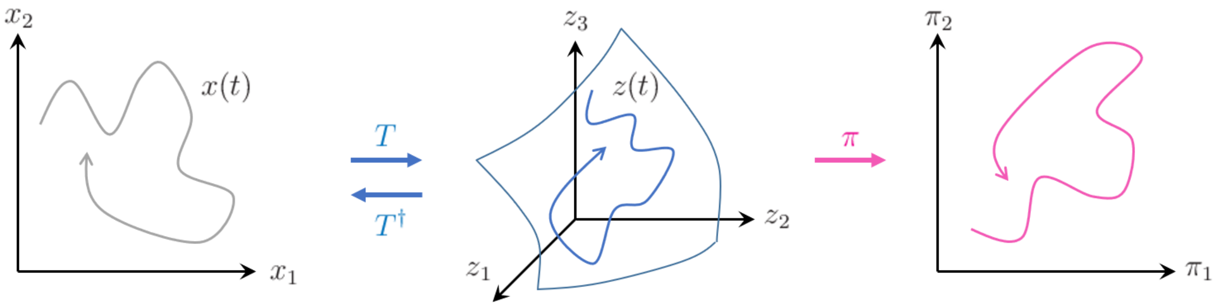


Online optimized Chen-Fliess series tracks the true states very well, especially when the states vary slowly.

3. Observer without State Information

- Previously: Supervised learning (regression) by empirical risk minimization – need to have labels
  - "*Somehow* the true states are available for training, although in operations they must be estimated."
  - A paradoxical setting – we must have a high-fidelity simulator – then why not model-based?

- Now: No labels, **unsupervised learning**
  - **Dimensionality reduction** problem: Find a mapping $z \mapsto \pi$, so that π and x are "equivalent"
    - Anyways, the concept of "states" is artificial and transformable by a diffeomorphism
    - Need π to be diffeomorphic to x: a very weak requirement that can be satisfied by PCA/kernel PCA

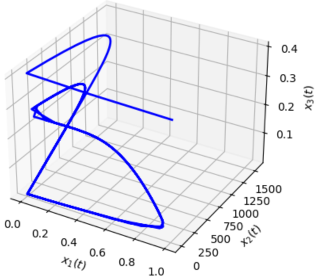


3. Observer without State Information

- Belousov-Zhabotinsky reactions (well-stirred)

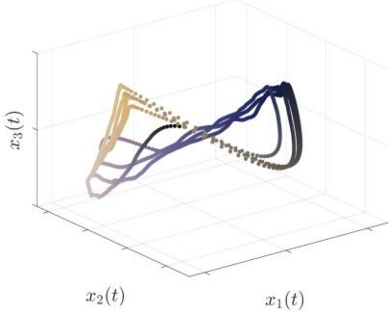
$$\epsilon\frac{dx_1}{dt} = qx_2 - x_1x_2 + x_1(1-x_1),$$
$$\delta\frac{dx_2}{dt} = -qx_2 - x_1x_2 + fx_3,$$
$$\frac{dx_3}{dt} = x_1 - x_3.$$

  - Measured output signal: Colors of 300 pixels in a **video**
    - https://www.youtube.com/watch?v=ieh9qIkkMJQ
  - KKL observer: A = 1200[th] order diagonal (placed pole to assign time constants), B = 1200-by-300, $T^\dagger$ by PCA



Simulated by model

- Observed state orbit exhibits a "bow-tie" shape, consistent with the true state orbits
- The cycles are slowly decaying – a physical reality honestly reflected by the data (but not captured by the model)



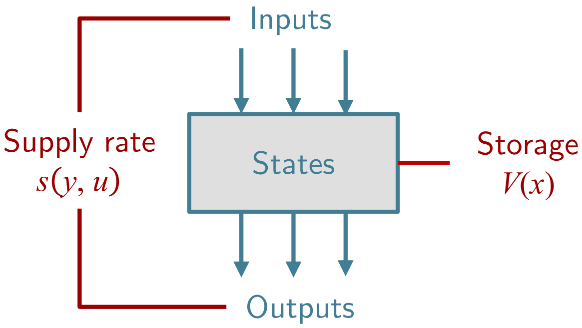
Estimated by observer

I. Data-Driven Nonlinear State Observation

Summary

- State observation is cast as a **machine learning** problem and becomes easier
  - Convex online optimization / nonconvex optimization done carefully
  - Satisfactory practical performance
- Potential applications to industrial systems with **massive real-time data** (esp. cameras)
  - Exploiting data to see "where the system is" → Monitoring and control
  - Combined with any **control strategy** that assumes state availability (e.g., RL/MPC)
- Ongoing directions
  - Observer for non-autonomous systems $dx/dt = f(x, u)$, $y = h(x, u)$

