MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

CSC

COMPUTATIONAL METHODS IN
SYSTEMS AND CONTROL THEORY

Computational Learning of Dynamical Systems with Stability Constraints

Peter Benner

Joint work with Igor Pontes Duff
and Pawan K. Goyal (appliedAI Initiative, Heilbronn/Germany)

Computational Learning for Model Reduction
January 6–10, 2025
Institute for Computational & Experimental
Research in Mathematics (ICERM)
Providence, RI (USA)

Supported by:

DFG-Graduiertenkolleg
MATHEMATISCHE
KOMPLEXITÄTSREDUKTION

IMPRS
ProEng
Magdeburg

Partners:

OTTO VON GUERICKE
UNIVERSITÄT
MAGDEBURG

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.

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.

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)

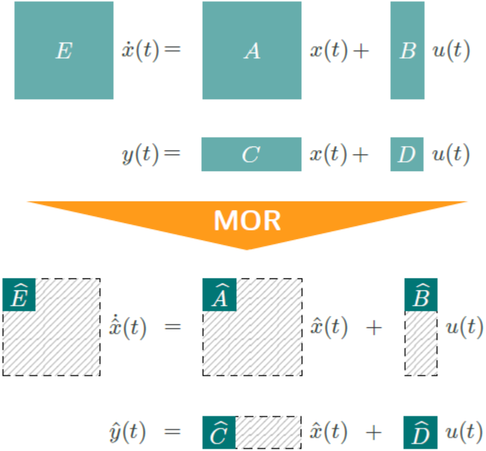
$$\widehat{\Sigma} : \begin{cases} \dot{\hat{x}}(t) = \widehat{A}\hat{x}(t) + \widehat{B}u(t), \\ \hat{y}(t) = \widehat{C}\hat{x}(t) + \widehat{D}u(t). \end{cases}$$

- 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:

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

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



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

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

From intrusive to non-intrusive MOR

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 (or surrogate) model** then is

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

From intrusive to non-intrusive MOR

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

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

The **reduced-order (or surrogate) model** then is

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

But: we need the matrices A, B, C, D to compute the reduced-order model!

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

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

The **reduced-order (or surrogate) model** then is

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

But: 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!

From intrusive to non-intrusive MOR

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

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

The **reduced-order (or surrogate) model** then is

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

But: 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!*

Now assume we are only given an oracle, allowing us to compute y (including cases with $y \equiv x$), given $u(t)$ or $U(s)$:

Now assume we are only given an oracle, allowing us to compute y (including cases with $y \equiv x$), 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 compute y (including cases with $y \equiv x$), 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 compute y (including cases with $y \equiv x$), 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, ...]

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

Now assume we are only given an oracle, allowing us to compute y (including cases with $y \equiv x$), 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 compute y (including cases with $y \equiv x$), 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, ...]

Now assume we are only given an oracle, allowing us to compute y (including cases with $y \equiv x$), 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, Klus, ...],
  for control systems [Kaiser/Kutz/Brunton 2017, B./Himpe/Mitchell 2018]

Now assume we are only given an oracle, allowing us to compute y (including cases with $y \equiv x$), 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, Klus, ...],
  for control systems [Kaiser/Kutz/Brunton 2017, B./Himpe/Mitchell 2018]
- Operator inference (OpInf): time domain [Kravtsov/Kondrashov/Ghil 2005, Peherstorfer/Willcox 2016; Kramer, Qian, Farcas, B., Goyal, Pontes Duff, Yildiz,...]

- 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.

- 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 the History of Learning Dynamical Systems (aka System Identification)

- 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].

- 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.

- 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 the History of Learning Dynamical Systems (aka System Identification)

- 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. . .

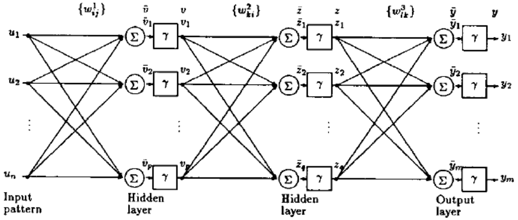


Fig. 2. A three layer neural network.

