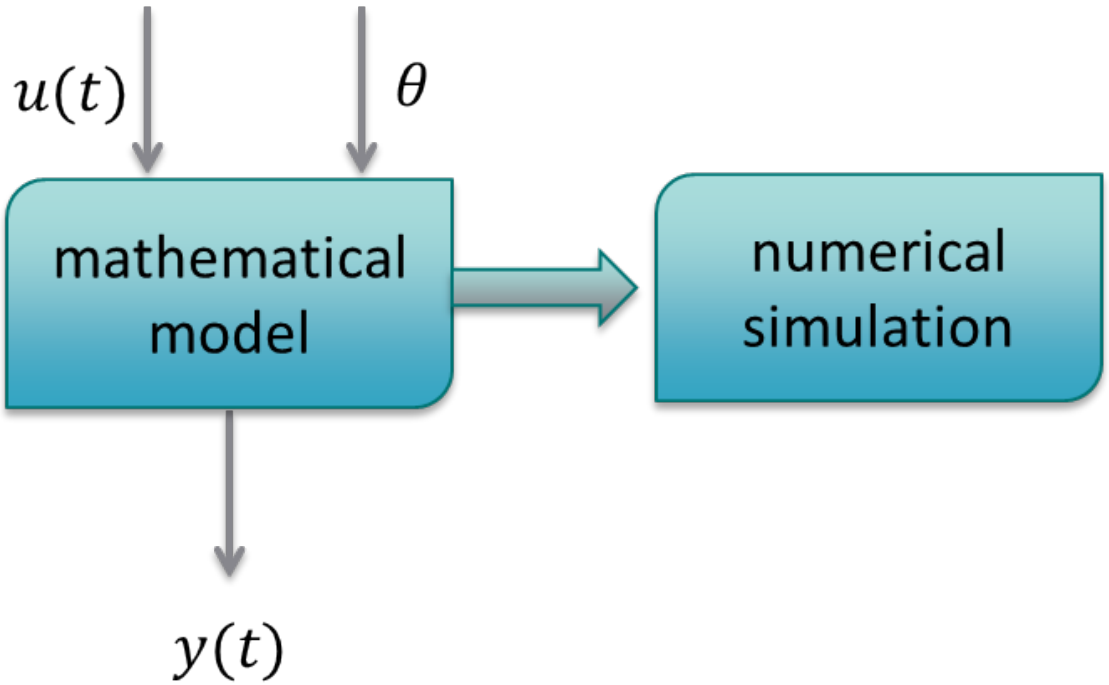
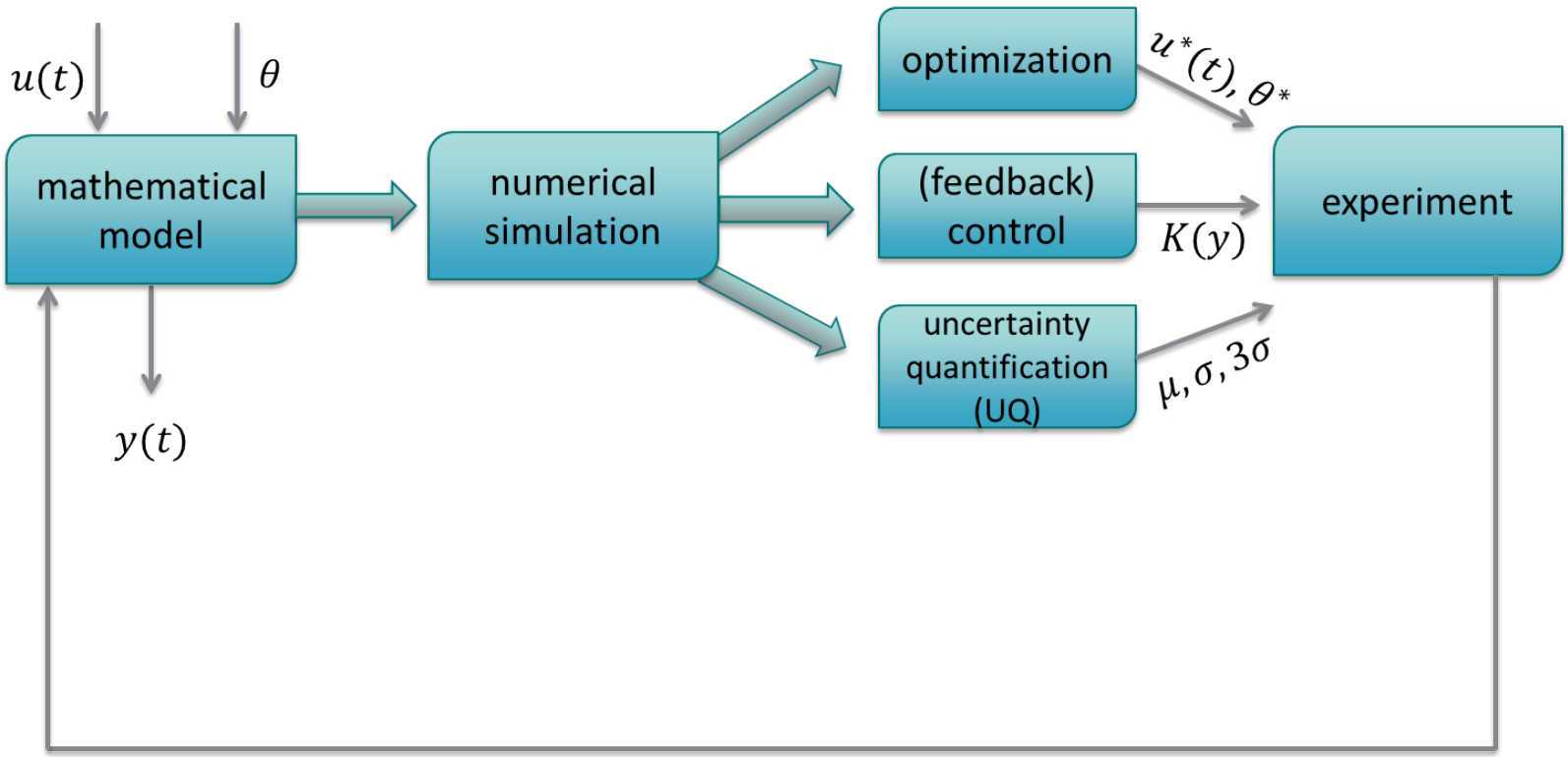

MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG
CSC
COMPUTATIONAL METHODS IN
SYSTEMS AND CONTROL THEORY
Learning State-Space Models of Dynamical Systems from Data
Peter Benner
Joint work with Pawan K. Goyal
Kolloquium der Arbeitsgruppe
Modellierung • Numerik • Differentialgleichungen
TU Berlin, October 25, 2022
Supported by:
DFG-Graduiertenkolleg
MATHEMATISCHE
KOMPLEXITÄTSREDUKTION
IMPRS
ProEng
Magdeburg
Partners:
OTTO VON GUERICKE
UNIVERSITÄT
MAGDEBURG


$u(t)$
$\theta$
mathematical model
numerical simulation
$y(t)$

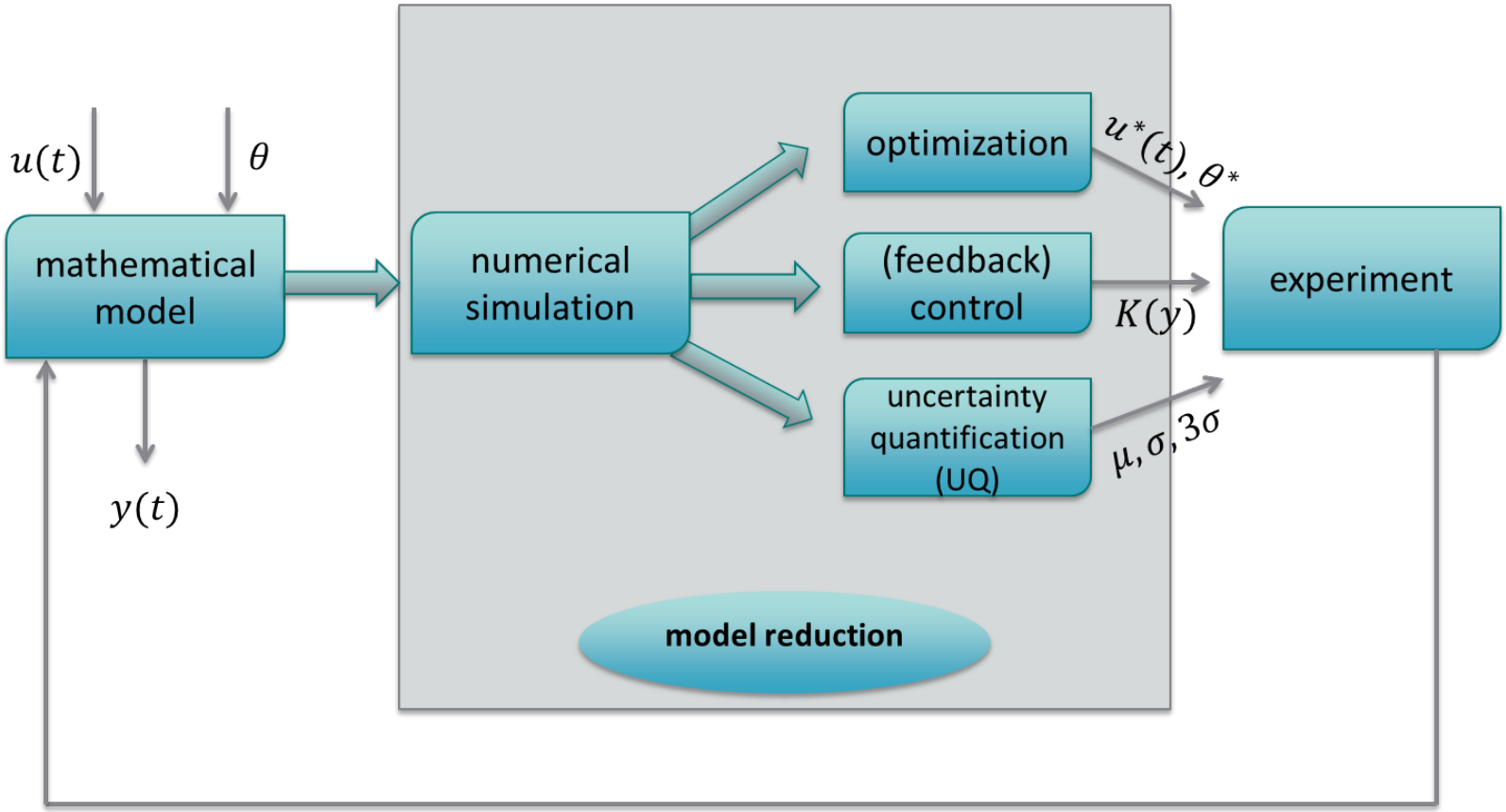
# A Process Chain in Computational Sciences and Engineering (CSE)

*Optimization, Control, and Uncertainty Quantification*



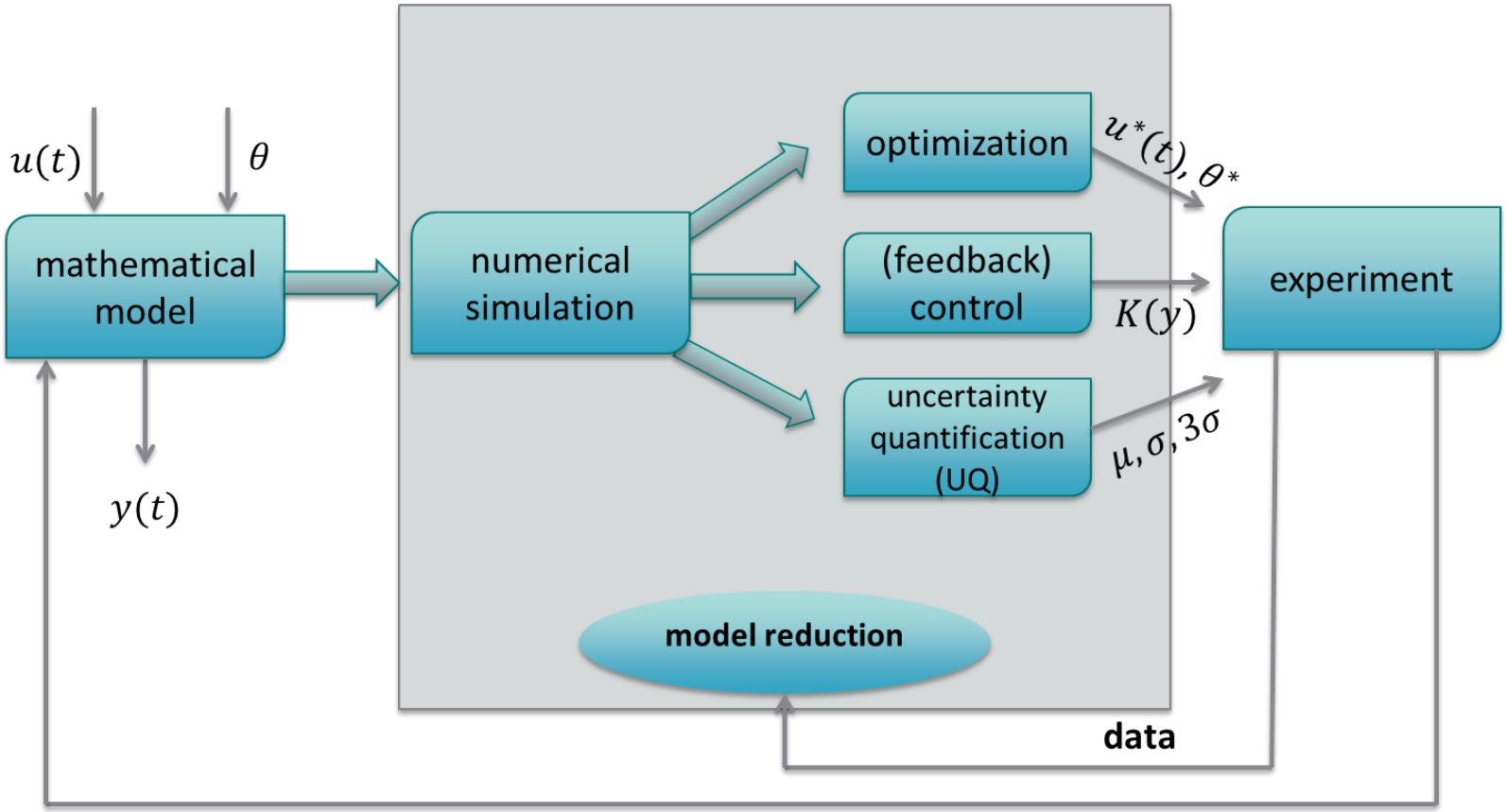
# A Process Chain in Computational Sciences and Engineering (CSE)

*Model Order Reduction (MOR)*

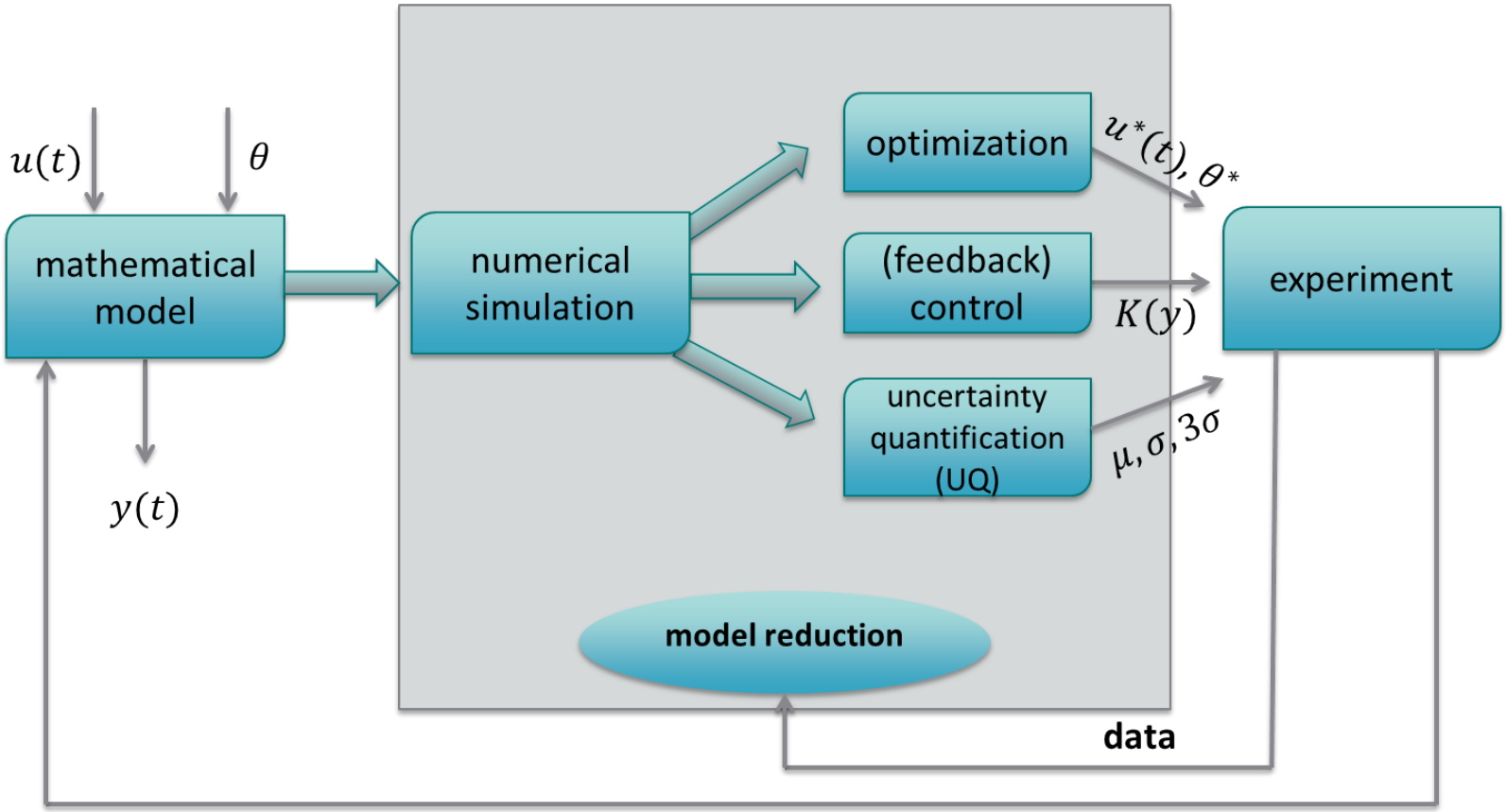


# A Process Chain in Computational Sciences and Engineering (CSE)

*Data-Driven Sciences*

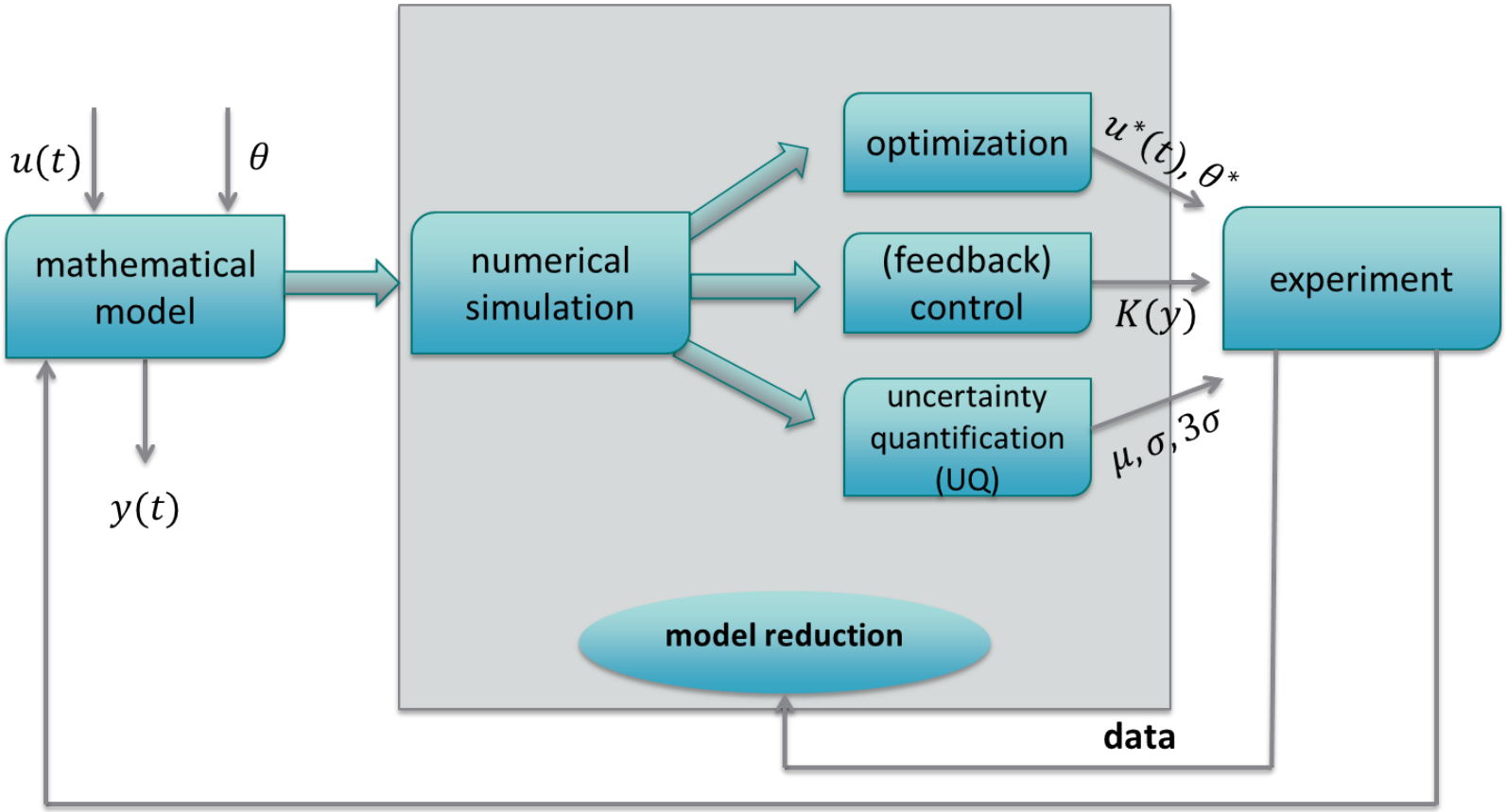


# A Process Chain in Computational Sciences and Engineering (CSE)



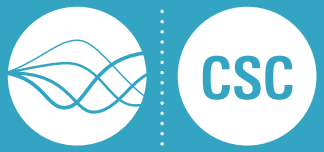

**Goal:** *Use all acquired knowledge about the model during the CSE process chain in the design of the reduced-order model, including experimental data.*