II. Dissipativity Learning Control [DLC]



$$\frac{dV(x(t))}{dt} \leq s(y(t), u(t))$$

Dissipativity

$$u = \kappa(y)$$

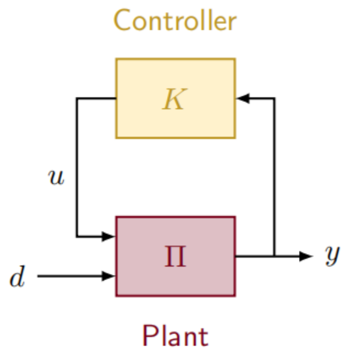
Dissipativity-based controller

Papers: Tang, W., & Daoutidis, P.
(2019). Input-output data-driven control through dissipativity learning. *American Control Conference* (pp. 4217-4222).
(2019). Dissipativity learning control (DLC): A framework of input–output data-driven control. *Comput. Chem. Eng.*, 130, 106576.
(2021). Dissipativity learning control (DLC): Theoretical foundations of input–output data-driven model-free control. *Syst. Control Lett.*, 147, 104831.
Tang, W., & Woelk, M. (2023). Dissipativity learning control through estimation from online trajectories. *American Control Conference* (pp. 3036-3041).

Dissipativity: Control-Relevant Information

- Relation to stability and performance
  - Stabilizing control: find $u = \kappa(y)$ such that $s(y, \kappa(y)) \leq 0$.
    - $\dot{V} \leq 0$ → closed-loop Lyapunov stability
  - L_2-gain: $u \to y$ has a finite L_2-gain bounded by $\beta^{1/2}$, if

$$s(y, u) \leq \beta \|u\|^2 - \|y\|^2$$



- Example: L_2-optimal control for disturbance rejection
  - Variable: Controller gain K
  - Objective: L_2-gain of $d \to (y, u)$
  - A multi-convex semidefinite programming problem

$$\min_{K \in \mathcal{K}} \beta$$
$$\text{s.t.} \begin{bmatrix} I & K^\top & 0 \\ 0 & 0 & I \end{bmatrix} \left(\Pi + \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & -\beta I \end{bmatrix} \right) \begin{bmatrix} I & 0 \\ K & 0 \\ 0 & I \end{bmatrix} \preceq 0$$

Rojas, O. J., Bao, J., & Lee, P. L. (2008). *J. Process Control*, 18, 515-526.
Brogliato, B. et al. (2020). *Dissipative systems analysis and control: Theory and applications* (3rd ed.). Springer.

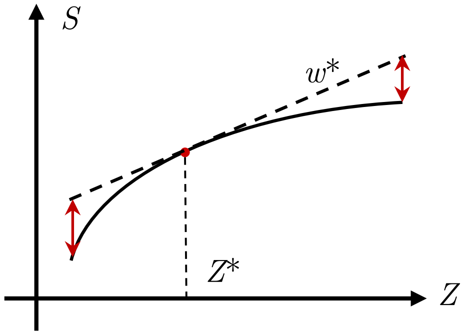
(Model-Based) Dissipativity Analysis

- Question: How do we know the dissipativity of a system?
  - Kalman-Yakubovich-Popov (KYP) lemma
    - Linear matrix inequality (LMI) / functional inequalities
  - **Thermodynamic analysis**
    - Difficult to find accurate thermodynamic relations
    - Conservative, suboptimal (e.g., fluid flow is not modeled)

B. E. Ydstie

K. M. Hangos

Alonso, A. A., & Ydstie, B. E. (2001). *Automatica*, 37, 1739-1755.
Hangos, K. M., et al. (2001). *AIChE J.*, 47, 1819-1831.



$$\frac{d\mathcal{A}}{dt} \le \int_{\partial\Omega} y^\top u dS$$

Extensive properties $Z = (U, V, m_1, \ldots, m_n)$

Intensive properties $w = \frac{\partial S}{\partial Z} = \left(\frac{1}{T}, \frac{P}{T}, -\frac{\mu_1}{T}, \ldots, \frac{\mu_n}{T}\right)$