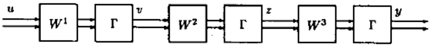
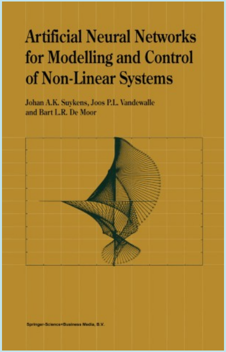


Fig. 3. A block diagram representation of a three layer network.

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

$$A_* := \operatorname{argmin}_{A \in \mathbb{R}^{n \times n}} \|X_1 - AX_0\|_F^2 + \mathcal{R}(A)$$

with a potential regularization term $\mathcal{R}(A)$, e.g., Tikhonov regularization aka kernel ridge regression: $\mathcal{R}(A) = \beta \|A\|_F^2$.

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

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):

$$(A_*, B_*, C_*, D_*) := \operatorname{argmin}_{(A,B,C,D)\in\mathbb{S}} \left\| \begin{bmatrix} X_1 \\ Y_0 \end{bmatrix} - \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} X_0 \\ U_0 \end{bmatrix} \right\|_F^2 + \mathcal{R}(A, B, C, D)$$

with a potential regularization term $\mathcal{R}(A, B, C, D)$.

- 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
- Mauroy, A., Mezić, I., Susuki, Y., eds., (2020): The Koopman Operator in Systems and Control. Concepts, Methodologies, and Applications. LNCIS 484, Springer, Cham.
- Gosea, I.V., Pontes Duff, I. (2021): Toward fitting structured nonlinear systems by means of dynamic mode decomposition. In Benner, P., et al, *Model Reduction of Complex Dynamical Systems*, ISNM 171, pp. 53–74, Birkhäuser, Basel.
- Morandin, R., Nicodemus, J., Unger, B. (2023): Port-Hamiltonian dynamic mode decomposition. SIAM J. Sci. Comp. 45(4):A1690–A1710. 10.1137/22M149329X

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

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

$$\hat{A}_* := \operatorname{argmin}_{\hat{A} \in \mathbb{R}^{r \times r}} \|\hat{X}_1 - \hat{A}\hat{X}_0\|_F^2 + \mathcal{R}(\hat{A}).$$

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

$$\hat{A}_* := \operatorname{argmin}_{\hat{A} \in \mathbb{R}^{r \times r}} \|\hat{X}_1 - \hat{A}\hat{X}_0\|_F^2 + \mathcal{R}(\hat{A}).$$

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 $P \otimes Q := \left[p_{ij}Q\right]_{ij}$ 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

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

- 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 $f(x_j, u_j)$ 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

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

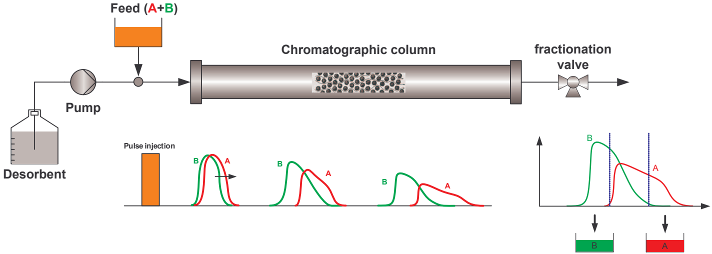
- Use compressed trajectories (via POD / PCA) $\rightsquigarrow \hat{X}$.
- Compress snapshot matrix of time derivatives: if residuals $f(x_j, u_j)$ are available

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

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

  $$(\hat{A}_*, \hat{H}_*, \hat{B}_*) := \operatorname{argmin}_{(\hat{A}, \hat{H}, \hat{B})} \left\| \dot{\hat{X}} - \begin{bmatrix} \hat{A} & \hat{H} & \hat{B} \end{bmatrix} \begin{bmatrix} \hat{X} \\ \hat{X} * \hat{X} \\ U \end{bmatrix} \right\|_F^2 + \mathcal{R}(\hat{A}, \hat{H}, \hat{B})$$

  with the Khatri-Rao product $\hat{X} * \hat{X} := [\hat{x}_0 \otimes \hat{x}_0, \ldots, \hat{x}_K \otimes \hat{x}_K]$.