# A Process Chain in Computational Sciences and Engineering (CSE)



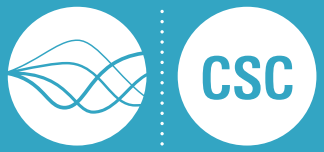

**Goal:** *Use all acquired knowledge about the model during the CSE process chain in the design of the reduced-order model, including experimental data.*

$\rightsquigarrow$ *Data-enhanced model reduction methods.*



# Outline

1. Model Order Reduction of Dynamical Systems
2. Data-driven/-enhanced Model Reduction
3. Operator Inference in Detail
4. Operator Inference for General Nonlinear Systems
5. Linear-Quadratic Residual Networks
6. Numerical Experiments



# Outline

## Original System

$$\Sigma : \begin{cases} \dot{x}(t) &= f(t, x(t), u(t)), \\ y(t) &= g(t, x(t), u(t)), \end{cases}$$

- states $x(t) \in \mathbb{R}^n$,
- inputs $u(t) \in \mathbb{R}^m$,
- outputs $y(t) \in \mathbb{R}^p$.

## Original System

$$\Sigma : \begin{cases} \dot{x}(t) &= f(t, x(t), u(t)), \\ y(t) &= g(t, x(t), u(t)), \end{cases}$$

- states $x(t) \in \mathbb{R}^n$,
- inputs $u(t) \in \mathbb{R}^m$,
- outputs $y(t) \in \mathbb{R}^p$.


## Reduced-Order Model (ROM)

$$\widehat{\Sigma} : \begin{cases} \dot{\hat{x}}(t) &= \widehat{f}(t, \hat{x}(t), u(t)), \\ \hat{y}(t) &= \widehat{g}(t, \hat{x}(t), u(t)), \end{cases}$$

- states $\hat{x}(t) \in \mathbb{R}^r$, $r \ll n$,
- inputs $u(t) \in \mathbb{R}^m$,
- outputs $\hat{y}(t) \in \mathbb{R}^p$.

# Model Order Reduction of Dynamical Systems

## Original System

$$\Sigma : \begin{cases} \dot{x}(t) &= f(t, x(t), u(t)), \\ y(t) &= g(t, x(t), u(t)), \end{cases}$$

- states $x(t) \in \mathbb{R}^n$,
- inputs $u(t) \in \mathbb{R}^m$,
- outputs $y(t) \in \mathbb{R}^p$.


## Reduced-Order Model (ROM)

$$\widehat{\Sigma} : \begin{cases} \dot{\hat{x}}(t) &= \widehat{f}(t, \hat{x}(t), u(t)), \\ \hat{y}(t) &= \widehat{g}(t, \hat{x}(t), u(t)), \end{cases}$$

- states $\hat{x}(t) \in \mathbb{R}^r$, $r \ll n$,
- inputs $u(t) \in \mathbb{R}^m$,
- outputs $\hat{y}(t) \in \mathbb{R}^p$.


## Goals:

$\|y - \hat{y}\| < \text{tolerance} \cdot \|u\|$ for all admissible input signals.

# Model Order Reduction of Dynamical Systems

## Original System

$$\Sigma : \begin{cases} \dot{x}(t) &= f(t, x(t), u(t)), \\ y(t) &= g(t, x(t), u(t)), \end{cases}$$

- states $x(t) \in \mathbb{R}^n$,
- inputs $u(t) \in \mathbb{R}^m$,
- outputs $y(t) \in \mathbb{R}^p$.


## Reduced-Order Model (ROM)

$$\widehat{\Sigma} : \begin{cases} \dot{\hat{x}}(t) &= \widehat{f}(t, \hat{x}(t), u(t)), \\ \hat{y}(t) &= \widehat{g}(t, \hat{x}(t), u(t)), \end{cases}$$

- states $\hat{x}(t) \in \mathbb{R}^r$, $r \ll n$,
- inputs $u(t) \in \mathbb{R}^m$,
- outputs $\hat{y}(t) \in \mathbb{R}^p$.


## Goals:

$\|y - \hat{y}\| < \text{tolerance} \cdot \|u\|$ for all admissible input signals.

Secondary goal: reconstruct approximation of $x$ from $\hat{x}$.

# Model Order Reduction of Linear Systems

## Linear Time-Invariant (LTI) Systems

### Original System

$$\Sigma : \begin{cases} \dot{x}(t) = Ax(t) + Bu(t), \\ y(t) = Cx(t) + Du(t). \end{cases}$$

- states $x(t) \in \mathbb{R}^n$,
- inputs $u(t) \in \mathbb{R}^m$,
- outputs $y(t) \in \mathbb{R}^p$.


### Reduced-Order Model (ROM)

$$\widehat{\Sigma} : \begin{cases} \dot{\hat{x}}(t) = \widehat{A}\hat{x}(t) + \widehat{B}u(t), \\ \hat{y}(t) = \widehat{C}\hat{x}(t) + \widehat{D}u(t). \end{cases}$$

- states $\hat{x}(t) \in \mathbb{R}^r$, $r \ll n$
- inputs $u(t) \in \mathbb{R}^m$,
- outputs $\hat{y}(t) \in \mathbb{R}^p$.


### Goals:

$\|y - \hat{y}\| < \text{tolerance} \cdot \|u\|$ for all admissible input signals.

Secondary goal: reconstruct approximation of $x$ from $\hat{x}$.

# Model Order Reduction of Linear Systems

## Model Reduction Schematically

$$E\,\dot{x}(t) = A\,x(t) + B\,u(t)$$

$$y(t) = C\,x(t) + D\,u(t)$$

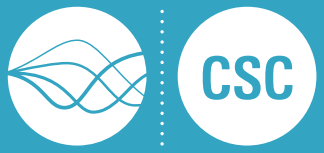
MOR

$$\hat{E}\,\dot{\hat{x}}(t) = \hat{A}\,\hat{x}(t) + \hat{B}\,u(t)$$

$$\hat{y}(t) = \hat{C}\,\hat{x}(t) + \hat{D}\,u(t)$$

- $E, A \in \mathbb{R}^{n\times n}$
- $B \in \mathbb{R}^{n\times m}$
- $C \in \mathbb{R}^{p\times n}$
- $D \in \mathbb{R}^{p\times m}$

- $\hat{E}, \hat{A} \in \mathbb{R}^{r\times r}$
- $\hat{B} \in \mathbb{R}^{r\times m}$
- $\hat{C} \in \mathbb{R}^{p\times r}$
- $\hat{D} \in \mathbb{R}^{p\times m}$



# MOR Methods Based on Projection

**Assumption:** trajectory $x(t;u)$ is contained in low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$.

# MOR Methods Based on Projection

**Assumption:** trajectory $x(t;u)$ is contained in low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$.
Thus, use Galerkin or Petrov-Galerkin-type projection of state-space onto $\mathcal{V}$ (trial space) along complementary subspace $\mathcal{W}$ (test space), where

$$\operatorname{range}(V) = \mathcal{V}, \quad \operatorname{range}(W) = \mathcal{W}, \quad W^T V = I_r.$$

# MOR Methods Based on Projection

**Assumption:** trajectory $x(t;u)$ is contained in low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$.
Thus, use Galerkin or Petrov-Galerkin-type projection of state-space onto $\mathcal{V}$ (trial space) along complementary subspace $\mathcal{W}$ (test space), where

$$\operatorname{range}(V) = \mathcal{V}, \quad \operatorname{range}(W) = \mathcal{W}, \quad W^T V = I_r.$$

Then, with $\hat{x} = W^T x$, we obtain $x \approx V\hat{x} = VW^T x =: \tilde{x}$ so that

$$||x - \tilde{x}|| = ||x - V\hat{x}||.$$

# MOR Methods Based on Projection

**Assumption:** trajectory $x(t;u)$ is contained in low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$.
Thus, use Galerkin or Petrov-Galerkin-type projection of state-space onto $\mathcal{V}$ (trial space) along complementary subspace $\mathcal{W}$ (test space), where

$$\operatorname{range}(V) = \mathcal{V}, \quad \operatorname{range}(W) = \mathcal{W}, \quad W^T V = I_r.$$

Then, with $\hat{x} = W^T x$, we obtain $x \approx V\hat{x} = VW^T x =: \tilde{x}$ so that

$$||x - \tilde{x}|| = ||x - V\hat{x}||.$$

For linear systems, the reduced-order model is

$$\hat{x} = W^T x, \quad \hat{A} := W^T AV, \quad \hat{B} := W^T B, \quad \hat{C} := CV, \quad (\hat{D} := D).$$

# MOR Methods Based on Projection

**Assumption:** trajectory $x(t;u)$ is contained in low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$.
Thus, use Galerkin or Petrov-Galerkin-type projection of state-space onto $\mathcal{V}$ (trial space) along complementary subspace $\mathcal{W}$ (test space), where

$$\operatorname{range}(V) = \mathcal{V}, \quad \operatorname{range}(W) = \mathcal{W}, \quad W^T V = I_r.$$

For linear systems, the reduced-order model is

$$\hat{x} = W^T x, \quad \hat{A} := W^T A V, \quad \hat{B} := W^T B, \quad \hat{C} := CV, \quad (\hat{D} := D).$$

Important observation:

- The state equation residual satisfies $\dot{\tilde{x}} - A\tilde{x} - Bu \perp \mathcal{W}$, since

$$W^T\left(\dot{\tilde{x}} - A\tilde{x} - Bu\right) \quad = \quad W^T\left(VW^T\dot{x} - AVW^T x - Bu\right)$$

# MOR Methods Based on Projection

**Assumption:** trajectory $x(t;u)$ is contained in low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$.
Thus, use Galerkin or Petrov-Galerkin-type projection of state-space onto $\mathcal{V}$ (trial space) along complementary subspace $\mathcal{W}$ (test space), where

$$\operatorname{range}(V) = \mathcal{V}, \quad \operatorname{range}(W) = \mathcal{W}, \quad W^T V = I_r.$$

For linear systems, the reduced-order model is

$$\hat{x} = W^T x, \quad \hat{A} := W^T A V, \quad \hat{B} := W^T B, \quad \hat{C} := CV, \quad (\hat{D} := D).$$

Important observation:

- The state equation residual satisfies $\dot{\tilde{x}} - A\tilde{x} - Bu \perp \mathcal{W}$, since

$$\begin{aligned} W^T\left(\dot{\tilde{x}} - A\tilde{x} - Bu\right) &= W^T\left(VW^T\dot{x} - AVW^T x - Bu\right) \\ &= \underbrace{W^T\dot{x}}_{\dot{\hat{x}}} - \underbrace{W^T A V}_{=\hat{A}}\underbrace{W^T x}_{=\hat{x}} - \underbrace{W^T B}_{=\hat{B}}u \end{aligned}$$

# MOR Methods Based on Projection

**Assumption:** trajectory $x(t;u)$ is contained in low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$.
Thus, use Galerkin or Petrov-Galerkin-type projection of state-space onto $\mathcal{V}$ (trial space) along complementary subspace $\mathcal{W}$ (test space), where

$$\operatorname{range}(V) = \mathcal{V}, \quad \operatorname{range}(W) = \mathcal{W}, \quad W^T V = I_r.$$

For linear systems, the reduced-order model is

$$\hat{x} = W^T x, \quad \hat{A} := W^T A V, \quad \hat{B} := W^T B, \quad \hat{C} := CV, \quad (\hat{D} := D).$$

Important observation:

- The state equation residual satisfies $\dot{\tilde{x}} - A\tilde{x} - Bu \perp \mathcal{W}$, since

$$\begin{aligned} W^T\left(\dot{\tilde{x}} - A\tilde{x} - Bu\right) &= W^T\left(VW^T\dot{x} - AVW^T x - Bu\right) \\ &= \underbrace{W^T\dot{x}}_{\dot{\hat{x}}} - \underbrace{W^T A V}_{=\hat{A}}\underbrace{W^T x}_{=\hat{x}} - \underbrace{W^T B}_{=\hat{B}}u \\ &= \dot{\hat{x}} - \hat{A}\hat{x} - \hat{B}u = 0. \end{aligned}$$

# MOR Methods Based on Projection

**Assumption:** trajectory $x(t;u)$ is contained in low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$.
Thus, use Galerkin or Petrov-Galerkin-type projection of state-space onto $\mathcal{V}$ (trial space) along complementary subspace $\mathcal{W}$ (test space), where

$$\operatorname{range}(V) = \mathcal{V}, \quad \operatorname{range}(W) = \mathcal{W}, \quad W^T V = I_r.$$

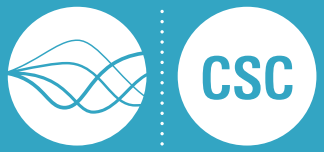
For linear systems, the reduced-order model is

$$\hat{x} = W^T x, \quad \hat{A} := W^T A V, \quad \hat{B} := W^T B, \quad \hat{C} := CV, \quad (\hat{D} := D).$$

Extends to nonlinear systems with some effort:

$$\begin{aligned} \dot{\hat{x}} &= W^T f(t, V\hat{x}, u), \\ \hat{y} &= g(t, V\hat{x}, u). \end{aligned}$$

Needs hyperreduction if the cost for evaluation of the functions $W^T f, g$ is not reduced!



# MOR Methods Based on Projection

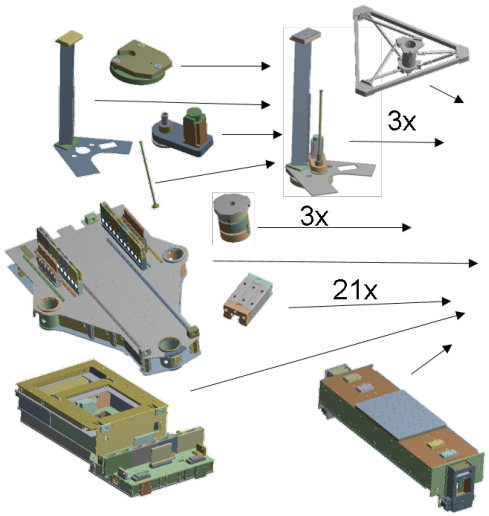
## Classes of Projection-based MOR Methods

1. Modal Truncation
2. Rational Interpolation / Moment Matching
   (Padé-Approximation and (rational) Krylov Subspace Methods)
3. Balanced Truncation
4. Proper Orthogonal Decomposition (POD) / Principal Component Analysis (PCA)
5. Reduced Basis Method
6. ...

# MOR Methods Based on Projection

## Example: Thermal model of experimental machine tool MAX

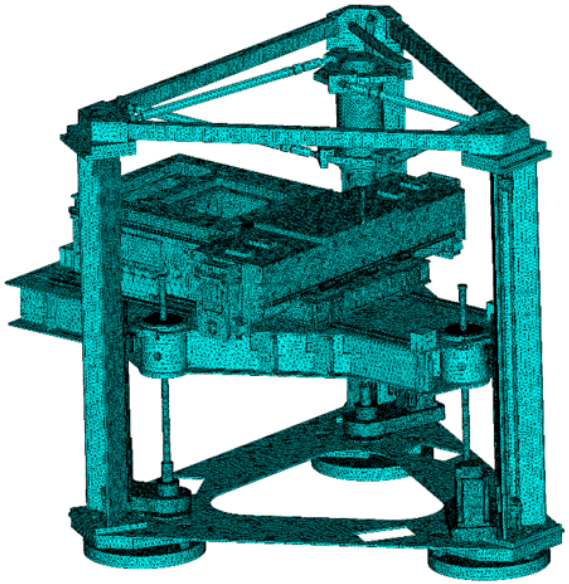
50 subassemblies




CAD model

$\overset{FEM}{\rightsquigarrow}$

FE-Model: 1.2M DOFs



TECHNISCHE UNIVERSITÄT DRESDEN · RWTH AACHEN UNIVERSITY · TECHNISCHE UNIVERSITÄT CHEMNITZ · Sonderforschungsbereich Transregio 96

# MAX: Results considering an inhomogeneous initial condition $T_0 \neq 0$

Results by Julia Vettermann (MilT/TU Chemnitz)

**FE-coupled**

| method | red. order tol $10^{-3}$ | $t_{red}$ |
|---|---|---|
| 2phase | 196 | 6.5h |
| BTX0 | 174 | 4.5h |

**output-coupled**

| method | red. order tol $10^{-3}$ | $t_{red}$ |
|---|---|---|
| 2phase | 3005 | 2h |
| BTX0 | 2515 | 1.8h |

# MAX: Results considering an inhomogeneous initial condition $T_0 \neq 0$

Results by Julia Vettermann (MilT/TU Chemnitz)

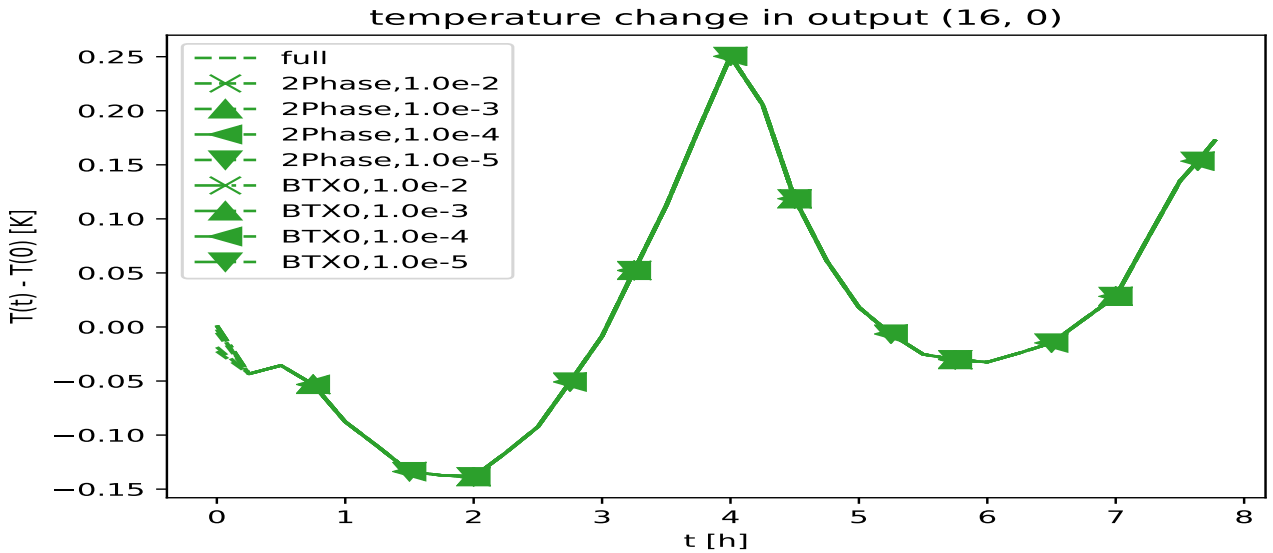
**FE-coupled**

| method | red. order tol $10^{-3}$ | $t_{red}$ |
|---|---|---|
| 2phase | 196 | 6.5h |
| BTX0 | 174 | 4.5h |

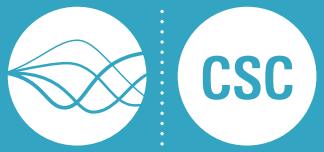
**output-coupled**

| method | red. order tol $10^{-3}$ | $t_{red}$ |
|---|---|---|
| 2phase | 3005 | 2h |
| BTX0 | 2515 | 1.8h |

→ *Required storage for reduced matrices just 1MB!*
→ *Simulation speed-up factors range from* $\approx 8$–$2,000$.