Legendre transform $A(Z, Z^*) = S(Z^*) + w^{*\top}(Z - Z^*) - S(Z)$

$$\frac{\partial A}{\partial t} = -\bar{w}^\top \frac{\partial \bar{Z}}{\partial t} \quad (\bar{w} := w - w^*, \bar{Z} := Z - Z^*)$$

$$\frac{d}{dt}\int_\Omega A dV = \int_{\partial\Omega} \bar{w}^\top (\bar{\mathbf{f}} \cdot \mathbf{n}) dS - \int_\Omega \bar{\mathbf{f}} : \nabla \bar{w} dV - \int_\Omega \bar{w}^\top \bar{\sigma} dV$$

Storage	Outputs: T, P, μ	Inputs: flows	≥ 0 (Onsager)	problematic term (assume small)

Data-Driven Dissipativity Learning: General Form

- Dissipative inequality in a **duality form**

$$V(x^+) - V(x) \leq s(u,y) \quad \Rightarrow \quad \langle g_{x,x^+,u,y}, m\rangle \geq 0$$

  - Dissipativity **function** $m = (V, s)$ (system property to be learned), defined on a **function space** $\mathscr{F}$
  - Evaluation **functional** $g_{x,x^+,u,y}$ (specified by data points), defined on its **dual space** $\mathscr{F}^*$

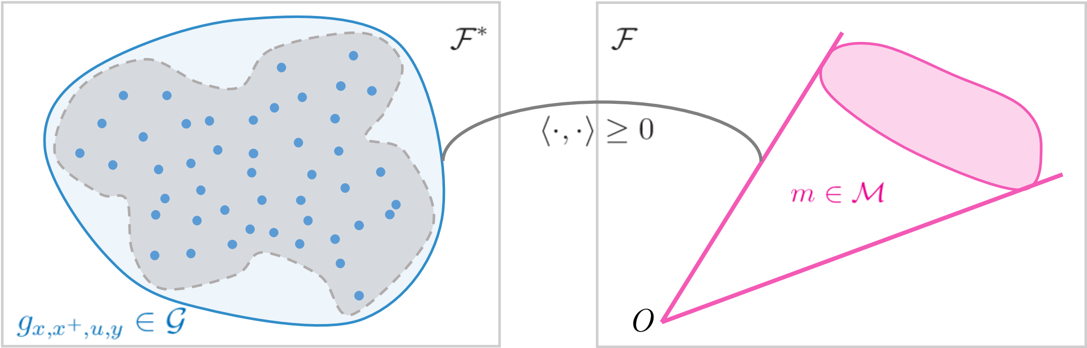
- Dual dissipativity set: All evaluation functionals from the "system population"

$$\mathscr{G} = \{g_{x,x^+,u,y} | (x, x^+, u, y) \in D\}$$

- Dissipativity set: All admissible dissipative properties

$$\mathscr{M} = \{m \in \mathscr{F} | \langle g, m\rangle \geq 0, \forall g \in \mathscr{G}\} = \mathscr{G}^*$$

Estimate the dual dissipativity set from data



Compute the dual cone as the dissipativity set

Data-Driven Dissipativity Learning: Quadratic Supply

- Linear parameterization

$$s(y,u) = \begin{bmatrix} y^\top & u^\top \end{bmatrix} \begin{bmatrix} \Pi_{yy} & \Pi_{yu} \\ \Pi_{yu}^\top & \Pi_{uu} \end{bmatrix} \begin{bmatrix} y \\ u \end{bmatrix} = \begin{bmatrix} y^\top & u^\top \end{bmatrix} \Pi \begin{bmatrix} y \\ u \end{bmatrix}$$

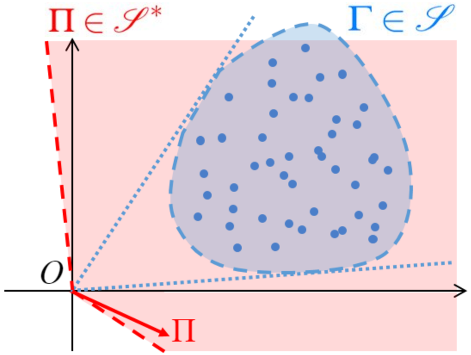
Quadratic form,
Parameters: Π or $\mathrm{vec}(\Pi)$