- 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.

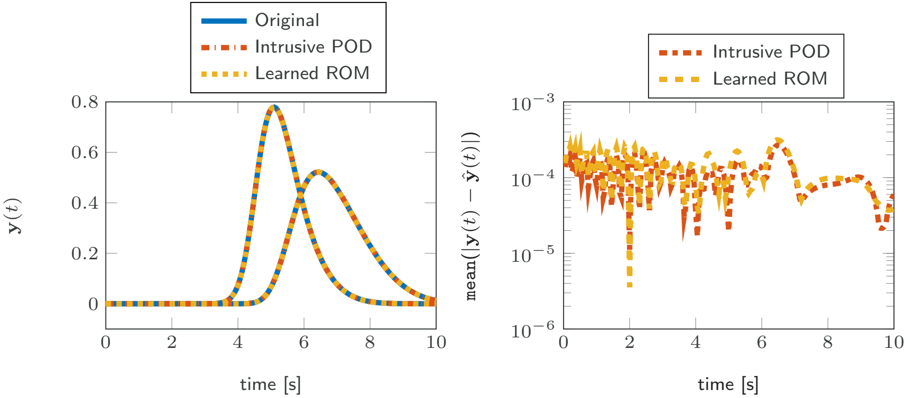


Figure: Batch chromatography example: A comparison of the POD intrusive model with the learned model of order $r = 4 \times 22$, where $n = 1600$ and $\mathrm{Pe} = 2000$.

- Parameterized shallow water equations are given by

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

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

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

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

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

Paramterized Operator Inference [Yildiz/Goyal/B./Karasözen 2021]

Numerical Example: Shallow Water Equations

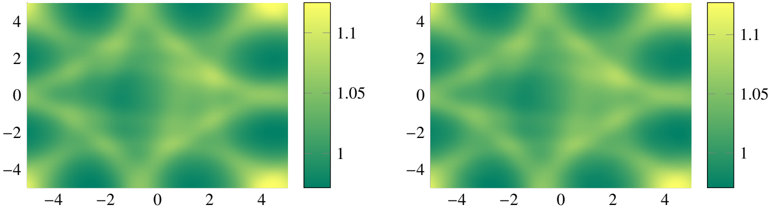
- Parameterized shallow water equations are given by

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

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

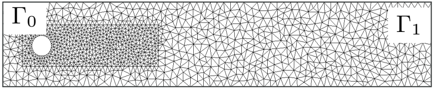
$$\frac{\partial}{\partial t}h = -(h\tilde{u})_x - (h\tilde{v})_y + \frac{1}{2}\delta\cos\theta(h^2)_x.$$

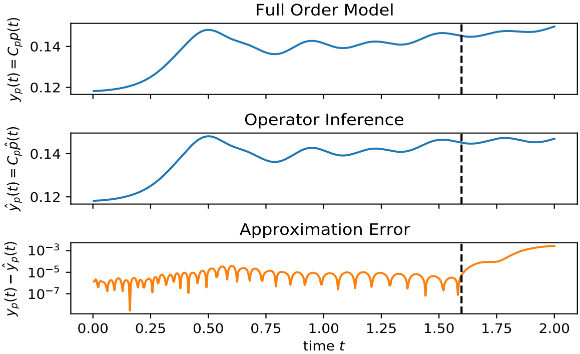
- Comparison of the height field for the parameter $\theta = \frac{5\pi}{24}$:



(a) FOM (b) Learned parametric model

Tailored operator inference for incompressible Navier-Stokes equations, by heeding incompressibility condition.





Problem: OpInf regression potentially yields unstable dynamics.
$\rightsquigarrow$ Even marginal instability can lead to unphysical simulation results.

Preserving Stability in Operator Inference

Problem: OpInf regression potentially yields unstable dynamics.
$\rightsquigarrow$ Even marginal instability can lead to unphysical simulation results.

Goal: infer systems with guaranteed stability.