Vettermann, J., Sauerzapf, S., Naumann, A., Beitelschmidt, M., Herzog, R., Benner, P., Saak, J. (2021): Model order reduction methods for coupled machine tool models. MM Science Journal, pp. 4652–4659.



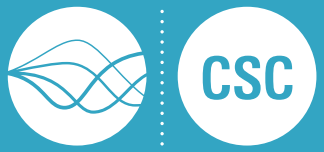
# What about the Data?

**Assumption:** trajectory $x(t;u)$ is contained in low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$.
Thus, use Galerkin or Petrov-Galerkin-type projection of state-space onto $\mathcal{V}$ (trial space) along complementary subspace $\mathcal{W}$ (test space), where

$$\operatorname{range}(V) = \mathcal{V}, \quad \operatorname{range}(W) = \mathcal{W}, \quad W^T V = I_r.$$

The reduced-order model is

$$\hat{x} = W^T x, \quad \hat{A} := W^T A V, \quad \hat{B} := W^T B, \quad \hat{C} := CV, \quad (\hat{D} := D).$$



# What about the Data?

**Assumption:** trajectory $x(t;u)$ is contained in low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$.
Thus, use Galerkin or Petrov-Galerkin-type projection of state-space onto $\mathcal{V}$ (trial space) along complementary subspace $\mathcal{W}$ (test space), where

$$\operatorname{range}(V) = \mathcal{V}, \quad \operatorname{range}(W) = \mathcal{W}, \quad W^T V = I_r.$$

The reduced-order model is

$$\hat{x} = W^T x, \quad \hat{A} := W^T A V, \quad \hat{B} := W^T B, \quad \hat{C} := CV, \quad (\hat{D} := D).$$

We need the matrices $A, B, C, D$ to compute the reduced-order model!

# What about the Data?

**Assumption:** trajectory $x(t;u)$ is contained in low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$.
Thus, use Galerkin or Petrov-Galerkin-type projection of state-space onto $\mathcal{V}$ (trial space) along complementary subspace $\mathcal{W}$ (test space), where

$$\operatorname{range}(V) = \mathcal{V}, \quad \operatorname{range}(W) = \mathcal{W}, \quad W^T V = I_r.$$

The reduced-order model is

$$\hat{x} = W^T x, \quad \hat{A} := W^T A V, \quad \hat{B} := W^T B, \quad \hat{C} := CV, \quad (\hat{D} := D).$$

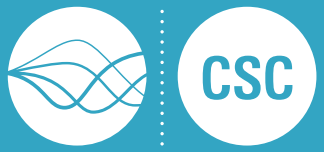
We need the matrices $A, B, C, D$ to compute the reduced-order model!

Using proprietary simulation software, we would need to **intrude** the software to get the matrices

$\rightsquigarrow$ **intrusive MOR**

$=$ *learning (compact, surrogate) models from (full, detailed) models.*

This is often impossible!



# What about the Data?

**Assumption:** trajectory $x(t;u)$ is contained in low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^n$.
Thus, use Galerkin or Petrov-Galerkin-type projection of state-space onto $\mathcal{V}$ (trial space) along complementary subspace $\mathcal{W}$ (test space), where

$$\operatorname{range}(V) = \mathcal{V}, \quad \operatorname{range}(W) = \mathcal{W}, \quad W^T V = I_r.$$

The reduced-order model is

$$\hat{x} = W^T x, \quad \hat{A} := W^T A V, \quad \hat{B} := W^T B, \quad \hat{C} := CV, \quad (\hat{D} := D).$$

We need the matrices $A, B, C, D$ to compute the reduced-order model!

Using proprietary simulation software, we would need to **intrude** the software to get the matrices

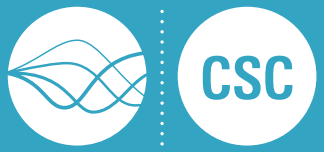
$\rightsquigarrow$ **intrusive MOR**

$=$ *learning (compact, surrogate) models from (full, detailed) models.*

This is often impossible!

$\rightsquigarrow$ **non-intrusive MOR**

$=$ *LEARNING (compact, surrogate) MODELS FROM DATA!*



# Outline

Now assume we are only given an oracle, allowing us to evaluate $\Sigma$, given $u(t)$ or $U(s)$:

Now assume we are only given an oracle, allowing us to evaluate $\Sigma$, given $u(t)$ or $U(s)$:


**Black box $\Sigma$:** the only information we can get is either

- time domain data / times series: $u_k \approx u(t_k)$ and $x_k \approx x(t_k)$ or $y_k \approx y(t_k)$, or

Now assume we are only given an oracle, allowing us to evaluate $\Sigma$, given $u(t)$ or $U(s)$:


**Black box $\Sigma$:** the only information we can get is either

- time domain data / times series: $u_k \approx u(t_k)$ and $x_k \approx x(t_k)$ or $y_k \approx y(t_k)$, or
- frequency domain data / measurements: $U_k \approx U(\jmath\omega_k)$ and $X_k \approx X(\jmath\omega_k)$ or $Y_k \approx Y(\jmath\omega_k)$.

Now assume we are only given an oracle, allowing us to evaluate $\Sigma$, given $u(t)$ or $U(s)$:


**Black box $\Sigma$:** the only information we can get is either

- time domain data / times series: $u_k \approx u(t_k)$ and $x_k \approx x(t_k)$ or $y_k \approx y(t_k)$, or
- frequency domain data / measurements: $U_k \approx U(\jmath\omega_k)$ and $X_k \approx X(\jmath\omega_k)$ or $Y_k \approx Y(\jmath\omega_k)$.

Some methods:

- System identification (incl. ERA, N4SID, MOESP): frequency and time domain
  [Ho/Kalman 1966; Ljung 1987/1999; Van Overschee/De Moor 1994; Verhaegen 1994; De Wilde, Eykhoff, Moonen, Sima, ...]

Now assume we are only given an oracle, allowing us to evaluate $\Sigma$, given $u(t)$ or $U(s)$:


**Black box $\Sigma$:** the only information we can get is either

- time domain data / times series: $u_k \approx u(t_k)$ and $x_k \approx x(t_k)$ or $y_k \approx y(t_k)$, or
- frequency domain data / measurements: $U_k \approx U(\jmath\omega_k)$ and $X_k \approx X(\jmath\omega_k)$ or $Y_k \approx Y(\jmath\omega_k)$.

Some methods:

- System identification (incl. ERA, N4SID, MOESP): frequency and time domain
  [Ho/Kalman 1966; Ljung 1987/1999; Van Overschee/De Moor 1994; Verhaegen 1994; De Wilde, Eykhoff, Moonen, Sima, ...]
- Neural networks: time domain [Narendra/Parthasarathy 1990; Lee/Carlberg 2019; ...]

Now assume we are only given an oracle, allowing us to evaluate $\Sigma$, given $u(t)$ or $U(s)$:


**Black box $\Sigma$:** the only information we can get is either

- time domain data / times series: $u_k \approx u(t_k)$ and $x_k \approx x(t_k)$ or $y_k \approx y(t_k)$, or
- frequency domain data / measurements: $U_k \approx U(\jmath\omega_k)$ and $X_k \approx X(\jmath\omega_k)$ or $Y_k \approx Y(\jmath\omega_k)$.

Some methods:

- System identification (incl. ERA, N4SID, MOESP): frequency and time domain
  [Ho/Kalman 1966; Ljung 1987/1999; Van Overschee/De Moor 1994; Verhaegen 1994; De Wilde, Eykhoff, Moonen, Sima, ...]
- Neural networks: time domain [Narendra/Parthasarathy 1990; Lee/Carlberg 2019; ...]
- Loewner interpolation: frequency and time domain
  [Antoulas/Anderson 1986; Mayo/Antoulas 2007; Gosea, Gugercin, Ionita, Lefteriu, Peherstorfer, ...]

Now assume we are only given an oracle, allowing us to evaluate $\Sigma$, given $u(t)$ or $U(s)$:


**Black box $\Sigma$:** the only information we can get is either

- time domain data / times series: $u_k \approx u(t_k)$ and $x_k \approx x(t_k)$ or $y_k \approx y(t_k)$, or
- frequency domain data / measurements: $U_k \approx U(\jmath\omega_k)$ and $X_k \approx X(\jmath\omega_k)$ or $Y_k \approx Y(\jmath\omega_k)$.

Some methods:

- System identification (incl. ERA, N4SID, MOESP): frequency and time domain
  [Ho/Kalman 1966; Ljung 1987/1999; Van Overschee/De Moor 1994; Verhaegen 1994; De Wilde, Eykhoff, Moonen, Sima, ...]
- Neural networks: time domain [Narendra/Parthasarathy 1990; Lee/Carlberg 2019; ...]
- Loewner interpolation: frequency and time domain
  [Antoulas/Anderson 1986; Mayo/Antoulas 2007; Gosea, Gugercin, Ionita, Lefteriu, Peherstorfer, ...]
- Koopman/Dynamic Mode Decomposition (DMD): time domain
  [Mezič 2005; Schmid 2008; Brunton, Kevrekidis, Kutz, Rowley, Noé, Nüske, Schütte, Peitz, ...],
  for control systems [Kaiser/Kutz/Brunton 2017, B./Himpe/Mitchell 2018]

Now assume we are only given an oracle, allowing us to evaluate $\Sigma$, given $u(t)$ or $U(s)$:


**Black box $\Sigma$:** the only information we can get is either

- time domain data / times series: $u_k \approx u(t_k)$ and $x_k \approx x(t_k)$ or $y_k \approx y(t_k)$, or
- frequency domain data / measurements: $U_k \approx U(\jmath\omega_k)$ and $X_k \approx X(\jmath\omega_k)$ or $Y_k \approx Y(\jmath\omega_k)$.

Some methods:

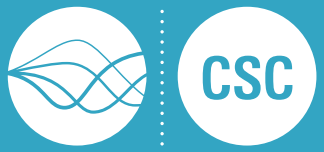
- System identification (incl. ERA, N4SID, MOESP): frequency and time domain
  [Ho/Kalman 1966; Ljung 1987/1999; Van Overschee/De Moor 1994; Verhaegen 1994; De Wilde, Eykhoff, Moonen, Sima, ...]
- Neural networks: time domain [Narendra/Parthasarathy 1990; Lee/Carlberg 2019; ...]
- Loewner interpolation: frequency and time domain
  [Antoulas/Anderson 1986; Mayo/Antoulas 2007; Gosea, Gugercin, Ionita, Lefteriu, Peherstorfer, ...]
- Koopman/Dynamic Mode Decomposition (DMD): time domain
  [Mezič 2005; Schmid 2008; Brunton, Kevrekidis, Kutz, Rowley, Noé, Nüske, Schütte, Peitz, ...],
  for control systems [Kaiser/Kutz/Brunton 2017, B./Himpe/Mitchell 2018]
- Operator inference (OpInf): time domain [Peherstorfer/Willcox 2016; Kramer, Qian, B., Goyal,...]

# A few Remarks on System Identification and DNNs

- System identification tries to infer discrete linear time-invariant (LTI) systems

$$x_{k+1} = Ax_k + Bu_k + Kw_k,$$
$$y_k = Cx_k + Du_k + v_k.$$

from input-output data, given as time series $(u_0, y_0), (u_1, y_1), \ldots, (u_K, y_K)$, where $v_k, w_k$ are uncorrelated Gaussian white noise processes.



# A few Remarks on System Identification and DNNs

- System identification tries to infer discrete linear time-invariant (LTI) systems

$$x_{k+1} = Ax_k + Bu_k + Kw_k,$$
$$y_k = Cx_k + Du_k + v_k.$$

  from input-output data, given as time series $(u_0, y_0), (u_1, y_1), \ldots, (u_K, y_K)$, where $v_k, w_k$ are uncorrelated Gaussian white noise processes.
- Early survey already 1971: Aström/Eykhoff, AUTOMATICA 7(2):123–162.

# A few Remarks on System Identification and DNNs

- System identification tries to infer discrete linear time-invariant (LTI) systems

$$x_{k+1} = Ax_k + Bu_k + Kw_k,$$
$$y_k = Cx_k + Du_k + v_k.$$

  from input-output data, given as time series $(u_0, y_0), (u_1, y_1), \ldots, (u_K, y_K)$, where $v_k, w_k$ are uncorrelated Gaussian white noise processes.

- Early survey already 1971: Aström/Eykhoff, Automatica 7(2):123–162.
- Popular methods are
  1. MOESP — Multivariable Output Error State-SPace [Verhaegen/Dewilde 1992],
  2. N4SID — Numerical algorithm for Subspace State Space System IDentification [Van Overschee/De Moor 1994].

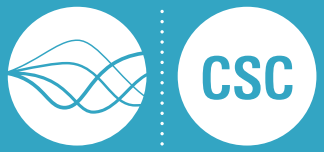
# A few Remarks on System Identification and DNNs

- System identification tries to infer discrete linear time-invariant (LTI) systems

$$x_{k+1} = Ax_k + Bu_k + Kw_k,$$
$$y_k = Cx_k + Du_k + v_k.$$

  from input-output data, given as time series $(u_0, y_0), (u_1, y_1), \ldots, (u_K, y_K)$, where $v_k, w_k$ are uncorrelated Gaussian white noise processes.

- Early survey already 1971: Aström/Eykhoff, Automatica 7(2):123–162.
- Popular methods are
  1. MOESP — Multivariable Output Error State-SPace [Verhaegen/Dewilde 1992],
  2. N4SID — Numerical algorithm for Subspace State Space System IDentification [Van Overschee/De Moor 1994].
- Both are based on decompositions of certain block-Hankel matrices built from the input-output data and are available in standard software packages like the MATLAB System Identification Toolbox and SLICOT.



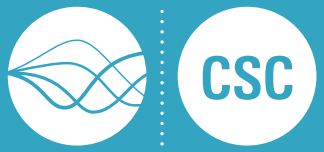
# A few Remarks on System Identification and DNNs

- System identification tries to infer discrete linear time-invariant (LTI) systems

$$x_{k+1} = Ax_k + Bu_k + Kw_k,$$
$$y_k = Cx_k + Du_k + v_k.$$

  from input-output data, given as time series $(u_0, y_0), (u_1, y_1), \ldots, (u_K, y_K)$, where $v_k, w_k$ are uncorrelated Gaussian white noise processes.

- Early survey already 1971: Aström/Eykhoff, Automatica 7(2):123–162.
- Popular methods are
  1. MOESP — Multivariable Output Error State-SPace [Verhaegen/Dewilde 1992],
  2. N4SID — Numerical algorithm for Subspace State Space System IDentification [Van Overschee/De Moor 1994].
- Both are based on decompositions of certain block-Hankel matrices built from the input-output data and are available in standard software packages like the MATLAB System Identification Toolbox and SLICOT.
- Continuous-time system can be identified, e.g., by "inverse" Euler method.



# A few Remarks on System Identification and DNNs

- System identification tries to infer discrete linear time-invariant (LTI) systems

$$x_{k+1} = Ax_k + Bu_k + Kw_k,$$
$$y_k = Cx_k + Du_k + v_k.$$

  from input-output data, given as time series $(u_0, y_0), (u_1, y_1), \ldots, (u_K, y_K)$, where $v_k, w_k$ are uncorrelated Gaussian white noise processes.

- Early survey already 1971: Aström/Eykhoff, Automatica 7(2):123–162.
- Popular methods are
  1. MOESP — Multivariable Output Error State-SPace [Verhaegen/Dewilde 1992],
  2. N4SID — Numerical algorithm for Subspace State Space System IDentification [Van Overschee/De Moor 1994].
- Both are based on decompositions of certain block-Hankel matrices built from the input-output data and are available in standard software packages like the MATLAB System Identification Toolbox and SLICOT.
- Continuous-time system can be identified, e.g., by "inverse" Euler method.
- Many extensions to nonlinear systems, imposing certain structural assumptions, including artificial neural networks...

# A few Remarks on System Identification and DNNs

## A paper from 1990...



### Identification and Control of Dynamical Systems Using Neural Networks

KUMPATI S. NARENDRA FELLOW, IEEE, AND KANNAN PARTHASARATHY

*Abstract*—**The paper demonstrates that neural networks can be used effectively for the identification and control of nonlinear dynamical systems. The emphasis of the paper is on models for both identification and control. Static and dynamic back-propagation methods for the adjustment of parameters are discussed. In the models that are introduced, multilayer and recurrent networks are interconnected in novel configurations and hence there is a real need to study them in a unified fashion. Simulation results reveal that the identification and adaptive control schemes suggested are practically feasible. Basic concepts and definitions are introduced throughout the paper, and theoretical questions which have to be addressed are also described.**

are well known for such systems [1]. In this paper our interest is in the identification and control of nonlinear dynamic plants using neural networks. Since very few results exist in nonlinear systems theory which can be directly applied, considerable care has to be exercised in the statement of the problems, the choice of the identifier and controller structures, as well as the generation of adaptive laws for the adjustment of the parameters.

Two classes of neural networks which have received considerable attention in the area of artificial neural net-

Narendra, K.S., Parthasarathy, K. (1990): Identification and control of dynamical systems using neural networks. IEEE Transactions on Neural Networks 1(1):4–27.