- Definitions
  - Dissipativity parameters $\Pi \in$ Dissipativity set
    - Property of the system to be learned
  - Dual dissipativity parameters $\Gamma \in$ Dual dissipativity set $\mathscr{S}$

$$\Gamma = \int_0^T \begin{bmatrix} y(t) \\ u(t) \end{bmatrix} \begin{bmatrix} y(t)^\top & u(t)^\top \end{bmatrix} \mathrm{d}t \succcurlyeq 0$$

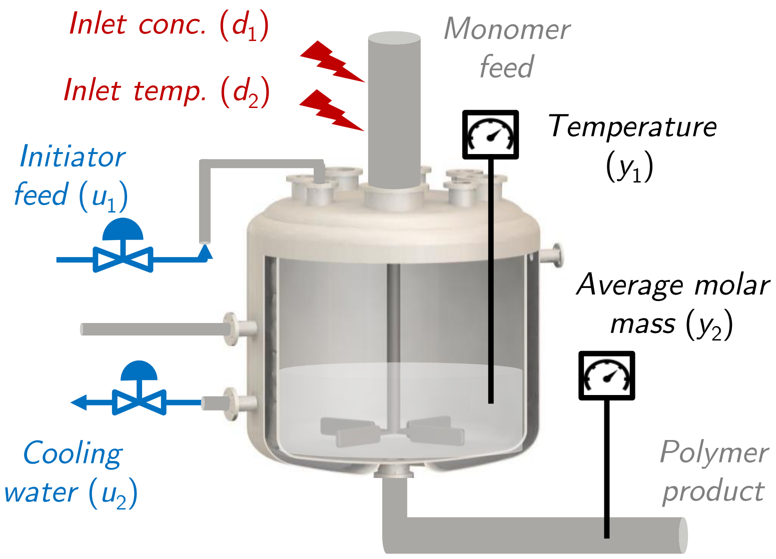
    - Property of data
  - **For any trajectory starting from 0,**

$$\mathrm{vec}(\Pi)^\top \mathrm{vec}(\Gamma) = \mathrm{trace}(\Pi^\top \Gamma) =: \langle \Pi, \Gamma \rangle \geq 0$$

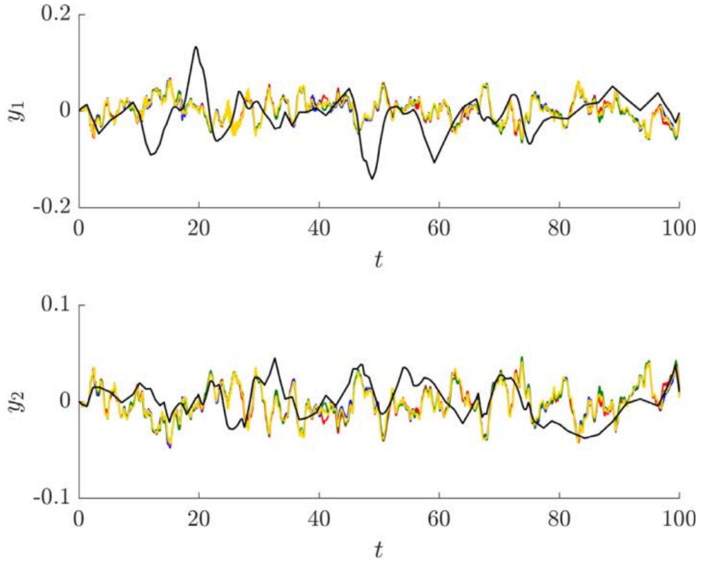


- Dissipativity learning procedure
  1. Collect Γ sample for trajectories **starting from 0**
  2. Estimate dual dissipativity set $\mathscr{S}$
  3. **Dual cone** of dual dissipativity set $\mathscr{S}^* =$ dissipativity set

Example 1: Polymerization Reactor

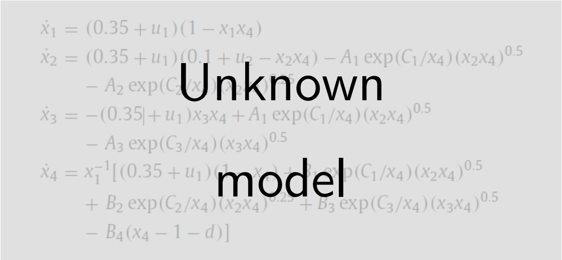


- Performance of DLC
  - Disturbances as Orstein-Uhlenbeck random processes in continuous time
  - $K = 0$ vs DLC-P controllers with 11 independent components and confidence levels 0.85, 0.90, 0.95, 0.99