Asymptotic (exponential, Lyapunov) stability of linear systems

$$\dot{x}(t) = Ax(t), \qquad x(0) = x_0,$$

can be explicitly parameterized:

Theorem (Gillis/Sharma 2017)

A matrix $A \in \mathbb{R}^{n \times n}$ is asymptotically stable (Hurwitz, Lyapunov stable) if and only if it can be represented as

$$A = (J - R)Q,$$

where $J = -J^T$ and $R = R^T$, $Q = Q^T$ are both positive definite.

Asymptotic (exponential, Lyapunov) stability of linear systems

$$\dot{x}(t) = Ax(t), \qquad x(0) = x_0,$$

can be explicitly parameterized:

Theorem (Gillis/Sharma 2017)

A matrix $A \in \mathbb{R}^{n \times n}$ is asymptotically stable (Hurwitz, Lyapunov stable) if and only if it can be represented as

$$A = (J - R)Q,$$

where $J = -J^T$ and $R = R^T$, $Q = Q^T$ are both positive definite.

$\Longrightarrow$ **Stability-preserving OpInf for linear systems** [Goyal/Pontes Duff/B. 2023]:

$$(S_*, L_*, K_*) := \operatorname{argmin}_{\substack{L,K \text{ upper triangular} \\ \text{with positive diagonals}}} \left(\|\dot{X} - (S - S^T - L^T L) K^T K X\|_F^2 + \mathcal{R}(L, K, S) \right).$$

The matrix obtained from this nonlinear (regularized) least-squares problem,

$$A_* = \left(S_* - S_*^T - L_*^T L_* \right) K_*^T K_*,$$

is guaranteed to be stable due to [Gillis/Sharma 2017].

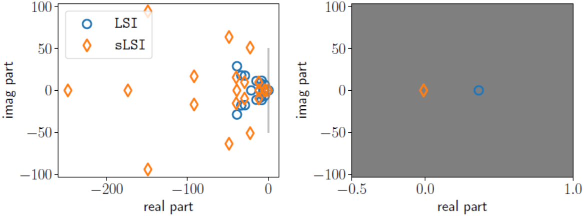
Related work by *Schwerdtner/Voigt*, *Unger*, ...

Consider 1D Burgers' equation for viscous flow

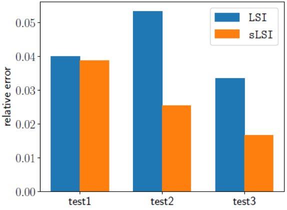
$$\begin{aligned} v_t + v v_x &= \nu v_{xx} \text{ in } (0,1)\times(0,T) \\ v_x(0,t) &= v_x(1,t) = 0, \\ v(x,0) &= v_0(x,\mu), \end{aligned}$$

discretized on uniform 1000×500 space-time grid for $17+3$ training+testing initial conditions.

Reduced-order model ($r = 21$) computed using standard ("LSI") and stabilized ("SLSI") OpInf applied to (POD)-projected data.
(Implementation using `PyTorch` and `Adam` optimizer for solving nonlinear regression problem.)



Eigenvalues of linearization



Errors for different initial conditions (test data)

Solving the OpInf regression problem

$$(A_*, H_*) := \operatorname{argmin}_{(A,H)} \left\| \dot{X} - \begin{bmatrix} A & H \end{bmatrix} \begin{bmatrix} X \\ X * X \end{bmatrix} \right\|_F^2 + \mathcal{R}(A\,H)$$

using the stability-constraint on A as just discussed leads to a nonlinear system with local Lyapunov stability, noting that the inferred $Q_* = K_*^T K_* > 0$ provides a quadratic Lyapunov function for the identified system [Goyal/Pontes Duff/B. 2023].

Solving the OpInf regression problem

$$(A_*, H_*) := \operatorname{argmin}_{(A,H)} \left\| \dot{X} - \begin{bmatrix} A & H \end{bmatrix} \begin{bmatrix} X \\ X * X \end{bmatrix} \right\|_F^2 + \mathcal{R}(A\, H)$$

using the stability-constraint on A as just discussed leads to a nonlinear system with local Lyapunov stability, noting that the inferred $Q_* = K_*^T K_* > 0$ provides a quadratic Lyapunov function for the identified system [Goyal/Pontes Duff/B. 2023].