# A few Remarks on System Identification and DNNs

## A paper from 1990...

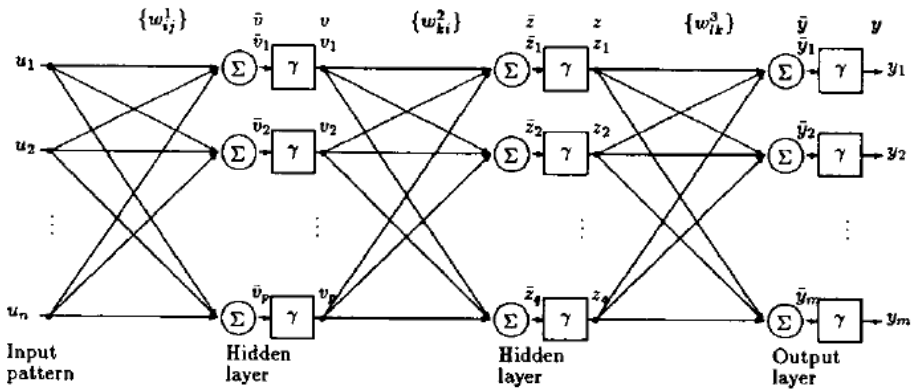



Fig. 2. A three layer neural network.

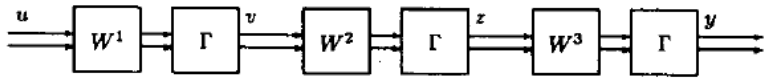
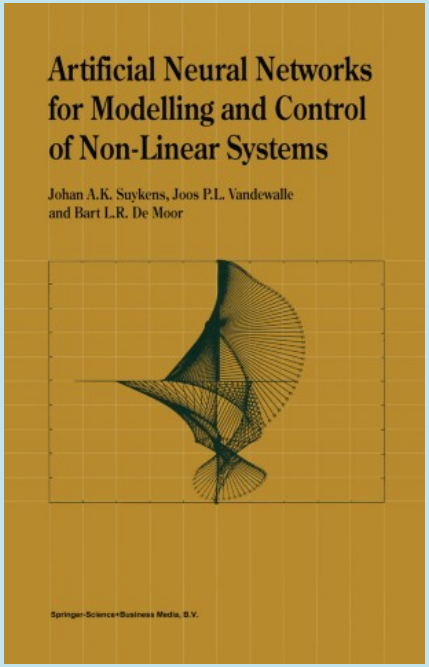



Fig. 3. A block diagram representation of a three layer network.

Narendra, K.S., Parthasarathy, K. (1990): Identification and control of dynamical systems using neural networks. IEEE Transactions on Neural Networks 1(1):4–27.

# A few Remarks on System Identification and DNNs

## A book from 1996...




Narendra, K.S., Parthasarathy, K. (1990): Identification and control of dynamical systems using neural networks. IEEE Transactions on Neural Networks 1(1):4–27.

Suykens, J.A.K., Vandewalle, J.P.L., de Moor, B.L. (1996): Artificial Neural Networks for Modelling and Control of Non-Linear Systems. Springer US.

# DMD in a Nutshell

## Basic Framework

Given a smooth dynamical system

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0 \in \mathbb{R}^n.$$

Take snapshots $x_k := x(t_k)$ on grid $t_k := kh$ for $k = 0, 1, \ldots, K$ and fixed $h > 0$ (using simulation software, or measurements from real life experiment $\rightsquigarrow$ nonintrusive!), and find "best possible" $A_*$ such that

$$x_{k+1} \approx A_* x_k.$$

Given a smooth dynamical system

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0 \in \mathbb{R}^n.$$

Take snapshots $x_k := x(t_k)$ on grid $t_k := kh$ for $k = 0, 1, \ldots, K$ and fixed $h > 0$ (using simulation software, or measurements from real life experiment $\rightsquigarrow$ nonintrusive!), and find "best possible" $A_*$ such that

$$x_{k+1} \approx A_* x_k.$$

**Motivation:** Koopman theory

- $\exists$ a linear, infinite-dimensional operator describing the evolution of $f(x(\cdot))$ in an appropriate function space setting.
- Can be considered as lifting of a finite-dimensional, nonlinear problem to a infinite-dimensional, linear problem.

# DMD in a Nutshell

## Basic Framework

Given a smooth dynamical system

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0 \in \mathbb{R}^n.$$

Take snapshots $x_k := x(t_k)$ on grid $t_k := kh$ for $k = 0, 1, \ldots, K$ and fixed $h > 0$ (using simulation software, or measurements from real life experiment $\rightsquigarrow$ nonintrusive!), and find "best possible" $A_*$ such that

$$x_{k+1} \approx A_* x_k.$$

**Motivation:** Koopman theory

- $\exists$ a linear, infinite-dimensional operator describing the evolution of $f(x(\cdot))$ in an appropriate function space setting.
- Can be considered as lifting of a finite-dimensional, nonlinear problem to a infinite-dimensional, linear problem.

### Basic DMD Algorithm

Set $X_0 := [\, x_0, x_1, \ldots, x_{K-1} \,] \in \mathbb{R}^{n \times K}$, $X_1 := [\, x_1, x_2, \ldots, x_K \,] \in \mathbb{R}^{n \times K}$ and note that $X_1 = AX_0$ is desired $\rightsquigarrow$ over-/underdetermined linear system, solved by linear least-squares problem (regression):

$$A_* := \arg\min_{A \in \mathbb{R}^{n \times n}} \|X_1 - AX_0\|_F + \beta \|A\|_q$$

with a potential regularization term choosing $\beta > 0$, $q = 0, 1, 2$.

Computation usually via singular value decomposition (SVD), many variants.

# DMD in a Nutshell

## DMD with Inputs and Outputs

Given a smooth control system

$$\dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0 \in \mathbb{R}^n, \qquad y(t) = g(x(t), u(t)),$$

with control $u(t) \in \mathbb{R}^m$ and output $y(t) \in \mathbb{R}^p$.

# DMD in a Nutshell

## DMD with Inputs and Outputs

Given a smooth control system

$$\dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0 \in \mathbb{R}^n, \qquad\qquad y(t) = g(x(t), u(t)),$$

with control $u(t) \in \mathbb{R}^m$ and output $y(t) \in \mathbb{R}^p$.

Take state, control, and output snapshots

$$x_k := x(t_k), \quad u_k := u(t_k), \quad y_k := y(t_k), \qquad k = 0, 1, \ldots, K$$

(using simulation software, or measurements from real life experiment $\rightsquigarrow$ nonintrusive!), and find "best possible" discrete-time LTI system such that

$$x_{k+1} \approx A_* x_k + B_* u_k, \qquad y_k \approx C_* x_k + D_* u_k.$$

Given a smooth control system

$$\dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0 \in \mathbb{R}^n, \qquad y(t) = g(x(t), u(t)),$$

with control $u(t) \in \mathbb{R}^m$ and output $y(t) \in \mathbb{R}^p$.

Take state, control, and output snapshots

$$x_k := x(t_k), \quad u_k := u(t_k), \quad y_k := y(t_k), \qquad k = 0, 1, \ldots, K$$

(using simulation software, or measurements from real life experiment $\rightsquigarrow$ nonintrusive!), and find "best possible" discrete-time LTI system such that

$$x_{k+1} \approx A_* x_k + B_* u_k, \qquad y_k \approx C_* x_k + D_* u_k.$$

## Basic ioDMD Algorithm ($\equiv$ N4SID)

Let $\mathbb{S} := \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times m} \times \mathbb{R}^{p\times n} \times \mathbb{R}^{p\times m}$. Set $X_0, X_1$ as before and

$$U_0 := [\, u_0, u_1, \ldots, u_{K-1} \,] \in \mathbb{R}^{m\times K}, \qquad Y_0 := [\, y_0, y_1, \ldots, y_{K-1} \,] \in \mathbb{R}^{p\times K}.$$

Solve the linear least-squares problem (regression):

$$(A_*, B_*, C_*, D_*) := \arg\min_{(A,B,C,D)\in\mathbb{S}} \left\| \begin{bmatrix} X_1 \\ Y_0 \end{bmatrix} - \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} X_0 \\ U_0 \end{bmatrix} \right\|_F + \beta \| [\, A\, B\, C\, D \,] \|_q$$

with a potential regularization term choosing $\beta > 0$, $q = 0, 1, 2$.

# DMD in a Nutshell

## Selected References (Chronological)


Koopman, B.O. (1931): Hamiltonian systems and transformation in Hilbert space. Proc. Natl. Acad. Sci. 17(5):315—381.

Mezić, I. (2005): Spectral properties of dynamical systems, model reduction and decompositions. Nonlinear Dyn. 41(1):309—325. 10.1007/s11071-005-2824-x

Schmid, P.J. (2010): Dynamic mode decomposition of numerical and experimental data. J. Fluid Mech. 656:5—28. 10.1017/S0022112010001217

Kutz, J.N., Brunton, S.L., Brunton, B.W., Proctor, J.L. (2016): Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems. SIAM, Philadelphia.

Proctor, J.L., Brunton, S.L., Kutz, J.N. (2016): Dynamic mode decomposition with control. SIAM J. Appl. Dyn. Syst. 15(1):142—161. 10.1137/15M1013857

Benner, P., Himpe, C., Mitchell, T. (2018): On reduced input-output dynamic mode decomposition. Adv. Comp. Math. 44(6):1751–1768. 10.1007/s10444-018-9592-x

Gosea, I.V., Pontes Duff, I. (2020): Toward fitting structured nonlinear systems by means of dynamic mode decomposition. arXiv:2003.06484.

Mauroy, A., Mezić, I., Susuki, Y., eds., (2020): The Koopman Operator in Systems and Control. Concepts, Methodologies, and Applications. LNCIS 484, Springer, Cham.

# Operator Inference

## Reduced-order / compressive DMD

Same setting as before: given a smooth dynamical system

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0 \in \mathbb{R}^n.$$

Take snapshots $x_k := x(t_k)$ on grid $t_k := kh$ for $k = 0, 1, \ldots, K$ and fixed $h > 0$ (using simulation software, or measurements from real life experiment $\rightsquigarrow$ nonintrusive!).

# Operator Inference

## Reduced-order / compressive DMD

Same setting as before: given a smooth dynamical system

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0 \in \mathbb{R}^n.$$

Take snapshots $x_k := x(t_k)$ on grid $t_k := kh$ for $k = 0, 1, \ldots, K$ and fixed $h > 0$ (using simulation software, or measurements from real life experiment $\rightsquigarrow$ nonintrusive!).

By construction, DMD yields a linear system of order $n$ — this may be too large!

Same setting as before: given a smooth dynamical system

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0 \in \mathbb{R}^n.$$

Take snapshots $x_k := x(t_k)$ on grid $t_k := kh$ for $k = 0, 1, \ldots, K$ and fixed $h > 0$ (using simulation software, or measurements from real life experiment $\rightsquigarrow$ nonintrusive!).

By construction, DMD yields a linear system of order $n$ — this may be too large!

**Idea:** compress trajectories using POD / PCA:

1. Let $X := [\, x_0, x_1, \ldots, x_{K-1}, x_K \,] \in \mathbb{R}^{n \times K+1}$ be the matrix of all snapshots.

Same setting as before: given a smooth dynamical system

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0 \in \mathbb{R}^n.$$

Take snapshots $x_k := x(t_k)$ on grid $t_k := kh$ for $k = 0, 1, \ldots, K$ and fixed $h > 0$ (using simulation software, or measurements from real life experiment $\rightsquigarrow$ nonintrusive!).

By construction, DMD yields a linear system of order $n$ — this may be too large!

**Idea:** compress trajectories using POD / PCA:

1. Let $X := [\, x_0, x_1, \ldots, x_{K-1}, x_K \,] \in \mathbb{R}^{n \times K+1}$ be the matrix of all snapshots.
2. Compute principal / dominant singular vectors via SVD $X = U\Sigma V^T$ and set $W := U(:, 1 : r)$ such that $\sum_{k=r+1}^{K+1} \sigma_k < \varepsilon$ (potentially, use centered data).

# Operator Inference

## Reduced-order / compressive DMD

Same setting as before: given a smooth dynamical system

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0 \in \mathbb{R}^n.$$

Take snapshots $x_k := x(t_k)$ on grid $t_k := kh$ for $k = 0, 1, \ldots, K$ and fixed $h > 0$ (using simulation software, or measurements from real life experiment $\rightsquigarrow$ nonintrusive!).

By construction, DMD yields a linear system of order $n$ — this may be too large!

**Idea:** compress trajectories using POD / PCA:

1. Let $X := [\, x_0, x_1, \ldots, x_{K-1}, x_K \,] \in \mathbb{R}^{n \times K+1}$ be the matrix of all snapshots.
2. Compute principal / dominant singular vectors via SVD $X = U\Sigma V^T$ and set $W := U(:, 1 : r)$ such that $\sum_{k=r+1}^{K+1} \sigma_k < \varepsilon$ (potentially, use centered data).
3. Compute compressed snapshot matrix $\hat{X} := W^T X$.

Same setting as before: given a smooth dynamical system

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0 \in \mathbb{R}^n.$$

Take snapshots $x_k := x(t_k)$ on grid $t_k := kh$ for $k = 0, 1, \ldots, K$ and fixed $h > 0$ (using simulation software, or measurements from real life experiment $\rightsquigarrow$ nonintrusive!).

By construction, DMD yields a linear system of order $n$ — this may be too large!

**Idea:** compress trajectories using POD / PCA:

1. Let $X := [\, x_0, x_1, \ldots, x_{K-1}, x_K \,] \in \mathbb{R}^{n \times K+1}$ be the matrix of all snapshots.
2. Compute principal / dominant singular vectors via SVD $X = U\Sigma V^T$ and set $W := U(:, 1:r)$ such that $\sum_{k=r+1}^{K+1} \sigma_k < \varepsilon$ (potentially, use centered data).
3. Compute compressed snapshot matrix $\hat{X} := W^T X$.
4. Apply DMD using $\hat{X}_0, \hat{X}_1$ and compute reduced-order $\hat{A}$ via

$$\hat{A}_* := \arg\min_{\hat{A} \in \mathbb{R}^{r \times r}} \|\hat{X}_1 - \hat{A}\hat{X}_0\|_F + \beta \|\hat{A}\|_q.$$

# Operator Inference

## Reduced-order / compressive DMD

Same setting as before: given a smooth dynamical system

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0 \in \mathbb{R}^n.$$

Take snapshots $x_k := x(t_k)$ on grid $t_k := kh$ for $k = 0, 1, \ldots, K$ and fixed $h > 0$ (using simulation software, or measurements from real life experiment $\rightsquigarrow$ nonintrusive!).

By construction, DMD yields a linear system of order $n$ — this may be too large!

**Idea:** compress trajectories using POD / PCA:

1. Let $X := [\, x_0, x_1, \ldots, x_{K-1}, x_K \,] \in \mathbb{R}^{n \times K+1}$ be the matrix of all snapshots.
2. Compute principal / dominant singular vectors via SVD $X = U\Sigma V^T$ and set $W := U(:, 1:r)$ such that $\sum_{k=r+1}^{K+1} \sigma_k < \varepsilon$ (potentially, use centered data).
3. Compute compressed snapshot matrix $\hat{X} := W^T X$.
4. Apply DMD using $\hat{X}_0, \hat{X}_1$ and compute reduced-order $\hat{A}$ via

$$\hat{A}_* := \arg\min_{\hat{A} \in \mathbb{R}^{r \times r}} \|\hat{X}_1 - \hat{A}\hat{X}_0\|_F + \beta \|\hat{A}\|_q.$$

Can be combined with ioDMD to obtain reduced-order LTI system.

# Operator Inference

## Basic Algorithm

**Basic idea:** apply compressive ioDMD in continuous-time setting,

$$\dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0 \in \mathbb{R}^n,$$

and impose a nonlinear structure.

**Basic idea:** apply compressive ioDMD in continuous-time setting,

$$\dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0 \in \mathbb{R}^n,$$

and impose a nonlinear structure.

Here: try to infer quadratic system

$$\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \hat{H}\left(\hat{x}(t) \otimes \hat{x}(t)\right) + \hat{B}u(t),$$

where $P \otimes Q := [p_{ij}Q]_{ij}$ denotes the Kronecker (tensor) product, from data

$$X := [\, x_0, x_1, \ldots, x_K \,] \in \mathbb{R}^{n\times(K+1)}, \quad U := [\, u_0, u_1, \ldots, u_K \,] \in \mathbb{R}^{m\times(K+1)}.$$

# Operator Inference
## Basic Algorithm

**Basic idea:** apply compressive ioDMD in continuous-time setting,

$$\dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0 \in \mathbb{R}^n,$$

and impose a nonlinear structure.

Here: try to infer quadratic system

$$\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \hat{H}\left(\hat{x}(t) \otimes \hat{x}(t)\right) + \hat{B}u(t),$$

where $P \otimes Q := [p_{ij}Q]_{ij}$ denotes the Kronecker (tensor) product, from data

$$X := [\, x_0, x_1, \ldots, x_K \,] \in \mathbb{R}^{n\times(K+1)}, \quad U := [\, u_0, u_1, \ldots, u_K \,] \in \mathbb{R}^{m\times(K+1)}.$$

- Use compressed trajectories (via POD / PCA) $\rightsquigarrow \hat{X}$.

**Basic idea:** apply compressive ioDMD in continuous-time setting,

$$\dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0 \in \mathbb{R}^n,$$

and impose a nonlinear structure.

Here: try to infer quadratic system

$$\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \hat{H}\left(\hat{x}(t) \otimes \hat{x}(t)\right) + \hat{B}u(t),$$

where $P \otimes Q := [p_{ij}Q]_{ij}$ denotes the Kronecker (tensor) product, from data

$$X := [\, x_0, x_1, \ldots, x_K \,] \in \mathbb{R}^{n\times(K+1)}, \quad U := [\, u_0, u_1, \ldots, u_K \,] \in \mathbb{R}^{m\times(K+1)}.$$

- Use compressed trajectories (via POD / PCA) $\rightsquigarrow \hat{X}$.
- Compress snapshot matrix of time derivatives: if residuals $f(t_j, u_j)$ are available

  $$\dot{\hat{X}} := [\, \dot{x}(0), \dot{x}(t_1), \ldots, \dot{x}(t_K) \,] \approx [\, f(x_0, u_0), f(x_1, u_1), \ldots, f(x_K, u_K) \,] \in \mathbb{R}^{n\times(K+1)},$$

  otherwise, approximate time-derivatives by finite differences $\rightsquigarrow \dot{\hat{X}}$.

**Basic idea:** apply compressive ioDMD in continuous-time setting,

$$\dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0 \in \mathbb{R}^n,$$

and impose a nonlinear structure.

Here: try to infer quadratic system

$$\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \hat{H}\left(\hat{x}(t) \otimes \hat{x}(t)\right) + \hat{B}u(t),$$

where $P \otimes Q := [p_{ij}Q]_{ij}$ denotes the Kronecker (tensor) product, from data

$$X := [\, x_0, x_1, \ldots, x_K \,] \in \mathbb{R}^{n\times(K+1)}, \quad U := [\, u_0, u_1, \ldots, u_K \,] \in \mathbb{R}^{m\times(K+1)}.$$

- Use compressed trajectories (via POD / PCA) $\rightsquigarrow \hat{X}$.
- Compress snapshot matrix of time derivatives: if residuals $f(t_j, u_j)$ are available

  $$\dot{\hat{X}} := [\, \dot{x}(0), \dot{x}(t_1), \ldots, \dot{x}(t_K) \,] \approx [\, f(x_0, u_0), f(x_1, u_1), \ldots, f(x_K, u_K) \,] \in \mathbb{R}^{n\times(K+1)},$$

  otherwise, approximate time-derivatives by finite differences $\rightsquigarrow \dot{\hat{X}}$.
- Solve the linear least-squares problem (regression):

  $$(\hat{A}_*, \hat{H}_*, \hat{B}_*) := \arg\min_{(\hat{A},\hat{H},\hat{B})} \left\| \dot{\hat{X}} - \begin{bmatrix} \hat{A} & \hat{H} & \hat{B} \end{bmatrix} \begin{bmatrix} \hat{X} \\ \widehat{X^2} \\ U \end{bmatrix} \right\|_F + \beta \left\| \begin{bmatrix} \hat{A}\, \hat{H}\, \hat{B} \end{bmatrix} \right\|_q$$

  with potential regularization as before and $\widehat{X^2} := [x_0 \otimes x_0, \ldots, x_K \otimes x_K]$.