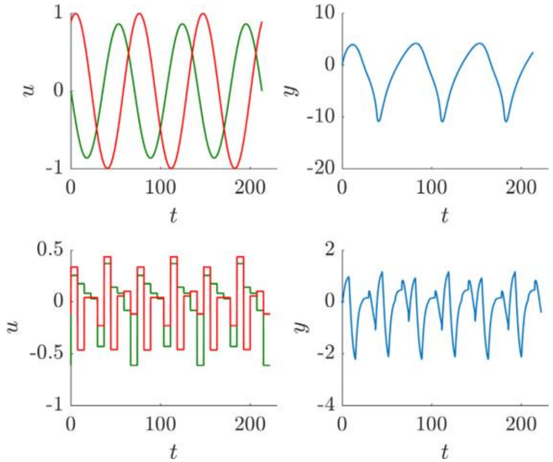


Daoutidis, P., Soroush, M., & Kravaris, C. (1990). *AIChE J.*, *36*(10), 1471-1484.

Example 2: Gas-Phase Reactor



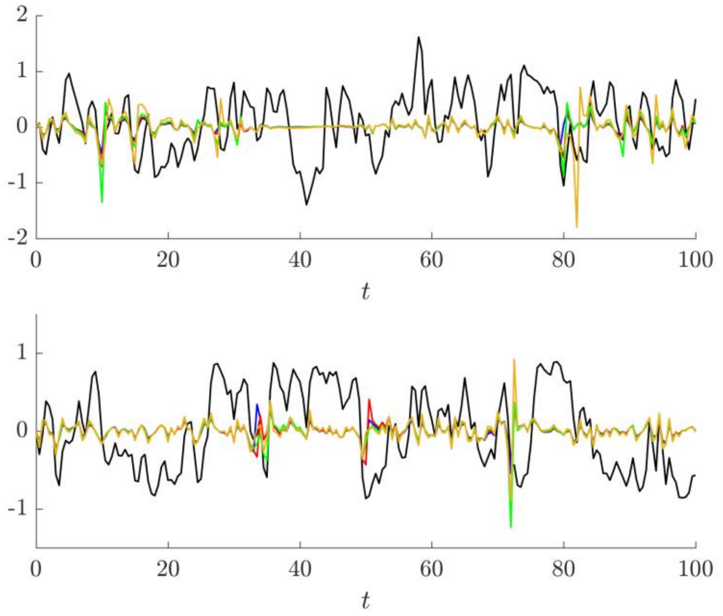
Reference trajectories for tracking control



Sinusoidal inputs

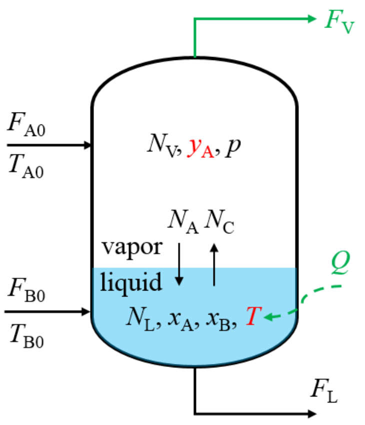
Periodic piecewise constant inputs

- Performance of DLC
  - $K = 0$ vs DLC-PID with 5 independent components and confidence levels 0.85, 0.90, 0.95, 0.99