We can achieve more for energy-preserving quadratic systems, i.e.,

$$H_{ijk} + H_{ikj} + H_{jik} + H_{jki} + H_{kij} + H_{kji} = 0 \quad \text{for all } i, j, k \in \{1, \dots, n\}.$$

Note: the latter is equivalent to $x^T H(x \otimes x) = 0$ for all $x \in \mathbb{R}^n$ [Schlegel/Noack 2015].

Solving the OpInf regression problem

$$(A_*, H_*) := \operatorname{argmin}_{(A,H)} \left\| \dot{X} - \begin{bmatrix} A & H \end{bmatrix} \begin{bmatrix} X \\ X * X \end{bmatrix} \right\|_F^2 + \mathcal{R}(A\,H)$$

using the stability-constraint on A as just discussed leads to a nonlinear system with local Lyapunov stability, noting that the inferred $Q_* = K_*^T K_* > 0$ provides a quadratic Lyapunov function for the identified system [Goyal/Pontes Duff/B. 2023].

We can achieve more for energy-preserving quadratic systems, i.e.,

$$H_{ijk} + H_{ikj} + H_{jik} + H_{jki} + H_{kij} + H_{kji} = 0 \quad \text{for all } i, j, k \in \{1, \ldots, n\}.$$

Note: the latter is equivalent to $x^T H(x \otimes x) = 0$ for all $x \in \mathbb{R}^n$ [Schlegel/Noack 2015].

Theorem (Goyal/Pontes Duff/B. 2023)

An energy-preserving quadratic system

$$\dot{z} = Az + H(z \otimes z)$$

is monotonically and globally asymptotically stable if and only if the symmetric part of A is asymptotically stable.

Theorem (Goyal/Pontes Duff/B. 2023)

An energy-preserving quadratic system

$$\dot{z} = Az + H(z \otimes z)$$

is monotonically and globally asymptotically stable (GAS) if and only if the symmetric part of A is asymptotically stable.

Question: can we encode the energy-preservation property explicitly, so that we constrain the OpInf problem accordingly? (If the answer is yes, then we can learn a GAS model using OpInf.)

Theorem (Goyal/Pontes Duff/B. 2023)

An energy-preserving quadratic system

$$\dot{z} = Az + H(z \otimes z)$$

is monotonically and globally asymptotically stable (GAS) if and only if the symmetric part of A is asymptotically stable.

Question: can we encode the energy-preservation property explicitly, so that we constrain the OpInf problem accordingly? (If the answer is yes, then we can learn a GAS model using OpInf.)

Answer: yes, we can!

Theorem (Goyal/Pontes Duff/B. 2023)

An energy-preserving quadratic system

$$\dot{z} = Az + H(z \otimes z)$$

is monotonically and globally asymptotically stable (GAS) if and only if the symmetric part of A is asymptotically stable.

Question: can we encode the energy-preservation property explicitly, so that we constrain the OpInf problem accordingly? (If the answer is yes, then we can learn a GAS model using OpInf.)

Answer: yes, we can!

Theorem (Goyal/Pontes Duff/B. 2023)

A locally Lyapunov stable quadratic system in $\mathbb{R}^n$

$$\dot{z} = Az + H(z \otimes z), \qquad A = (J - R)Q,\ J = -J^T,\ R = R^T > 0,\ Q = Q^T > 0,$$

is generalized energy-preserving w.r.t. Q, i.e., $x^T QH(x \otimes x) = 0$ for all x, if

$$H = [H_1 Q, \ldots, H_n Q], \quad \text{where} \quad H_j = -H_j^T, \quad j = 1, \ldots, n.$$

Moreover, $V(x) = \frac{1}{2} x^T Qx$ is a global Lyapunov function for the quadratic system.