# Operator Inference
## Selected References (Chronological)


Peherstorfer, B., Willcox, K. (2016): Data-driven operator inference for nonintrusive projection-based model reduction. Comput. Methods Appl. Mech. Eng. 306:196–215.

Brunton, B.W., Johnson, L.A., Ojemann, J.G., Kutz, J.N. (2016): Extracting spatial-temporal coherent patterns in large-scale neural recordings using dynamic mode decomposition. J. Neurosci. Methods 258:1–15.

Annoni, J., Seiler, P. (2017): A method to construct reduced-order parameter-varying models. Int. J. Robust Nonlinear Control 27(4):582–597.

Qian, E., Kramer, B., Peherstorfer, B., Willcox, K. (2020): Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems. Physica D: Nonlinear Phenomena 406:132401.

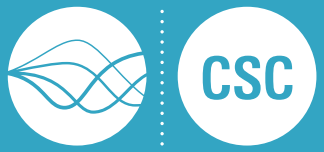
Benner, P., Goyal, P., Kramer, B., Peherstorfer, B., Willcox, K. (2020): Operator inference for non-intrusive model reduction of systems with non-polynomial nonlinear terms. Comp. Meth. Appl. Mech. Eng., 372:113433.

Yıldız, S., Goyal, P., Benner, P., Karasozen, B. (2021): Learning reduced-order dynamics for parametrized shallow water equations from data. Int. J. Numer. Meth. Eng., 93(8):2803–2821.

Benner, P., Goyal, P., Heiland, J., Pontes Duff, I. (2022): Operator inference and physics-informed learning of low-dimensional models for incompressible flows. Elec. Trans. Numer. Anal., 56:28–51.

# Conclusions — Part I

- DMD and operator inference (OpInf) are regression-based powerful methods to infer linear and certain nonlinear dynamical systems from data.
- Both look simple, but the devil is in the details.
- Choice of good observables? (Learning to learn?)
- Statistical aspects are not too well understood: impact of noise in the data on inferred models?
- Recent work combines OpInf with neural networks to solve nonlinear identification problems **($\rightsquigarrow$ Part II).**
- Error bounds for non-intrusive MOR not well developed yet, but theoretic results indicate that the OpInf model asymptotically (when increasing the number of snapshots) yields the POD model. Then, intrusive MOR error bounds can be applied.

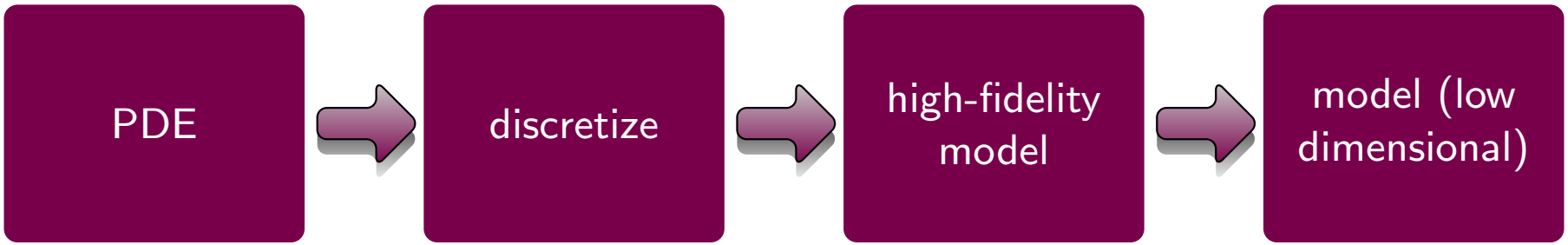


# Outline

# Part II: Enhancing Operator Inference by Deep Learning

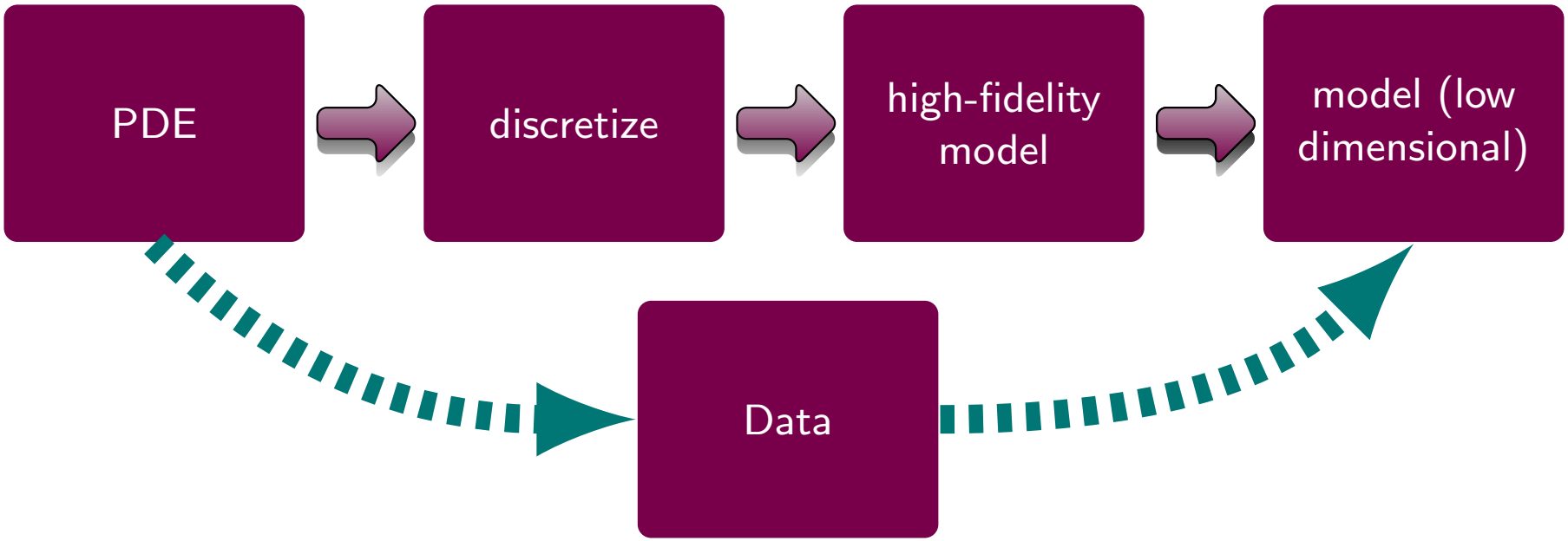
- Engineering processes are supported by domain knowledge and first principles

  $\rightsquigarrow$ a PDE model can be obtained that adequately explains the dynamics

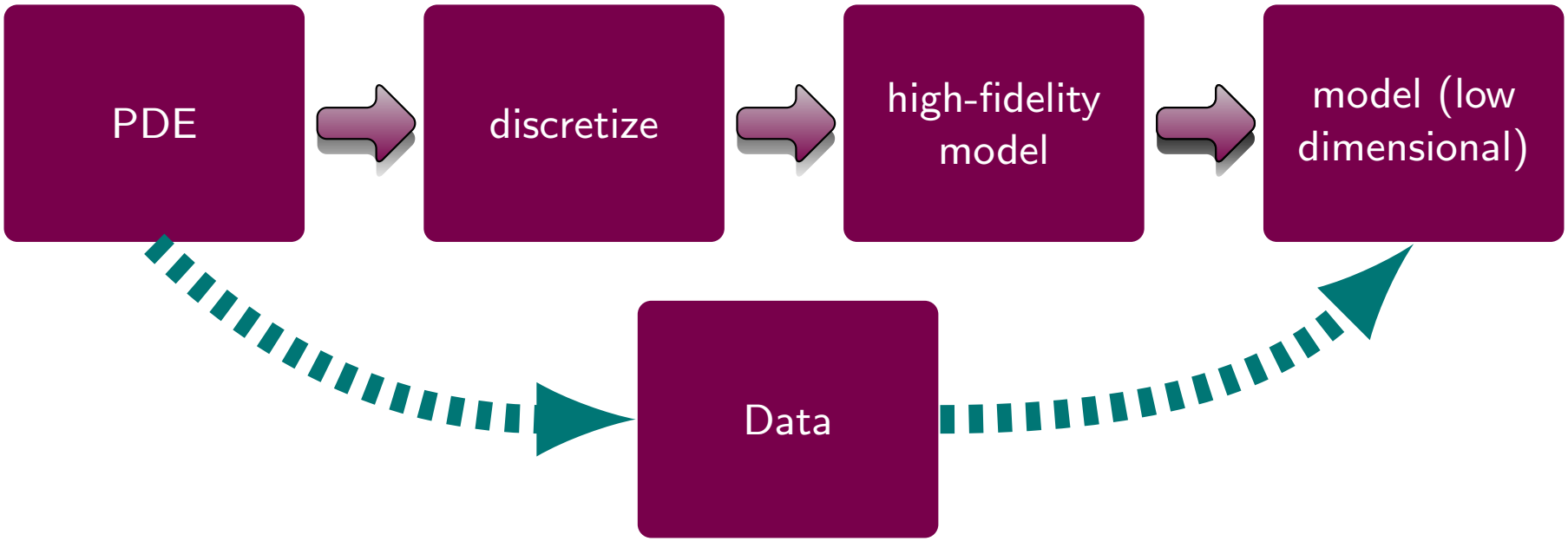


- Engineering processes are supported by domain knowledge and first principles

  $\rightsquigarrow$ a PDE model can be obtained that adequately explains the dynamics



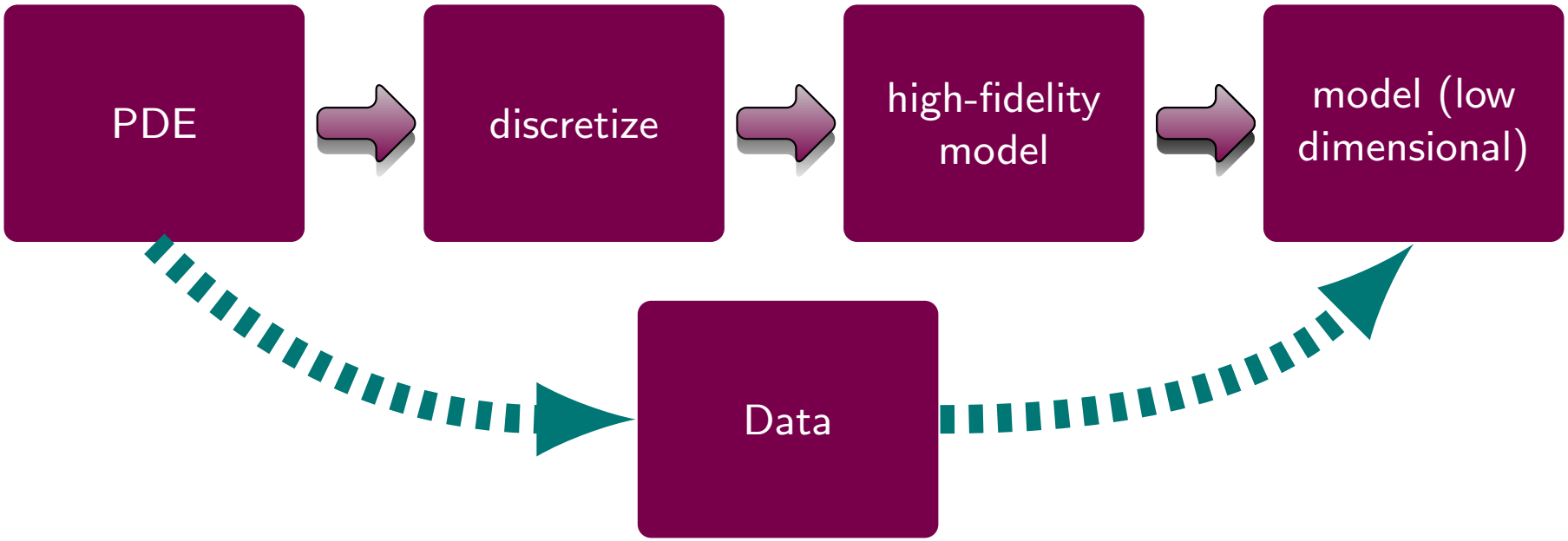
- Engineering processes are supported by domain knowledge and first principles
  $\rightsquigarrow$ a PDE model can be obtained that adequately explains the dynamics




- Data collection: obtained using a legacy code, or commercial software, or experiments.

# Part II: Enhancing Operator Inference by Deep Learning

- Engineering processes are supported by domain knowledge and first principles
  $\rightsquigarrow$ a PDE model can be obtained that adequately explains the dynamics




- Data collection: obtained using a legacy code, or commercial software, or experiments.
- Ideal goal: obtain the same reduced-order model (ROM) as obtained by intrusive model order reduction using data, so that error bounds and convergence analysis for ROMs can be directly employed!

# Operator Inference in Detail

## Non-intrusive approach

### Operator inference framework [Peherstorfer/Willcox '16]

- Operator inference leverages the known physical structure at the PDE level.
- Assume a quadratic high-fidelity model resulting from an underlying PDE $\frac{\partial x}{\partial t} = \mathcal{A}(x) + \mathcal{H}(x)$ with linear and quadratic terms:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{H}(\mathbf{x}(t) \otimes \mathbf{x}(t))$$

## Operator inference framework [Peherstorfer/Willcox '16]

- Operator inference leverages the known physical structure at the PDE level.
- Assume a quadratic high-fidelity model resulting from an underlying PDE $\frac{\partial x}{\partial t} = \mathcal{A}(x) + \mathcal{H}(x)$ with linear and quadratic terms:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{H}(\mathbf{x}(t) \otimes \mathbf{x}(t))$$

- Data preparation (in reduced dimension)
  1. Build temporal snapshot matrix $\mathbf{X} := \begin{bmatrix} | & | & & | \\ \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_k \\ | & | & & | \end{bmatrix}$.

# Operator Inference in Detail

Non-intrusive approach

## Operator inference framework [Peherstorfer/Willcox '16]

- Operator inference leverages the known physical structure at the PDE level.
- Assume a quadratic high-fidelity model resulting from an underlying PDE $\frac{\partial x}{\partial t} = \mathcal{A}(x) + \mathcal{H}(x)$ with linear and quadratic terms:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{H}(\mathbf{x}(t) \otimes \mathbf{x}(t))$$

- Data preparation (in reduced dimension)
  1. Build temporal snapshot matrix $\mathbf{X} := \begin{bmatrix} | & | & & | \\ \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_k \\ | & | & & | \end{bmatrix}$.
  2. Compute projection matrix $\mathbf{V}$ using dominant POD basis vectors.

## Operator inference framework [Peherstorfer/Willcox '16]

- Operator inference leverages the known physical structure at the PDE level.
- Assume a quadratic high-fidelity model resulting from an underlying PDE $\frac{\partial x}{\partial t} = \mathcal{A}(x) + \mathcal{H}(x)$ with linear and quadratic terms:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{H}(\mathbf{x}(t) \otimes \mathbf{x}(t))$$

- Data preparation (in reduced dimension)
  1. Build temporal snapshot matrix $\mathbf{X} := \begin{bmatrix} | & | & & | \\ \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_k \\ | & | & & | \end{bmatrix}$.
  2. Compute projection matrix $\mathbf{V}$ using dominant POD basis vectors.
  3. Reduced state vectors

$$\hat{\mathbf{X}} := V^T\mathbf{X} = \begin{bmatrix} | & | & & | \\ \hat{\mathbf{x}}_0 & \hat{\mathbf{x}}_1 & \cdots & \hat{\mathbf{x}}_k \\ | & | & & | \end{bmatrix}, \qquad \hat{\mathbf{X}}^{\otimes} := \begin{bmatrix} | & | & & | \\ \hat{\mathbf{x}}_0^{\otimes} & \hat{\mathbf{x}}_1^{\otimes} & \cdots & \hat{\mathbf{x}}_k^{\otimes} \\ | & | & & | \end{bmatrix}.$$

with $\hat{\mathbf{x}}_i = \mathbf{V}^\top \mathbf{x}_i$ and $\hat{\mathbf{x}}_i^{\otimes} = \hat{\mathbf{x}}_i \otimes \hat{\mathbf{x}}_i$

## Operator inference framework [Peherstorfer/Willcox '16]

- Operator inference leverages the known physical structure at the PDE level.
- Assume a quadratic high-fidelity model resulting from an underlying PDE $\frac{\partial x}{\partial t} = \mathcal{A}(x) + \mathcal{H}(x)$ with linear and quadratic terms:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{H}(\mathbf{x}(t) \otimes \mathbf{x}(t))$$

- Data preparation (in reduced dimension)
  1. Build temporal snapshot matrix $\mathbf{X} := \begin{bmatrix} | & | & & | \\ \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_k \\ | & | & & | \end{bmatrix}$.
  2. Compute projection matrix $\mathbf{V}$ using dominant POD basis vectors.
  3. Reduced state vectors

$$\hat{\mathbf{X}} := V^T\mathbf{X} = \begin{bmatrix} | & | & & | \\ \hat{\mathbf{x}}_0 & \hat{\mathbf{x}}_1 & \cdots & \hat{\mathbf{x}}_k \\ | & | & & | \end{bmatrix}, \qquad \hat{\mathbf{X}}^{\otimes} := \begin{bmatrix} | & | & & | \\ \hat{\mathbf{x}}_0^{\otimes} & \hat{\mathbf{x}}_1^{\otimes} & \cdots & \hat{\mathbf{x}}_k^{\otimes} \\ | & | & & | \end{bmatrix}.$$

  4. Approximate time-derivative data $\dot{\hat{\mathbf{X}}} := \begin{bmatrix} | & | & & | \\ \dot{\hat{\mathbf{x}}}_0 & \dot{\hat{\mathbf{x}}}_1 & \cdots & \dot{\hat{\mathbf{x}}}_k \\ | & | & & | \end{bmatrix}$.

## Operator inference framework [Peherstorfer/Willcox '16]

A ROM of the form

$$\dot{\hat{\mathbf{x}}}(t) = \hat{\mathbf{A}}\hat{\mathbf{x}}(t) + \hat{\mathbf{H}}(\hat{\mathbf{x}}(t) \otimes \hat{\mathbf{x}}(t))$$

can be obtained using projected data by solving the optimization problem

$$\min_{\hat{\mathbf{A}},\hat{\mathbf{H}}} \left\| \dot{\hat{\mathbf{X}}} - \hat{\mathbf{A}}\hat{\mathbf{X}} - \hat{\mathbf{H}}\hat{\mathbf{X}}^{\otimes} \right\|.$$

## Operator inference framework [Peherstorfer/Willcox '16]

A ROM of the form

$$\dot{\hat{\mathbf{x}}}(t) = \hat{\mathbf{A}}\hat{\mathbf{x}}(t) + \hat{\mathbf{H}}(\hat{\mathbf{x}}(t) \otimes \hat{\mathbf{x}}(t))$$

can be obtained using projected data by solving the optimization problem

$$\min_{\hat{\mathbf{A}},\hat{\mathbf{H}}} \left\| \dot{\hat{\mathbf{X}}} - \hat{\mathbf{A}}\hat{\mathbf{X}} - \hat{\mathbf{H}}\hat{\mathbf{X}}^{\otimes} \right\|.$$

Remarks:

- Notice that we do not require at any step the full-order discretized model.

## Operator inference framework [Peherstorfer/Willcox '16]

A ROM of the form

$$\dot{\hat{\mathbf{x}}}(t) = \hat{\mathbf{A}}\hat{\mathbf{x}}(t) + \hat{\mathbf{H}}(\hat{\mathbf{x}}(t) \otimes \hat{\mathbf{x}}(t))$$

can be obtained using projected data by solving the optimization problem

$$\min_{\hat{\mathbf{A}},\hat{\mathbf{H}}} \left\| \dot{\hat{\mathbf{X}}} - \hat{\mathbf{A}}\hat{\mathbf{X}} - \hat{\mathbf{H}}\hat{\mathbf{X}}^{\otimes} \right\|.$$

Remarks:

- Notice that we do not require at any step the full-order discretized model.
- Operator inference recovers intrusive POD reduced model if data are Markovian.

