MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

CSC
COMPUTATIONAL METHODS IN
SYSTEMS AND CONTROL THEORY

Physics-Informed Learning for Low-Dimensional Nonlinear Dynamical Systems Using Operator Inference

Part I: From Projection-based to Data-driven Model Order Reduction — an Overview

Peter Benner Pawan Goyal

IMPRS ProEng Summer School 2021
Magdeburg, September 27, 2021

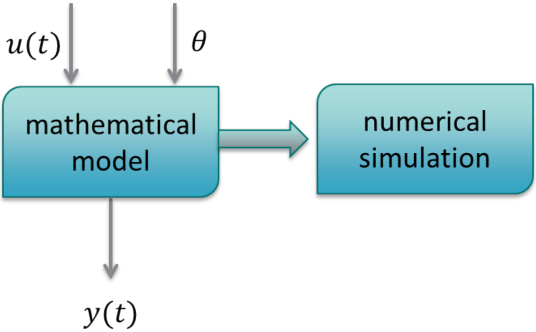
Supported by:

DFG-Graduiertenkolleg
MATHEMATISCHE
KOMPLEXITÄTSREDUKTION

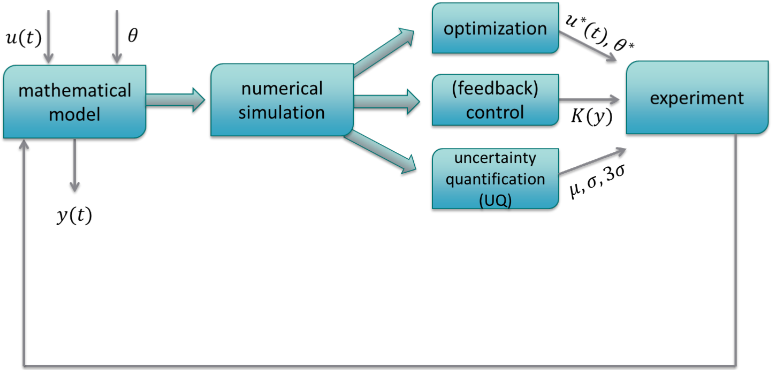
IMPRS
ProEng
Magdeburg

Partners:

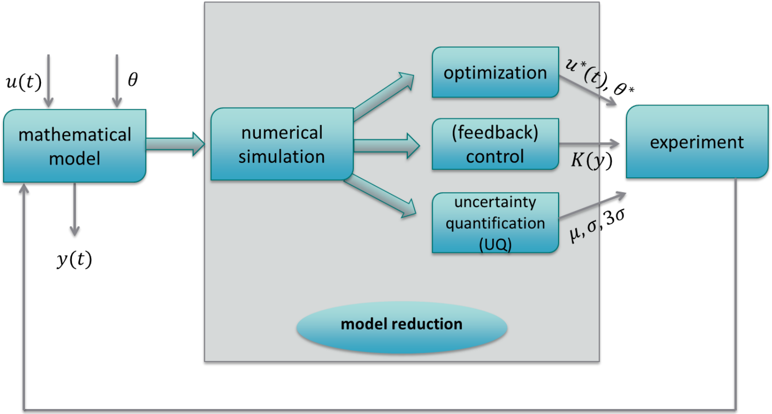
OTTO VON GUERICKE
UNIVERSITÄT
MAGDEBURG

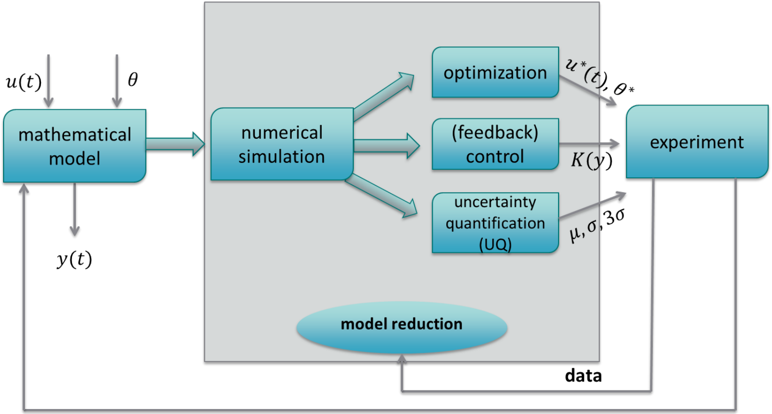


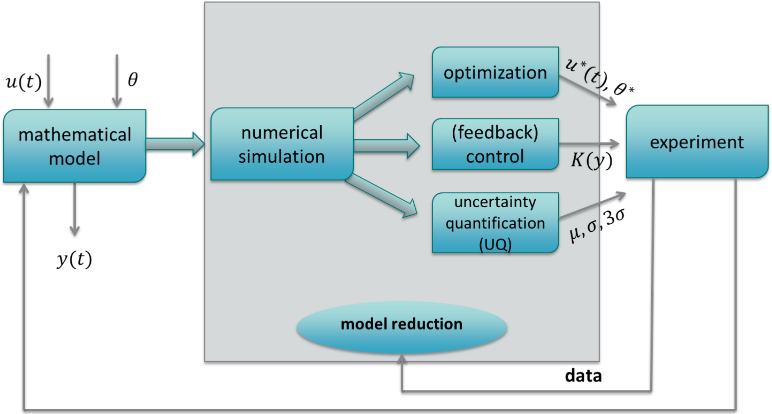
$u(t)$
θ
mathematical model
numerical simulation
$y(t)$



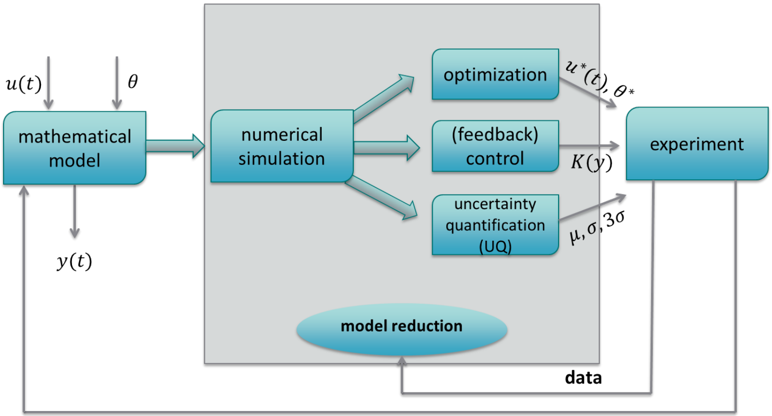
$u(t)$
θ
mathematical model
$y(t)$
numerical simulation
optimization
(feedback) control
uncertainty quantification (UQ)
$u^*(t), \theta^*$
$K(y)$
$\mu, \sigma, 3\sigma$
experiment


$u(t)$
θ
mathematical model
$y(t)$
numerical simulation
optimization
(feedback) control
uncertainty quantification (UQ)
$u^*(t), \theta^*$
$K(y)$
$\mu, \sigma, 3\sigma$
experiment
model reduction


$u(t)$
θ
mathematical model
$y(t)$
numerical simulation
optimization
(feedback) control
uncertainty quantification (UQ)
$u^*(t), \theta^*$
$K(y)$
$\mu, \sigma, 3\sigma$
experiment
model reduction
data



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



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

1. Model Order Reduction of Dynamical Systems
   - Model Order Reduction of Linear Systems
   - MOR Methods Based on Projection

2. Data-driven/-enhanced Model Reduction

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 $y(t) \in \mathbb{R}^p$.

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 $\hat{y}(t) \in \mathbb{R}^p$.

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:

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

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:

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

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

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)

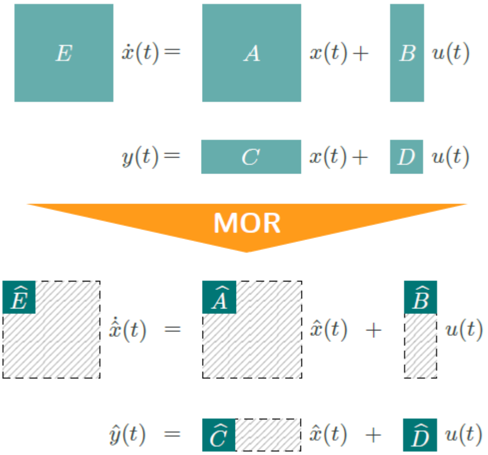
$$\widehat{\Sigma} : \begin{cases} \dot{\hat{x}}(t) = \widehat{A}\hat{x}(t) + \widehat{B}u(t), \\ \hat{y}(t) = \widehat{C}\hat{x}(t) + \widehat{D}u(t). \end{cases}$$