Constrained OpInf problem for learning GAS systems [Goyal/Pontes Duff/B. 2023]

$$(A_*, H_*) := \operatorname{argmin}_{(A,H)} \left\| \dot{X} - \begin{bmatrix} A & H \end{bmatrix} \begin{bmatrix} X \\ X * X \end{bmatrix} \right\|_F^2 + \mathcal{R}(A\,H)$$

subject to the stability constraints

$$A = \left(S - S^T - L^T L\right) K^T K \quad \text{with } L, K \text{ upper triangular with positive diagonals}$$

$$H = [H_1 Q, \ldots, H_n Q], \quad \text{with} \quad H_j = -H_j^T, \quad j = 1, \ldots, n.$$

Constrained OpInf problem for learning GAS systems [Goyal/Pontes Duff/B. 2023]

$$(A_*, H_*) := \operatorname{argmin}_{(A,H)} \left\| \dot{X} - \begin{bmatrix} A & H \end{bmatrix} \begin{bmatrix} X \\ X * X \end{bmatrix} \right\|_F^2 + \mathcal{R}(A\, H)$$

subject to the stability constraints

$$A = \left(S - S^T - L^T L\right) K^T K \quad \text{with } L, K \text{ upper triangular with positive diagonals}$$

$$H = [H_1 Q, \ldots, H_n Q], \quad \text{with} \quad H_j = -H_j^T, \quad j = 1, \ldots, n.$$

Implementation:

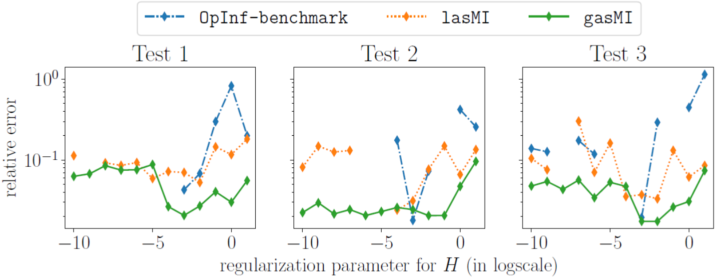
- Usually, as discussed before, the data are projected onto the leading r PCA modes for dimension reduction.
- Quite involved optimization problem, can be solved via stochastic gradient descent (Adam) and backpropagation (setting $Q = I_r$ may be necessary).
- We do not explicitly need derivative data by using a Neural ODE approach for noisy data [Goyal/B. 2023].

Consider again 1D Burgers' equation for viscous flow

$$\begin{aligned} v_t + v v_x &= \nu v_{xx} \text{ in } (0,1)\times(0,T) \\ v(0,t) &= v(1,t) = 0, \\ v(x,0) &= v_0(x,\mu), \end{aligned}$$

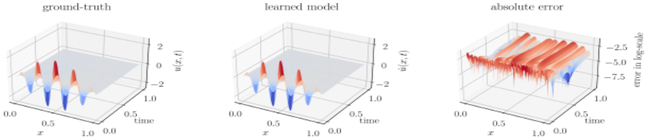
discretized on uniform 250×500 space-time grid for $17+3$ training+testing initial conditions and $\nu = 0.05$.

Reduced-order model ($r = 20$) computed using standard, locally stable (lasMI) and globally stable (gasMI) OpInf applied to (POD)-projected data.
(Implementation using `PyTorch` and `Adam` optimizer for solving nonlinear regression problem.)

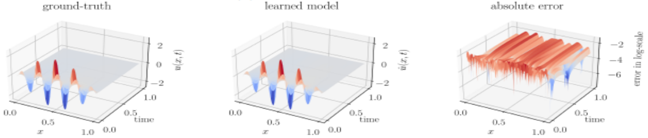


(missing data = blow-up in numerical simulation)

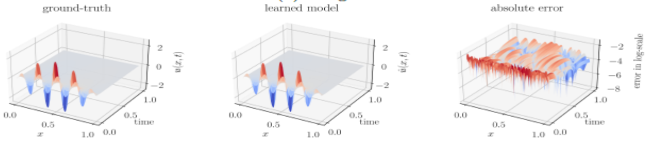
Consider again 1D Burgers' equation for viscous flow



(a) Using **opInf-benchmark**.



(b) Using **lasMI**.



(c) Using **gasMI**.

Full simulation for test initial condition (not seen during training)

- So far, we considered asymptotically stable systems.

- So far, we considered asymptotically stable systems.
- However, there exist quadratic systems without any stable points, e.g., chaotic Lorenz example.