[Peherstorfer '20]

## Operator inference framework [Peherstorfer/Willcox '16]

A ROM of the form

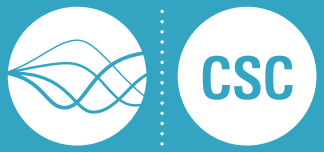
$$\dot{\hat{\mathbf{x}}}(t) = \hat{\mathbf{A}}\hat{\mathbf{x}}(t) + \hat{\mathbf{H}}(\hat{\mathbf{x}}(t) \otimes \hat{\mathbf{x}}(t))$$

can be obtained using projected data by solving the optimization problem

$$\min_{\hat{\mathbf{A}},\hat{\mathbf{H}}} \left\| \dot{\hat{\mathbf{X}}} - \hat{\mathbf{A}}\hat{\mathbf{X}} - \hat{\mathbf{H}}\hat{\mathbf{X}}^{\otimes} \right\| + \mathcal{R}(\hat{\mathbf{A}}, \hat{\mathbf{H}}).$$

Remarks:

- Notice that we do not require at any step the full-order discretized model.
- Operator inference recovers intrusive POD reduced model if data are Markovian.
  [Peherstorfer '20]
- Typically, the least-squares problem is ill-conditioned, hence need regularization.
  [McQuarrie et al. '21, B./Goyal/Heiland/Pontes '21]



# Outline

# Operator Inference for General Nonlinear Systems

## Nonlinear systems [B./Goyal/Kramer/Peherstorfer/Willcox '20]

- Consider a nonlinear system of the form

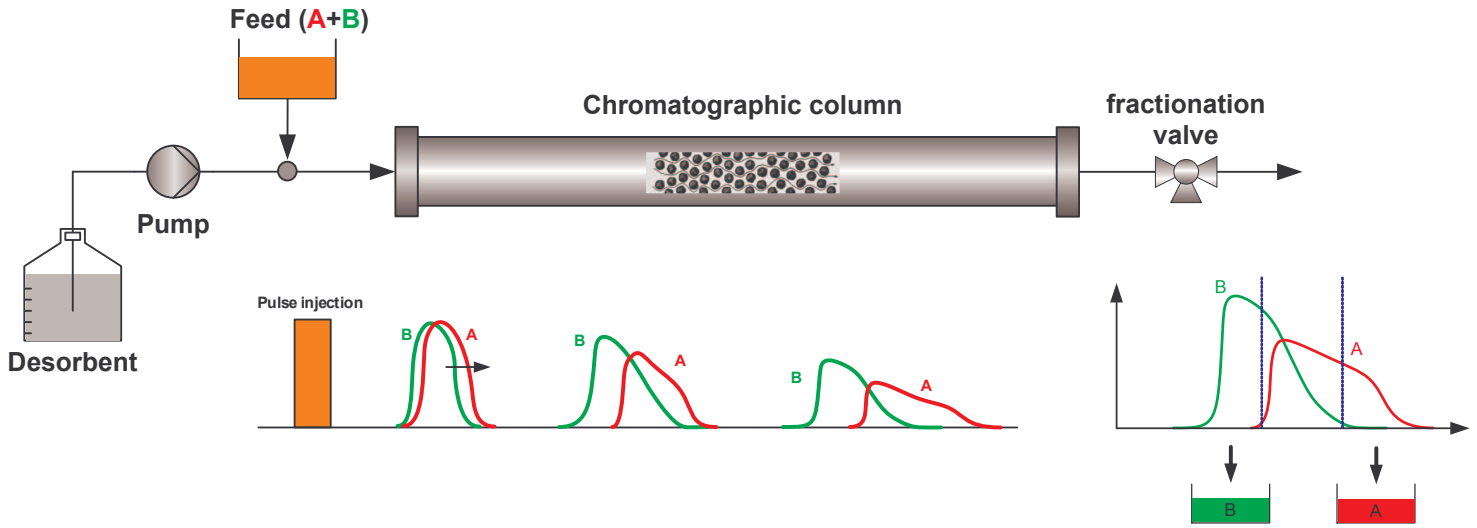
$$\frac{\partial s}{\partial t} = \mathcal{A}(s) + \mathcal{H}(s) + \mathcal{F}(t, s),$$

  where the analytic form of $\mathcal{F}(t, s)$ is known.
- We can learn a ROM of the form

$$\dot{\hat{\mathbf{s}}}(t) = \hat{\mathbf{A}}\hat{\mathbf{s}} + \hat{\mathbf{H}}\left(\hat{\mathbf{s}} \otimes \hat{\mathbf{s}}\right) + \hat{\mathbf{f}}(t, \hat{\mathbf{s}})$$

  directly from data!
- Simulation of reduced nonlinear system can be further accelerated using hyper-reduction.

# Batch Chromatography: A Chemical Separation Process




- The dynamics of a batch chromatography column can be described by the coupled PDE system of advection-diffusion type:

$$\frac{\partial c_i}{\partial t} + \frac{1-\epsilon}{\epsilon}\frac{\partial q_i}{\partial t} + \frac{\partial c_i}{\partial x} - \frac{1}{\mathrm{Pe}}\frac{\partial^2 c_i}{\partial x^2} = 0,$$
$$\frac{\partial q_i}{\partial t} = \kappa_i \left( q_i^{Eq} - q_i \right).$$

- It is a coupled PDE; thus, the coupling structure is desired to be preserved in learned ROM
- This is achieved by block diagonal projection, thereby not mixing separate physical quantities.

# Batch Chromatography: A Chemical Separation Process

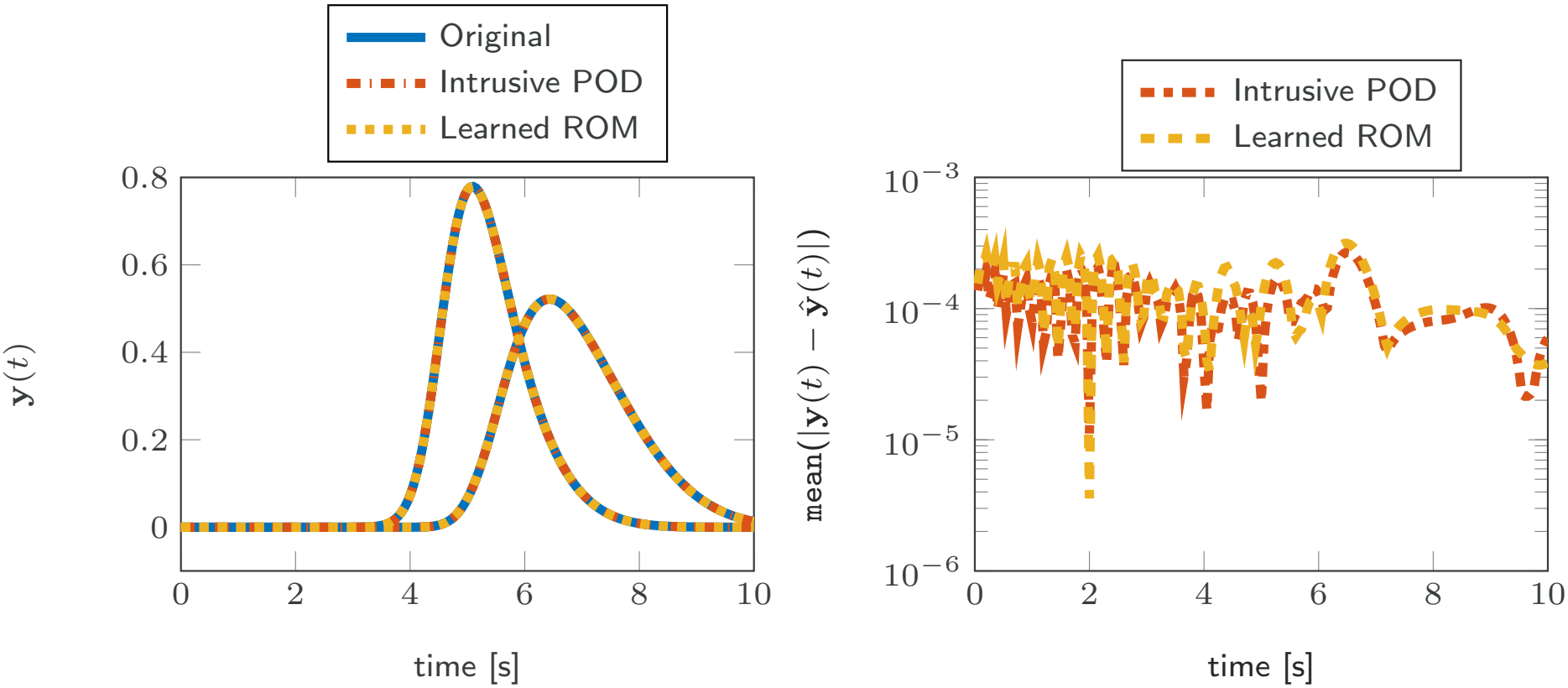



Figure: Batch chromatography example: A comparison of the POD intrusive model with the learned model of order $r = 4 \times 22$, where $n = 1600$ and $\mathrm{Pe} = 2000$.

# Parameterized Shallow Water Equations

- Parameterized shallow water equations are given by [Yildiz et al '21]

$$\begin{aligned}
\frac{\partial}{\partial t}\tilde{u} &= -h_x + \sin\theta\ \tilde{v} - \tilde{u}\tilde{u}_x - \tilde{v}\tilde{u}_y + \delta\cos\theta(h\tilde{u})_x - \frac{3}{8}\left(\delta\cos\theta\right)^2(h^2)_x,\\
\frac{\partial}{\partial t}\tilde{v} &= -h_y + \sin\theta\ \tilde{u} + \frac{1}{2}\delta\sin\theta\cos\theta\ h - \tilde{u}\tilde{v}_x - \tilde{v}\tilde{v}_y\\
&\quad + \delta\cos\theta\left((h\tilde{u})_y + \frac{1}{2}h\left(\tilde{v}_x - \tilde{u}_y\right)\right) - \frac{3}{8}\left(\delta\cos\theta\right)^2(h^2)_y,\\
\frac{\partial}{\partial t}h &= -(h\tilde{u})_x - (h\tilde{v})_y + \frac{1}{2}\delta\cos\theta(h^2)_x.
\end{aligned}$$

- Parameterized by the latitude $\theta$.
- $\tilde{\mathbf{u}} =: (\tilde{u}; \tilde{v})$ is the canonical velocity.
- $h$ is the height field.
- We collect the training data for 5 different parameter realizations $\theta$ in $\left[\frac{\pi}{6}, \frac{\pi}{3}\right]$.
- Infer a reduced parametric model directly from data of order $r = 75$.

# Parameterized Shallow Water Equations

- Parameterized shallow water equations are given by [Yildiz et al '21]

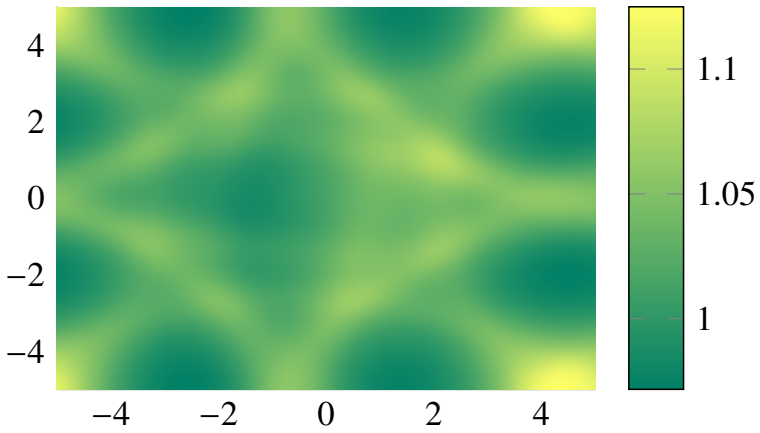
$$\frac{\partial}{\partial t}\tilde{u} = -h_x + \sin\theta\ \tilde{v} - \tilde{u}\tilde{u}_x - \tilde{v}\tilde{u}_y + \delta\cos\theta(h\tilde{u})_x - \frac{3}{8}\left(\delta\cos\theta\right)^2\left(h^2\right)_x,$$

$$\frac{\partial}{\partial t}\tilde{v} = -h_y + \sin\theta\ \tilde{u} + \frac{1}{2}\delta\sin\theta\cos\theta\ h - \tilde{u}\tilde{v}_x - \tilde{v}\tilde{v}_y$$

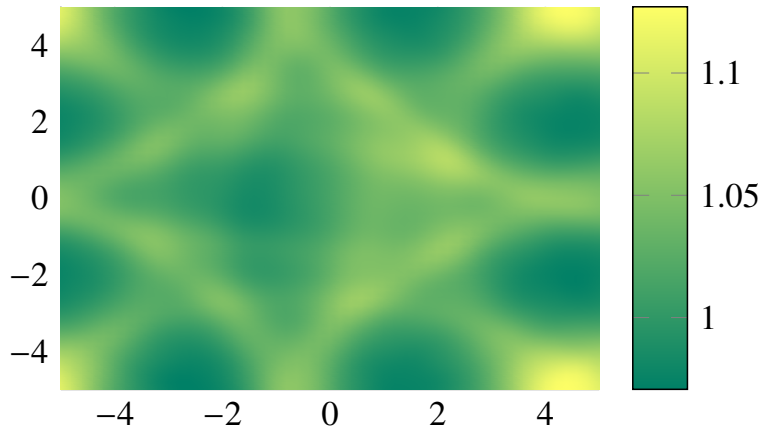
$$+ \delta\cos\theta\left((h\tilde{u})_y + \frac{1}{2}h\left(\tilde{v}_x - \tilde{u}_y\right)\right) - \frac{3}{8}\left(\delta\cos\theta\right)^2\left(h^2\right)_y,$$

$$\frac{\partial}{\partial t}h = -(h\tilde{u})_x - (h\tilde{v})_y + \frac{1}{2}\delta\cos\theta(h^2)_x.$$

- Comparison of the height field for the parameter $\theta = \frac{5\pi}{24}$:



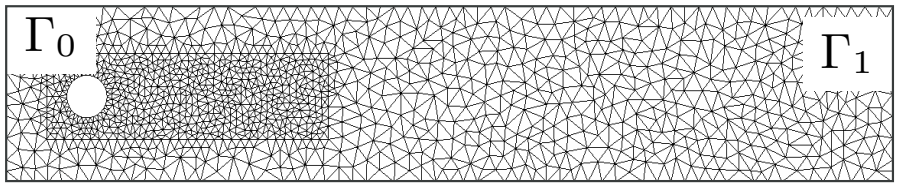

(a) FOM

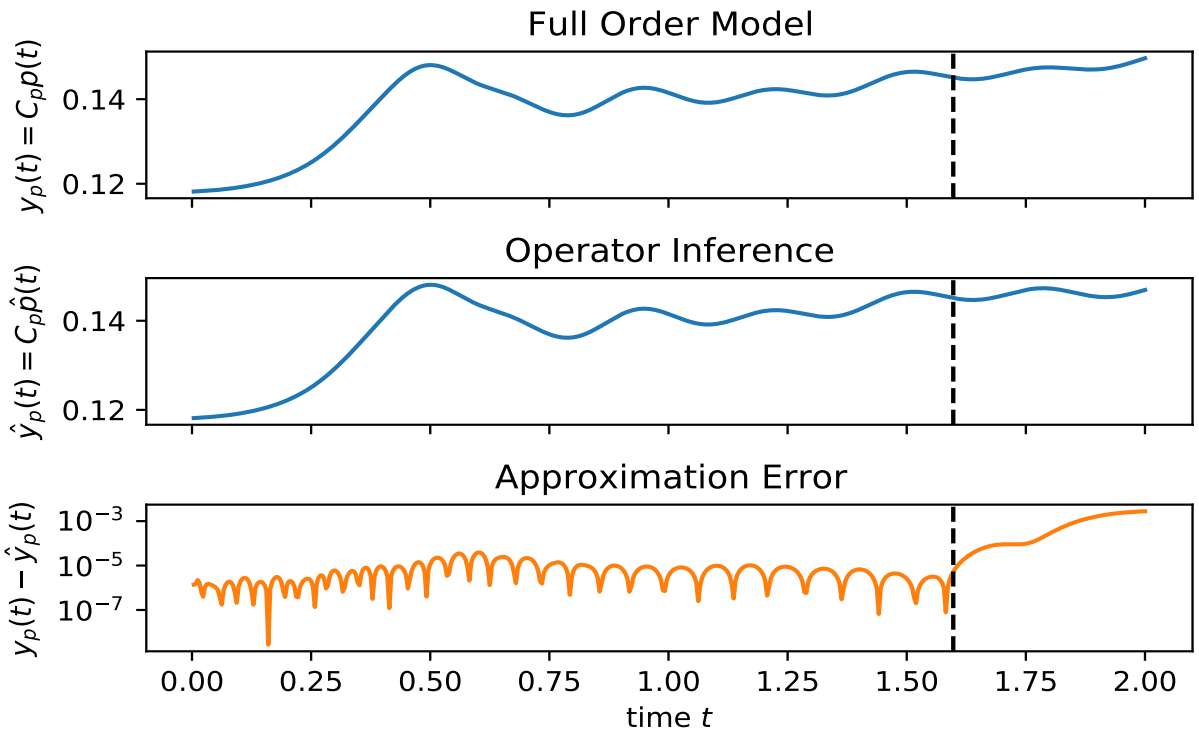



(b) Learned parametric model

# Operator Inference for Structured DAE Systems

Tailored operator inference for incompressible Navier-Stokes equations, by heeding incompressibility condition. [B./Goyal/Heiland/Pontes '22]



# Combining Operator Inference with Deep Learning

# Operator Inference for General Nonlinear Systems

## Problem formulation

$$\dot{\mathbf{v}}(t) = \mathbf{f}(\mathbf{v}(t)) + \mathbf{r}(\mathbf{v}(t))$$

- $\mathbf{f}(\mathbf{v}(t))$: known from physical laws or expert knowledge;
  - e.g., for chemical reaction models, we expect to have an Arrhenius-type term.
- $\mathbf{r}(\mathbf{v}(t))$: unknown terms
  - e.g., friction terms in robotics or vibration systems, effects of removed higher-frequency dynamics on the low-frequency response, etc.

# Operator Inference for General Nonlinear Systems

## Problem formulation

$$\dot{\mathbf{v}}(t) = \mathbf{f}(\mathbf{v}(t)) + \mathbf{r}(\mathbf{v}(t))$$

- $\mathbf{f}(\mathbf{v}(t))$: known from physical laws or expert knowledge;
  - e.g., for chemical reaction models, we expect to have an Arrhenius-type term.
- $\mathbf{r}(\mathbf{v}(t))$: unknown terms
  - e.g., friction terms in robotics or vibration systems, effects of removed higher-frequency dynamics on the low-frequency response, etc.

## Observation

- Often, governing equations are **quadratic**, i.e., $\mathbf{f}(\mathbf{v}) := \mathbf{A}\mathbf{v} + \mathbf{H}(\mathbf{v} \otimes \mathbf{v})$.
- If no additional information is given, we assume $\mathbf{f}$ to be quadratic.
- Moreover, possible to find artificial variables in which dynamics are quadratic.