- 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:

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

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



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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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:

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

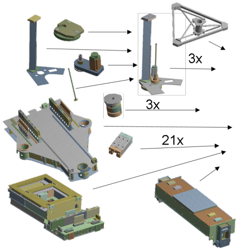
Needs hyperreduction if the cost for evaluation of the functions $W^T f, g$ is not reduced!

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. ...

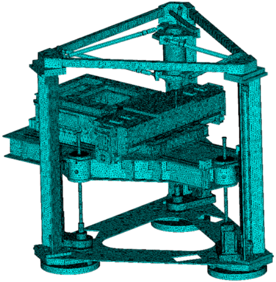
50 subassemblies



CAD model

$\overset{FEM}{\rightsquigarrow}$

FE-Model: 1.2M DOFs



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

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

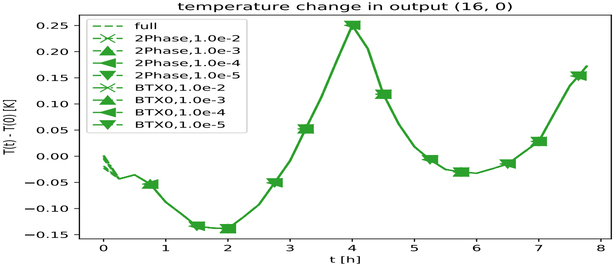
FE-coupled

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* ≈ 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

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

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 Σ, given $u(t)$ or $U(s)$:

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

Black box Σ: 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 Σ, given $u(t)$ or $U(s)$:

Black box Σ: 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 Σ, given $u(t)$ or $U(s)$:

Black box Σ: 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 Σ, given $u(t)$ or $U(s)$:

Black box Σ: 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 Σ, given $u(t)$ or $U(s)$:

Black box Σ: 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, ...]

Data-driven/-enhanced Model Reduction
Learning Models from Data

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

Black box Σ: 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 Σ, given $u(t)$ or $U(s)$:

Black box Σ: 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,...]

- 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.

- 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

$$
\begin{aligned}
x_{k+1} &= Ax_k + Bu_k + Kw_k, \\
y_k &= Cx_k + Du_k + v_k.
\end{aligned}
$$

  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

$$
\begin{aligned}
x_{k+1} &= Ax_k + Bu_k + Kw_k, \\
y_k &= Cx_k + Du_k + v_k.
\end{aligned}
$$

  from input-output data, given as time series $(u_0, y_0), (u_1, y_1), \dots, (u_K, y_K)$, 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 paper from 1990. . .

4 IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 1, NO. 1, MARCH 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 paper from 1990...

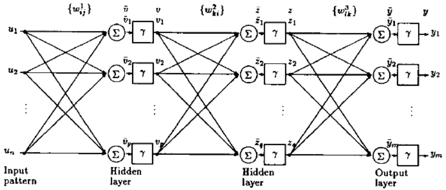


Fig. 2. A three layer neural network.

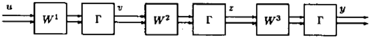
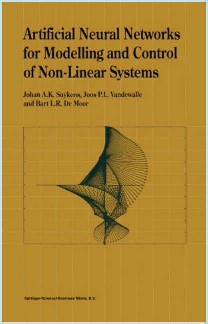


Fig. 3. A block diagram representation of a three layer network.

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 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.

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.

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.

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 $y(t) \in \mathbb{R}^p$.

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

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

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 $\beta > 0$, $q = 0, 1, 2$.

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.

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!).

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 $k = 0, 1, \dots, K$ 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).

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

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

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.

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

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

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):

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

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

- 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. (2021): Operator inference and physics-informed learning of low-dimensional models for incompressible flows. Elec. Trans. Numer. Anal., to appear. arXiv:2010.06701

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.