- So far, we considered asymptotically stable systems.
- However, there exist quadratic systems without any stable points, e.g., chaotic Lorenz example.
- Despite having no stable point, these systems might have an attractor, meaning there exists a bounded region (a ball) where all trajectories for some set of initial conditions get trapped. (Attractor is sometimes also called "trapping region".) Call such systems ATR systems.

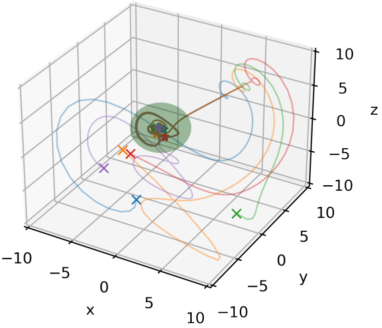


Figure: An illustration of nonlinear dynamics with attractor.

- So far, we considered asymptotically stable systems.
- However, there exist quadratic systems without any stable points, e.g., chaotic Lorenz example.
- Despite having no stable point, these systems might have an attractor, meaning there exists a bounded region (a ball) where all trajectories for some set of initial conditions get trapped. (Attractor is sometimes also called "trapping region".) Call such systems ATR systems.

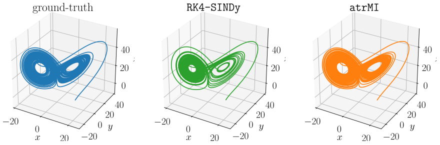
Inference of ATR quadratic systems [Goyal/Pontes Duff/B. 2023]

- It can be shown that for energy-preserving quadratic systems, an ATR system can be turned into a GAS system by translation $x(t) \to x(t) - y$
- We, thus, require to solve the following constraint problem:

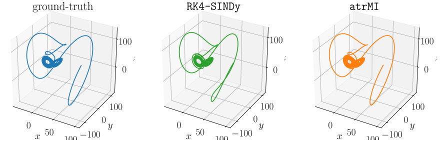
$$\min_{A,H,y} \left\| \dot{X} - A(X-y) - H\left((X-y) * (X-y)\right) \right\|$$

subject to $\Lambda(A) \in \mathbb{C}^-$ and H is energy preserving.

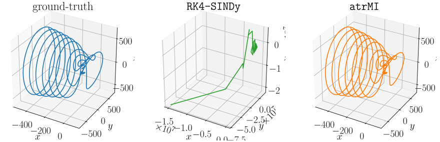
- Note that we do not know y a priori, it is learned from the data.



(a) For initial condition $[10, 10, -10]$.



(b) For initial condition $[100, -100, 100]$.



(c) For initial condition $[-500, 500, 500]$.

A comparison of the time-domain simulations of the learned models for testing initial conditions.

Conclusions

- Operator inference (OpInf) is a regression-based powerful method to infer linear and certain nonlinear dynamical systems from data, very similar to DMD in the linear case.
- Looks simple, but the devil is in the details.
- Stability constraints can be encoded explicitly in the regression problem for the model inference.
- For application to control problems, see MTNS2024 contribution by Pontes Duff [Pontes Duff/Goyal/B. 2024].
- The same approach can also be used to infer stable systems using sparse regression (SINDy).
- Recent work combines OpInf with neural networks to solve nonlinear parametric identification problems with stability guarantee (preprint coming out soon).
- Error bounds for non-intrusive MOR need to be further developed.
- Better solvers for special nonlinear regression problems needed!

References

Kravtsov, S., Kondrashov, D., Ghil, M. (2005): Multilevel regression modeling of nonlinear processes: Derivation and applications to climatic variability. J. Climate, 18(21):4404–4424.

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.

Goyal, P., Benner, P. (2023): Neural ordinary differential equations with irregular and noisy data. Royal Society Open Science, 10(7):221475.

Goyal, P., Pontes Duff, I., Benner, P. (2023): Inference of continuous linear systems from data with guaranteed stability. arXiv:2301.10060

Goyal, P., Pontes Duff, I., Benner, P. (2023): Guaranteed stable quadratic models and their applications in SINDy and operator inference. arXiv:2308.13819

Pontes Duff, I., Goyal, P., Benner, P. (2024): Stability-Certified Learning of Control Systems with Quadratic Nonlinearities. Proc. MTNS 2024 / arXiv:2403.00646.