  Philosophy: Lift & learn [Qian et al. '20]

Navier-Stokes equations

$$\rho\left(\frac{\partial u_r}{\partial t} + u_r\frac{\partial u_r}{\partial r} + u_z\frac{\partial u_r}{\partial z}\right) = -\frac{\partial p}{\partial r} + \mu\left(\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u_r}{\partial r}\right) + \frac{\partial^2 u_r}{\partial z^2} - \frac{u_r}{r^2}\right) + \rho g_r$$
$$\rho\left(\frac{\partial u_z}{\partial t} + u_r\frac{\partial u_z}{\partial r} + u_z\frac{\partial u_z}{\partial z}\right) = -\frac{\partial p}{\partial z} + \mu\left(\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u_z}{\partial r}\right) + \frac{\partial^2 u_z}{\partial z^2}\right) + \rho g_z$$
$$\frac{1}{r}\frac{\partial}{\partial r}(ru_r) + \frac{\partial u_z}{\partial z} = 0.$$

Fisher's equation

$$u_t = u(1-u) + u_{xx}$$

# Operator Inference for General Nonlinear Systems

## Problem formulation

$$\dot{\mathbf{v}}(t) = \mathbf{f}(\mathbf{v}(t)) + \mathbf{r}(\mathbf{v}(t))$$

- $\mathbf{f}(\mathbf{v}(t))$: known from physical laws or expert knowledge;
  - e.g., for chemical reaction models, we expect to have an Arrhenius-type term.
- $\mathbf{r}(\mathbf{v}(t))$: unknown terms
  - e.g., friction terms in robotics or vibration systems, effects of removed higher-frequency dynamics on the low-frequency response, etc.

## Lifting

[Gu '09/'11, B./Breiten '12/'15, Qian et al '20]

Consider the nonlinear system:

$$\dot{\mathbf{x}} = -\mathbf{x} + e^{-\mathbf{x}}$$

# Operator Inference for General Nonlinear Systems

## Problem formulation

$$\dot{\mathbf{v}}(t) = \mathbf{f}(\mathbf{v}(t)) + \mathbf{r}(\mathbf{v}(t))$$

- $\mathbf{f}(\mathbf{v}(t))$: known from physical laws or expert knowledge;
  - e.g., for chemical reaction models, we expect to have an Arrhenius-type term.
- $\mathbf{r}(\mathbf{v}(t))$: unknown terms
  - e.g., friction terms in robotics or vibration systems, effects of removed higher-frequency dynamics on the low-frequency response, etc.

## Lifting

[Gu '09/'11, B./Breiten '12/'15, Qian et al '20]

Consider the nonlinear system:

$$\dot{\mathbf{x}} = -\mathbf{x} + e^{-\mathbf{x}}$$

- Define $\mathbf{z}(t) = e^{-\mathbf{x}} \rightsquigarrow \dot{\mathbf{z}}(t) = -e^{-\mathbf{x}}\dot{\mathbf{x}} = -\mathbf{z}(t)\left(-\mathbf{x}(t) + \mathbf{z}(t)\right)$

# Operator Inference for General Nonlinear Systems

## Problem formulation

$$\dot{\mathbf{v}}(t) = \mathbf{f}(\mathbf{v}(t)) + \mathbf{r}(\mathbf{v}(t))$$

- $\mathbf{f}(\mathbf{v}(t))$: known from physical laws or expert knowledge;
  - e.g., for chemical reaction models, we expect to have an Arrhenius-type term.
- $\mathbf{r}(\mathbf{v}(t))$: unknown terms
  - e.g., friction terms in robotics or vibration systems, effects of removed higher-frequency dynamics on the low-frequency response, etc.

## Lifting

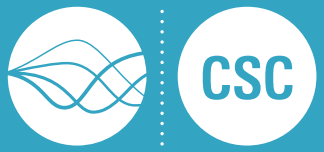
[Gu '09/'11, B./Breiten '12/'15, Qian et al '20]

Consider the nonlinear system:

$$\dot{\mathbf{x}} = -\mathbf{x} + e^{-\mathbf{x}}$$

- Define $\mathbf{z}(t) = e^{-\mathbf{x}} \rightsquigarrow \dot{\mathbf{z}}(t) = -e^{-\mathbf{x}}\dot{\mathbf{x}} = -\mathbf{z}(t)\left(-\mathbf{x}(t) + \mathbf{z}(t)\right)$
- The system becomes linear-quadratic in $(\mathbf{x}(t), \mathbf{z}(t))$, i.e.,

$$\begin{bmatrix}\dot{\mathbf{x}}(t)\\ \dot{\mathbf{z}}(t)\end{bmatrix} = \begin{bmatrix}-\mathbf{x}(t) + z(t)\\ 0\end{bmatrix} + \begin{bmatrix}0\\ \mathbf{z}(t)\left(\mathbf{x}(t) - \mathbf{z}(t)\right)\end{bmatrix}.$$



# Outline

# Linear-Quadratic Residual Networks

For simplicity, consider the form:

$$\dot{\mathbf{v}}(t) = f(\mathbf{v}(t)) = \mathbf{A}\mathbf{v}(t) + \mathbf{H}\left(\mathbf{v}(t) \otimes \mathbf{v}(t)\right) + \mathbf{r}(\mathbf{v}(t)),$$

where

- $\mathbf{r}(\mathbf{v}(t))$ can be interpreted as a residual that cannot be resolved by the quadratic-form or prior knowledge.

# Linear-Quadratic Residual Networks

For simplicity, consider the form:

$$\dot{\mathbf{v}}(t) = f(\mathbf{v}(t)) = \mathbf{A}\mathbf{v}(t) + \mathbf{H}\left(\mathbf{v}(t) \otimes \mathbf{v}(t)\right) + \mathbf{r}(\mathbf{v}(t)),$$

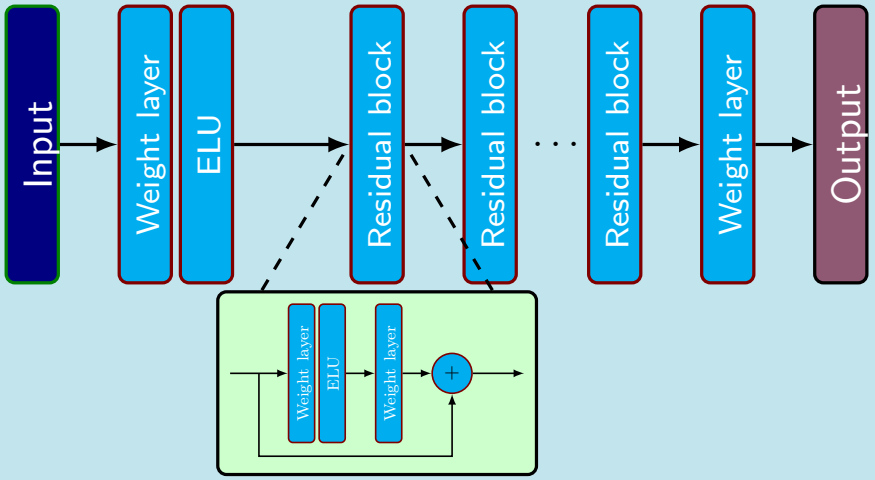
where

- $\mathbf{r}(\mathbf{v}(t))$ can be interpreted as a residual that cannot be resolved by the quadratic-form or prior knowledge.

## Residual networks

[He/Ren/Sun '16]

- Have shown their power in computer vision applications.



# Linear-Quadratic Residual Networks

For simplicity, consider the form:

$$\dot{\mathbf{v}}(t) = f(\mathbf{v}(t)) = \mathbf{A}\mathbf{v}(t) + \mathbf{H}\left(\mathbf{v}(t) \otimes \mathbf{v}(t)\right) + \mathbf{r}(\mathbf{v}(t)),$$

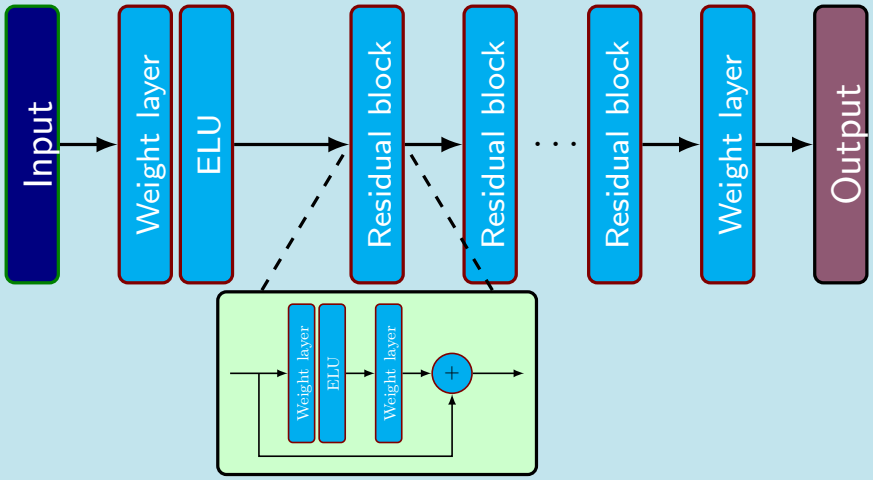
where

- $\mathbf{r}(\mathbf{v}(t))$ can be interpreted as a residual that cannot be resolved by the quadratic-form or prior knowledge.

## Residual networks

[He/Ren/Sun '16]

- Have shown their power in computer vision applications.
- There is an established connection to dynamical systems.



# Linear-Quadratic Residual Networks

For simplicity, consider the form:

$$\dot{\mathbf{v}}(t) = f(\mathbf{v}(t)) = \mathbf{A}\mathbf{v}(t) + \mathbf{H}\left(\mathbf{v}(t) \otimes \mathbf{v}(t)\right) + \mathbf{r}(\mathbf{v}(t)),$$

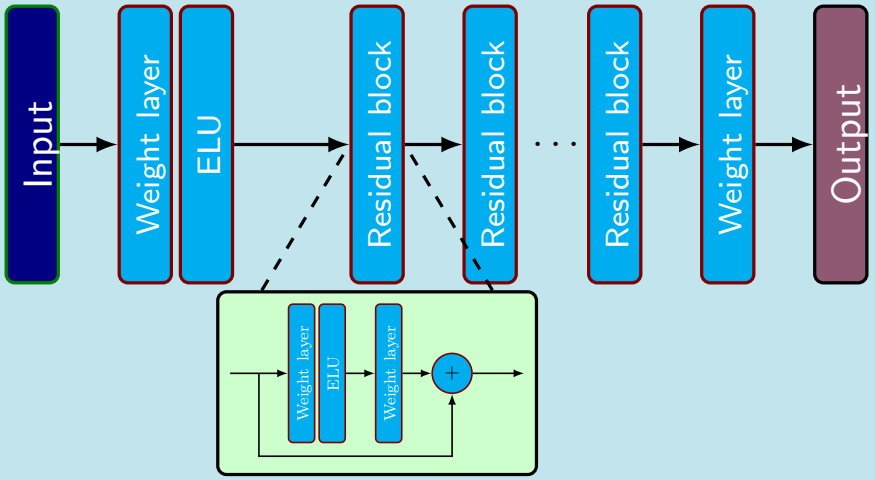
where

- $\mathbf{r}(\mathbf{v}(t))$ can be interpreted as a residual that cannot be resolved by the quadratic-form or prior knowledge.

## Residual networks

[He/Ren/Sun '16]

- Have shown their power in computer vision applications.
- There is an established connection to dynamical systems.
- Residual type connections hint to adaptive refinement of solution or features.



# Linear-Quadratic Residual Networks

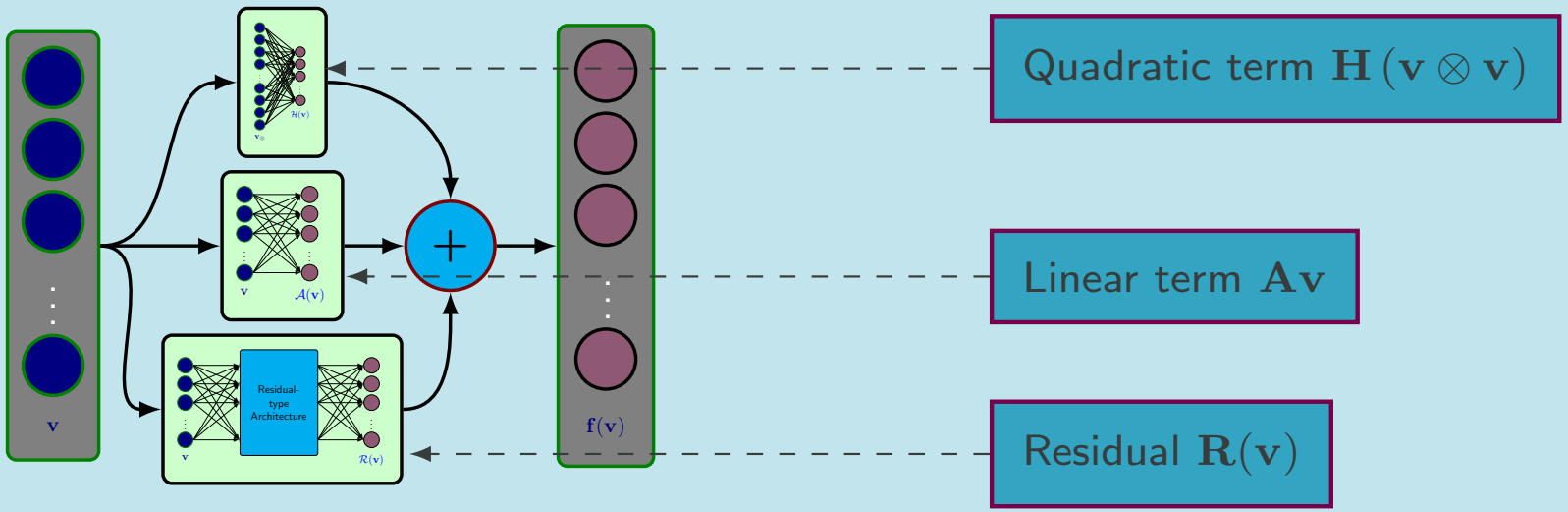
For simplicity, consider the form:

$$\dot{\mathbf{v}}(t) = f(\mathbf{v}(t)) = \mathbf{A}\mathbf{v}(t) + \mathbf{H}\left(\mathbf{v}(t) \otimes \mathbf{v}(t)\right) + \mathbf{r}(\mathbf{v}(t)),$$

where

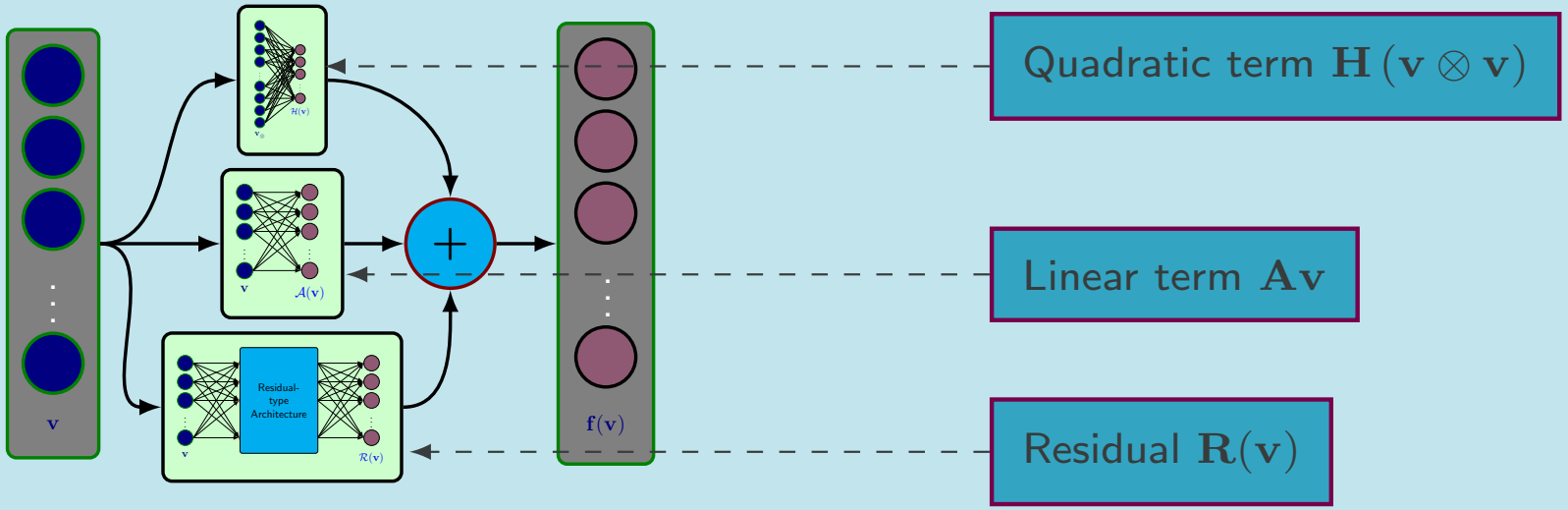
- $\mathbf{r}(\mathbf{v}(t))$ can be interpreted as a residual that cannot be resolved by the quadratic-form or prior knowledge.

## Linear-Quadratic Residual Networks (LQResNet) [Goyal/B. '21]



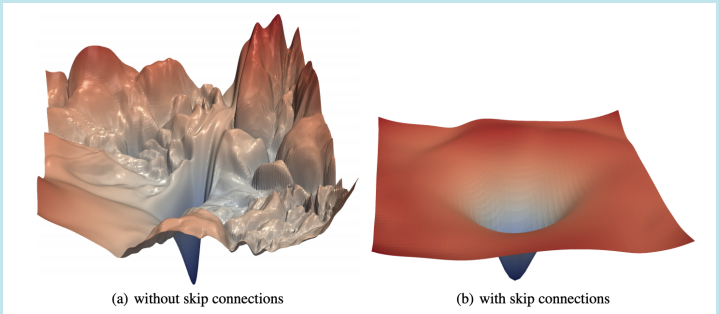
# Advantages of the Architecture

## Linear-Quadratic Residual Networks (LQResNet) [Goyal/B. '21]



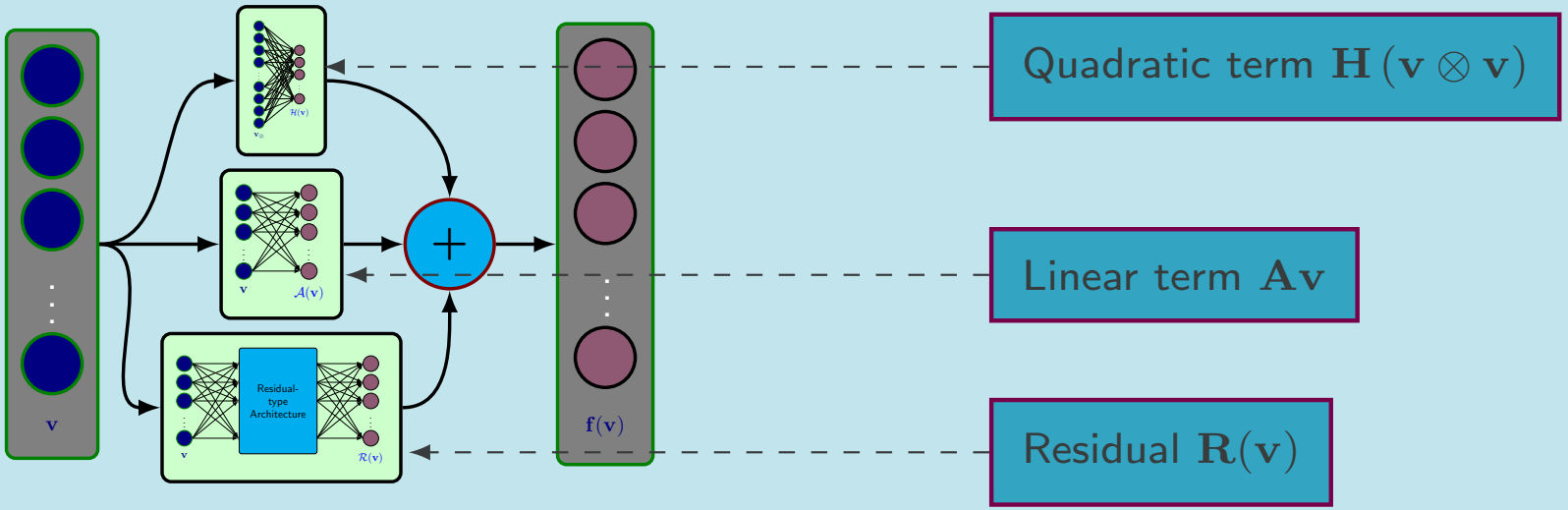

## Remarks

- Due to skip connections, loss landscape becomes less bumpy. [Li et al. '18]



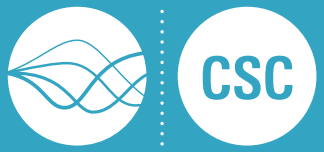
# Advantages of the Architecture

## Linear-Quadratic Residual Networks (LQResNet) [Goyal/B. '21]




## Remarks

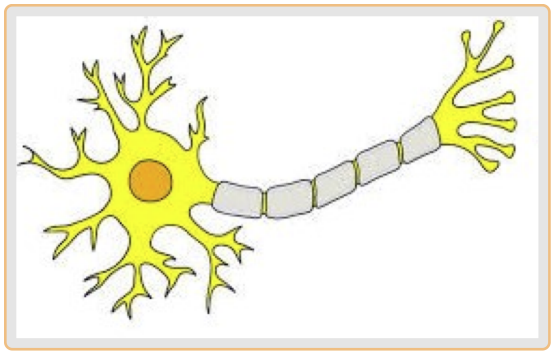
- Due to skip connections, loss landscape becomes less bumpy. [Li et al. '18]
- Layers can be added without restarting whole optimization as deep residual layers tend to refine the mapping.