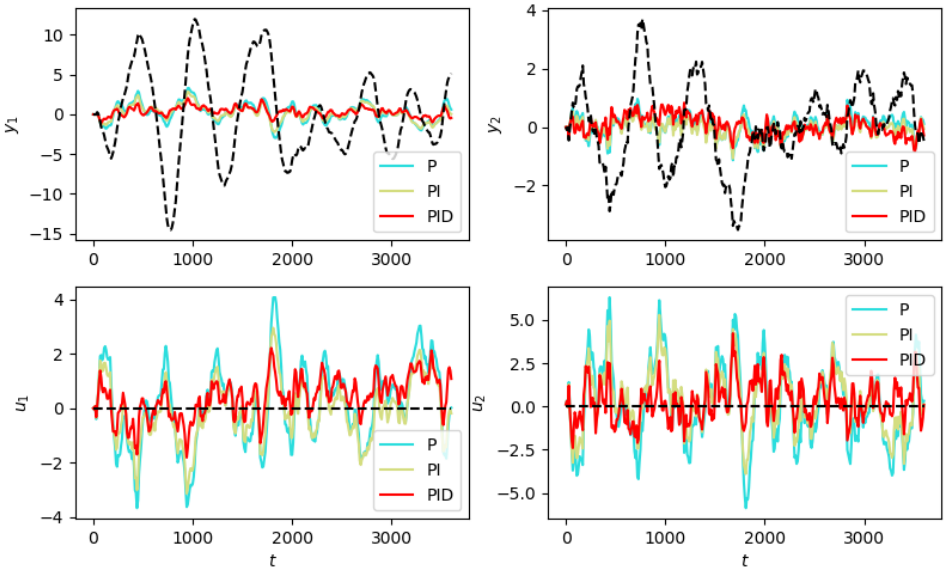


Özgülşen, F., *et al.* (1992). *Chem. Eng. Sci.*, *47*(3), 605–613.
Chen, C.-C., *et al.* (1994). *Can. J. Chem. Eng.*, *72*(4), 672–682.

Example 3: Two-Phase Reactor



Kumar, A., & Daoutidis, P. (1995). *AIChE J.*, *41*(3), 619-636.



Controller	Open-Loop	DLC-PID	DLC-PI	DLC-P	Linear SysID + LQG
ISE + ISC	35.0907	**2.5846**	**2.4316**	**2.5345**	**2.6766**

II. Dissipativity Learning Control [DLC]

Summary

- Dissipativity learning as a **machine learning** problem and becomes easier
  - Estimating a data distribution and finding its dual cone
  - Convex/multiconvex optimization for control performance
- Theoretical framework and preliminary works → Much more to be done to realize its potential
- Advantages of DLC as a technology [Ongoing research to establish them]
  - Inherently physics-informed, stability and performance-guaranteed
  - Structured and scalable to large systems
  - Flexible with big data (truly nonlinear) or small data (comparable with linear identification)

Optimization Algorithms as Dynamical Systems

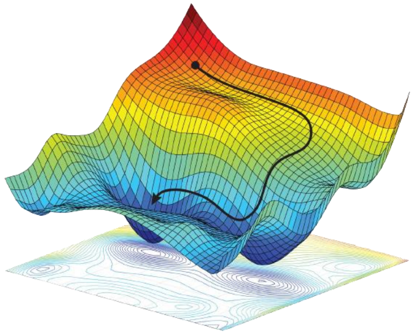
- Convex optimization $\min f(x)$
  - First-order dynamics (gradient flow) $\dot{x}(t) = -\nabla f(x(t))$
    - Forward difference → Gradient descent algorithm
    - Backward difference → Proximal algorithm [*non-smooth*]
  - Second-order dynamics
    - With vanishing damping → Nesterov's momentum

$$\ddot{x}(t) + \frac{\alpha}{t}\dot{x}(t) + \nabla f(x(t)) = 0$$

    - With Hessian damping → Attouch and Peypouquet

$$\ddot{x}(t) + \frac{\alpha}{t}\dot{x}(t) + \beta \nabla^2 f(x(t))\dot{x}(t) + \nabla f(x(t)) = 0$$

    - ...