# Outline

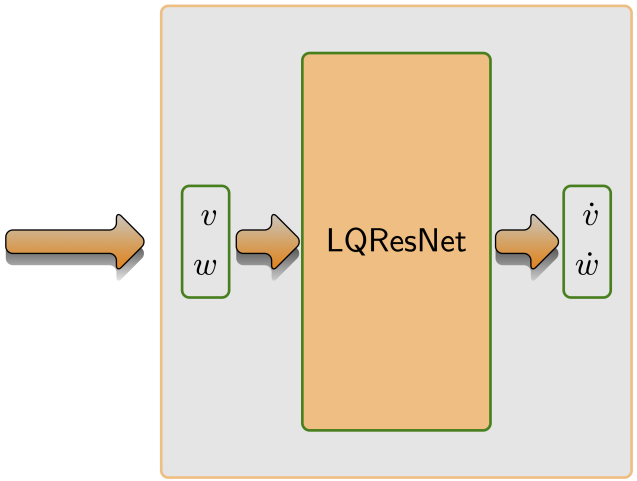
## Set-up

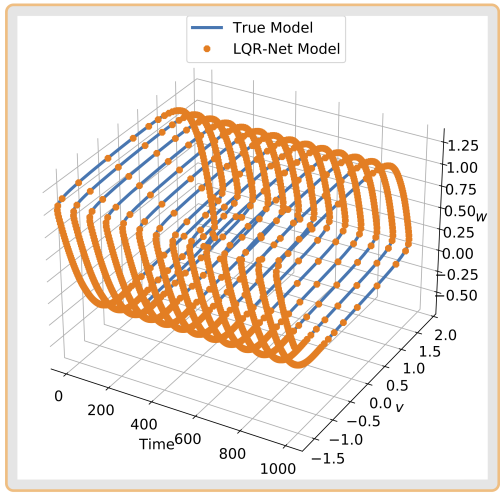
- The FitzHugh-Nagumo model is a coupled PDE-ODE model describing the spiking of a neuron.
- Assume to have time-series data for $10$ different initial conditions.
- We build different networks for both variables.
- We check the predictive capabilities of the inferred model under new initial condition.



Governing equations

$$\dot{v}(t) = v - \frac{v^3}{3} - w + RI_{ext}$$
$$\dot{w}(t) = v + a - bw$$



## Set-up

- Represents complex wide-range dynamical behavior in yeast glycolysis.

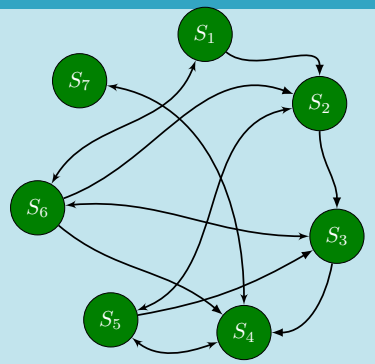



Figure: Interaction topology for 7 species.

## Set-up

- Represents complex wide-range dynamical behavior in yeast glycolysis.
- There are 7 involved species.

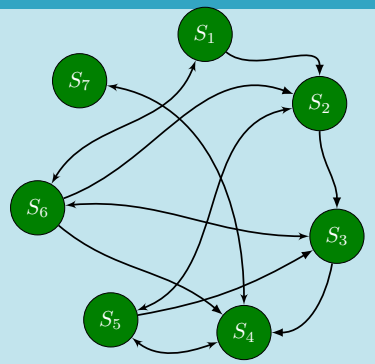



Figure: Interaction topology for 7 species.

## Set-up

- Represents complex wide-range dynamical behavior in yeast glycolysis.
- There are 7 involved species.
- Data for 30 different initial conditions.

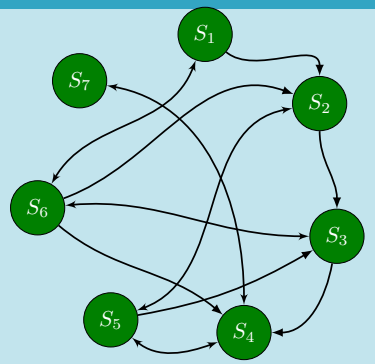



Figure: Interaction topology for 7 species.

## Set-up

- Represents complex wide-range dynamical behavior in yeast glycolysis.
- There are 7 involved species.
- Data for 30 different initial conditions.
- Utilized interaction topology in learning.

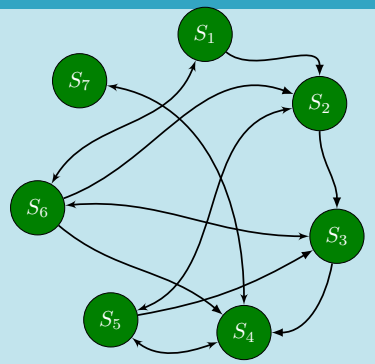



Figure: Interaction topology for 7 species.

## Set-up

- Represents complex wide-range dynamical behavior in yeast glycolysis.
- There are 7 involved species.
- Data for 30 different initial conditions.
- Utilized interaction topology in learning.
- Check the predictive capabilities under new condition.

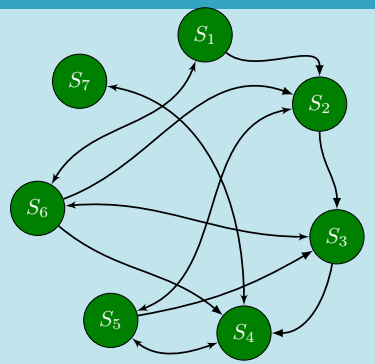
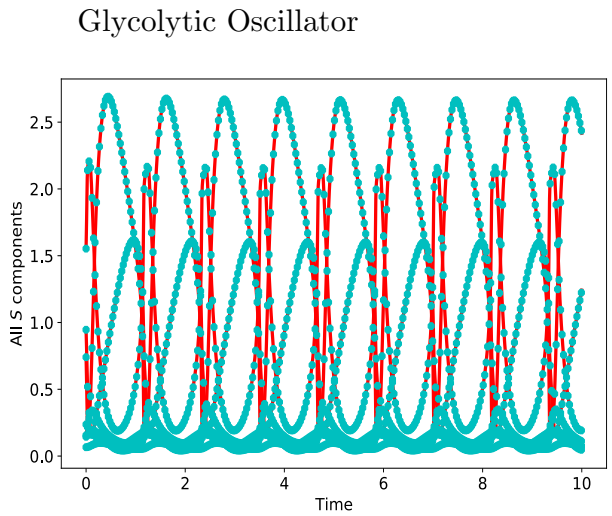
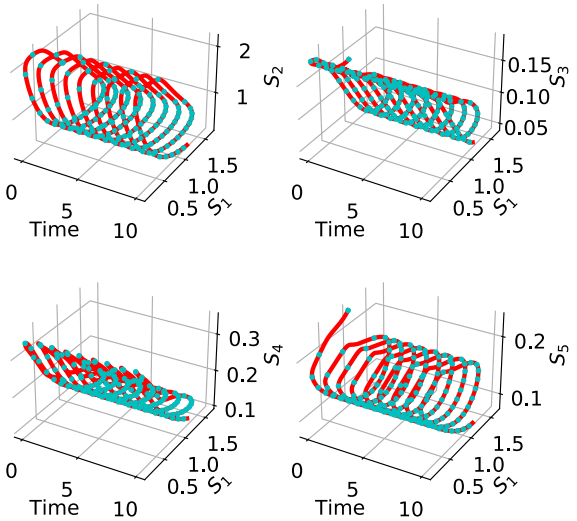



Figure: Interaction topology for 7 species.



# Tubular Reactor Model

- One dimensional model with a single reaction, describing dynamics of the species concentration $\psi(x,t)$ and temperature $\theta(x,t)$ via

$$\begin{aligned}\frac{\partial \psi}{\partial t} &= \frac{1}{\mathrm{Pe}}\frac{\partial^2 \psi}{\partial x^2} - \frac{\partial \psi}{\partial x} - \mathcal{D}\mathcal{F}(\psi,\theta;\gamma),\\ \frac{\partial \theta}{\partial t} &= \frac{1}{\mathrm{Pe}}\frac{\partial^2 \theta}{\partial x^2} - \frac{\partial \theta}{\partial x} - \beta(\theta-\theta_{\mathsf{ref}}) + \mathcal{B}\mathcal{D}\mathcal{F}(\psi,\theta;\gamma),\end{aligned}$$

  with spatial variable $x \in (0,1)$, time $t > 0$ and Arrhenius reaction term

$$\mathcal{F}(\psi,\theta;\gamma) = \psi \exp\left(\gamma - \frac{\gamma}{\theta}\right).$$

- The quantity of interest is the temperature oscillation at the reactor exit:

$$\mathbf{y}(t) = \theta(\mathbf{x}=1,t).$$

# Tubular Reactor Model

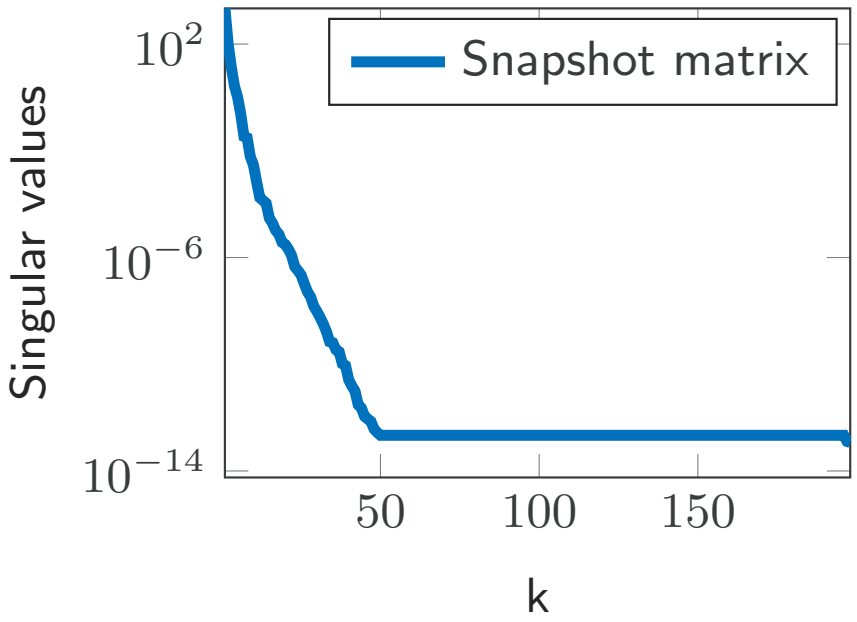



Figure: Decay of singular values of the snapshots.

- Rapid decay of singular values of training data $\rightsquigarrow$ possibility of lower order models.
- The dominant three POD modes capture more than $99.8\%$ of the energy, yet the POD model is unstable.

# Tubular Reactor Model

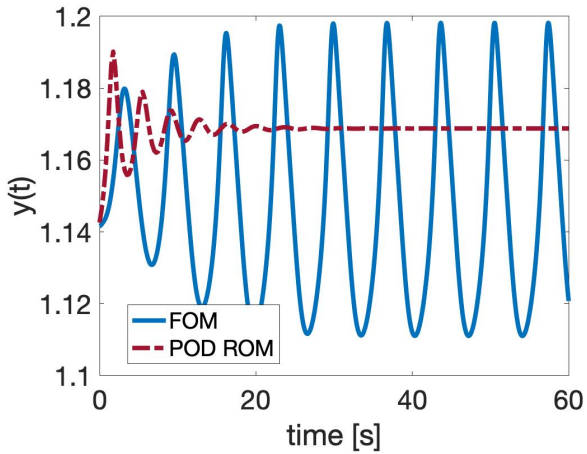



Figure: A comparison of the temperature oscillations at exit.

- Rapid decay of singular values of training data $\rightsquigarrow$ possibility of lower order models.
- The dominant three POD modes capture more than $99.8\%$ of the energy, yet the POD model is unstable.
- We employ the LQResNet approach to learn the correction.

# Tubular Reactor Model

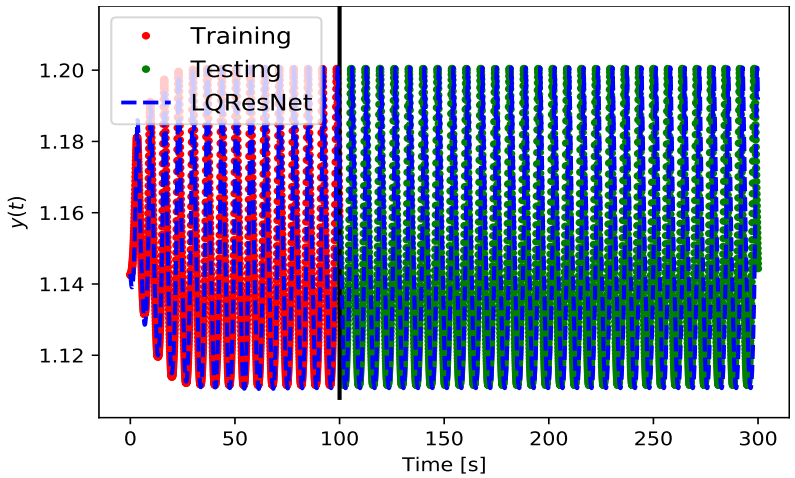



Figure: A comparison of the temperature oscillations at exit.

- Rapid decay of singular values of training data $\rightsquigarrow$ possibility of lower order models.
- The dominant three POD modes capture more than $99.8\%$ of the energy, yet the POD model is unstable.
- We employ the LQResNet approach to learn the correction.

# Tubular Reactor Model

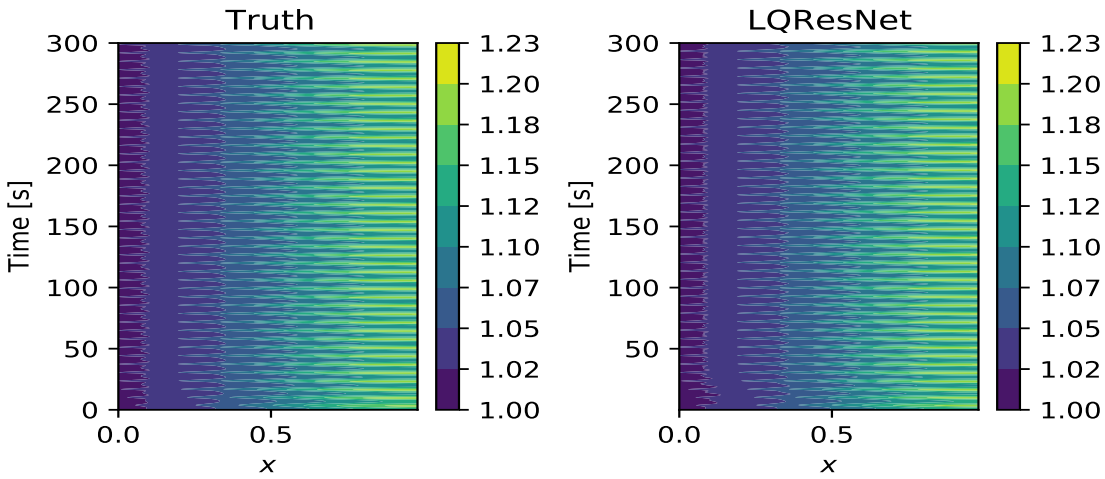



Figure: A comparison of the temperature oscillations in the whole domain.

- Rapid decay of singular values of training data $\rightsquigarrow$ possibility of lower order models.
- The dominant three POD modes capture more than $99.8\%$ of the energy, yet the POD model is unstable.
- We employ the LQResNet approach to learn the correction.

# Outlook

## Contribution

- We have studied an approach to learn a mathematical model to describe nonlinear dynamics.
- Basis: operator inference and its extensions, utilizing prior PDE knowledge.
- New: model residual identified using architecture LQResNet, inspired by residual network.
- The design allows us to incorporate prior hypotheses about the process.

# Outlook

## Contribution

- We have studied an approach to learn a mathematical model to describe nonlinear dynamics.
- Basis: operator inference and its extensions, utilizing prior PDE knowledge.
- New: model residual identified using architecture LQResNet, inspired by residual network.
- The design allows us to incorporate prior hypotheses about the process.

## On-going work

- Very often, we can build a dictionary of good candidate basis functions, but probably do not want all of them in the dictionary. Therefore, we seek a parsimonious model
  - to pick few entries from the dictionary and learn residual by deep learning.
- Appropriate treatment of noise . . . [Rudy/Kutz/Brunton '19]
- Missing/corrupted data in time series.
- Working with several applications in material science and chemical engineering.

# Outlook

## Contribution

- We have studied an approach to learn a mathematical model to describe nonlinear dynamics.
- Basis: operator inference and its extensions, utilizing prior PDE knowledge.
- New: model residual identified using architecture LQResNet, inspired by residual network.
- The design allows us to incorporate prior hypotheses about the process.

## On-going work

- Very often, we can build a dictionary of good candidate basis functions, but probably do not want all of them in the dictionary. Therefore, we seek a parsimonious model
  - to pick few entries from the dictionary and learn residual by deep learning.
- Appropriate treatment of noise . . . [Rudy/Kutz/Brunton '19]
- Missing/corrupted data in time series.
- Working with several applications in material science and chemical engineering.

**Thank you for your attention!!**

# Selected References, Part II


Benner, P., Goyal, P., Kramer, B., Peherstorfer, B., and Willcox, K. (2020).
Operator inference for non-intrusive model reduction of systems with non-polynomial nonlinear terms.
*Comp. Meth. Appl. Mech. Eng.*, 372:113433.

Brunton, S. L., Proctor, J. L., and Kutz, J. N. (2016).
Discovering governing equations from data by sparse identification of nonlinear dynamical systems.
*Proc. Nat. Acad. Sci. U.S.A.*, 113(15):3932–3937.

Goyal, P. and Benner, P. (2021).
LQResNet: A deep neural network architecture for learning dynamic processes.
*arXiv preprint arXiv:2103.02249*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016).
Deep residual learning for image recognition.
In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, pages 770–778.

Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. (2018).
Visualizing the loss landscape of neural nets.
In *Neural Inform. Processing Syst.*

Peherstorfer, B. and Willcox, K. (2016).
Data-driven operator inference for nonintrusive projection-based model reduction.
*Comp. Meth. Appl. Mech. Eng.*, 306:196–215.

Qian, E., Kramer, B., Peherstorfer, B., and Willcox, K. (2020).
Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems.
*Physica D: Nonlinear Phenomena*, 406:132401.

Rudy, S. H., Kutz, J. N., and Brunton, S. L. (2019).
Deep learning of dynamics and signal-noise decomposition with time-stepping constraints.
*J. Comput. Phys.*, 396:483–506.