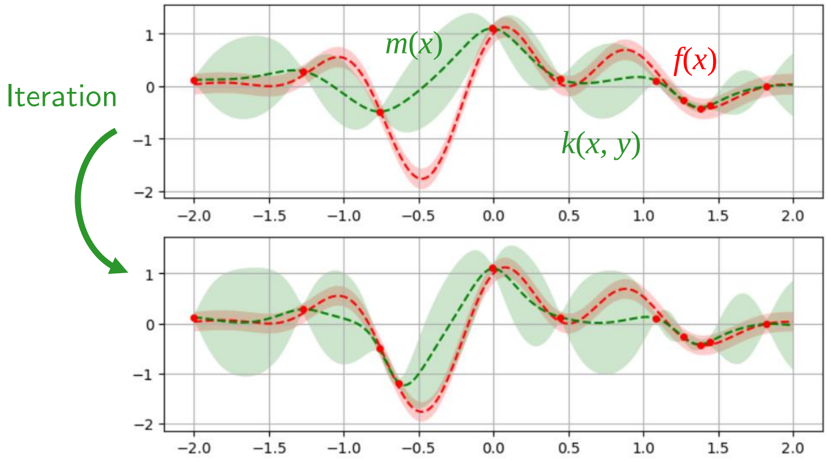


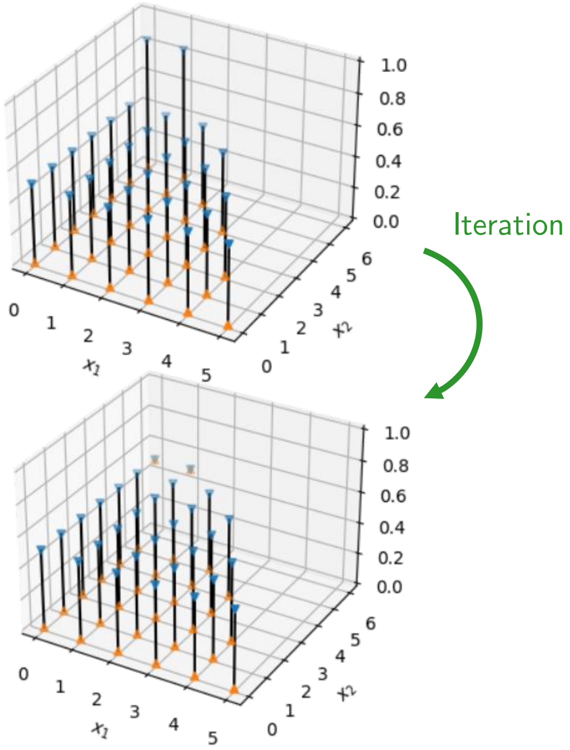
- Benefits of using dynamical analysis in optimization algorithms
  - Intuitive understanding of algorithm → Creation of new algorithms / combinations
  - Control-theoretic convergence proofs → Tuning of algorithm hyperparameters

Su, W., Boyd, S., & Candès, E. J. (2016). *J. Mach. Learn. Res.*, *17*(153), 1-43.
Attouch, H., & Peypouquet, J. (2019). *Math. Program.*, 174, 391-432.
Lessard, L., Recht, B., & Packard, A. (2016). *SIAM J. Optim.*, 26(1), 57-95.
Boț, R. I., & Nguyen, D. K. (2023). *SIAM J. Numer. Anal.*, 61(6), 2813-2843.

Global Optimization as Dynamical Systems ...

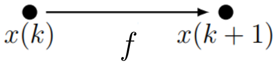
- Postulate – Dynamics on a function space?
  - Bayesian optimization: Dynamics of (m, k)
  - Branch-and-bound (and other): Dynamics of (UB, LB) on the feasible region Ω





Data-Driven Dynamical Analysis for Optimization

- Koopman approach
  - Nonlinear dynamics f on X (Euclidean or function spaces) … might be complicated
  - But consider the dynamics on its dual space X^*
  - For any **functional** $\varphi \in X^*$, $\varphi \mapsto \varphi \circ f$ specifies a linear operator – called Koopman operator



$$(\mathcal{K}\varphi)(x) = \varphi(f(x))$$

A nonlinear system is in fact a linear one in its (infinite-dimensional) dual space.

Data-driven approximation

Data: snapshots of the dynamics

- Dynamical mode analysis $\mathcal{K}\varphi = \lambda\varphi \quad \Rightarrow \quad \varphi(x(t)) \propto \lambda^t$
  - **Eigenfunctionals**: linearly evolving modes
    - Contractions, oscillations, conservations
  - Identifying dynamic modes from data → Info about algorithm behavior

Novel algorithms/proofs?
Auto-tuning/selection?
Interpretability?
…

Williams, M. O., Kevrekidis, I. G., & Rowley, C. W. (2015). *J. Nonlin. Sci.*, *25*, 1307-1346.
Mauroy, A., Susuki, Y., & Mezić, I. (2020). *Koopman operator in systems and control*. Springer.
Dietrich, F., Thiem, T. N., & Kevrekidis, I. G. (2020). *SIAM J. Appl. Dyn. Syst.*, 19, 860-885.
Redman, W. T., et al. (2022). *IEEE CDC* (pp. 6006-6011).

Acknowledgement

Graduate Students

Moritz Woelk

Cormak Weeks

Damilola Fasiku

Funding Sources

- UNC System – ROI
- ACS – PRF
- NCSU – FRPD Seed Grant
- NCSU – Startup Fund

THANK YOU!