Robust Data-Driven Control with Noisy Data

Chin-Yao Chang

National Renewable Energy Laboratory

5th Autonomous Energy System Workshop, Golden, CO, July 12-15, 2022

This work was authored by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by DOE Office of Electricity, Advanced Grid Modeling Program, through agreement NO. 33652. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government.

July 15, 2022

From Data to More Robust/Resilient Power Systems

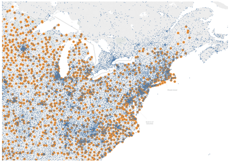
Availability of Data

- Many new data sources
- Noisy (disturbance/asynchrony)
- Sparse sensors
- Different time constants

Control of Power Systems

- **Robustness** - stay stable under uncertainty/unexpected events
- **Resiliency** - quick restoration from abrupt changes/failures

How to maximize the values of newly available data for robust and resilient control of power systems with increased uncertainties?



From Data to More Robust/Resilient Power Systems

How to maximize the values of newly available data for robust and resilient control of power systems with increased uncertainties?

- Machine learning alone most likely won't do because of the high priority of robustness and resiliency
- Classical *system ID and controller design* may struggle dealing with the data collected (scalability, noisy sources, etc.)

From Data to More Robust/Resilient Power Systems

How to maximize the values of newly available data for robust and resilient control of power systems with increased uncertainties?

- Machine learning alone most likely won't do because of the high priority of robustness and resiliency
- Classical *system ID and controller design* may struggle dealing with the data collected (scalability, noisy sources, etc.)

A system ID framework that provides insights for

1. What amount of data is sufficient
2. In what scenarios would supervised learning help
3. Bounds for the modeling errors originated from noisy data
4. Methods of pre-processing data matrices to reduce the errors

Given the error bounds, we can build robust controllers

High-Level Ideas of the System ID Framework

- Inspired by behavioral system theory
  - Originally developed in the 80's [J. C. Willems 1986]
  - Recent resurgence with new insights [J. Coulson, J Lygeros, F Dorfler 2019] and [C. De Persis and P. Tesi 2019]

High-Level Ideas of the System ID Framework

- Inspired by behavioral system theory
  - Originally developed in the 80's [J. C. Willems 1986]
  - Recent resurgence with new insights [J. Coulson, J Lygeros, F Dorfler 2019] and [C. De Persis and P. Tesi 2019]
- Formulate system ID of *linear systems, switched linear systems, nonlinear systems* as solving the following equation

$$Y = A\Psi(U)$$

where U and Y are resp. constructed by the input and output data

High-Level Ideas of the System ID Framework

- Inspired by behavioral system theory
  - Originally developed in the 80's [J. C. Willems 1986]
  - Recent resurgence with new insights [J. Coulson, J Lygeros, F Dorfler 2019] and [C. De Persis and P. Tesi 2019]
- Formulate system ID of *linear systems, switched linear systems, nonlinear systems* as solving the following equation

$$Y = A\Psi(U)$$

  where U and Y are resp. constructed by the input and output data
- If Ψ is known, system ID is all about solving the linear equations for A

High-Level Ideas of the System ID Framework

- Inspired by behavioral system theory
  - Originally developed in the 80's [J. C. Willems 1986]
  - Recent resurgence with new insights [J. Coulson, J Lygeros, F Dorfler 2019] and [C. De Persis and P. Tesi 2019]
- Formulate system ID of *linear systems, switched linear systems, nonlinear systems* as solving the following equation

$$Y = A\Psi(U)$$

  where U and Y are resp. constructed by the input and output data
- If Ψ is known, system ID is all about solving the linear equations for A
- Supervised learning can help fill the gap of unknown parts of the basis function Ψ

High-Level Ideas of the System ID Framework

- Inspired by behavioral system theory
  - Originally developed in the 80's [J. C. Willems 1986]
  - Recent resurgence with new insights [J. Coulson, J Lygeros, F Dorfler 2019] and [C. De Persis and P. Tesi 2019]
- Formulate system ID of *linear systems, switched linear systems, nonlinear systems* as solving the following equation

$$Y = A\Psi(U)$$

  where U and Y are resp. constructed by the input and output data
- If Ψ is known, system ID is all about solving the linear equations for A
- Supervised learning can help fill the gap of unknown parts of the basis function Ψ
- Analyzing the equations $Y = A\Psi(U)$ provides great insights on the *bound of the modeling errors* and *pre-processing methods*

Case 1: Linear Systems

- System model: $x(k+1) = Ax(k) + Bu(k)$

Case 1: Linear Systems

- System model: $x(k+1) = Ax(k) + Bu(k)$
- Given the measured data for $k = 0, \cdots, T$, define

$$
\begin{aligned}
U_0 &= [u_d(0), \cdots, u_d(T-1)], \\
X_0 &= [x_d(0), \cdots, x_d(T-1)], \\
X_1 &= [x_d(1), \cdots, x_d(T)],
\end{aligned}
$$

Case 1: Linear Systems

- System model: $x(k+1) = Ax(k) + Bu(k)$
- Given the measured data for $k = 0, \cdots, T$, define

$$U_0 = [u_d(0), \cdots, u_d(T-1)],$$
$$X_0 = [x_d(0), \cdots, x_d(T-1)],$$
$$X_1 = [x_d(1), \cdots, x_d(T)],$$

- The system matrices can then be identified directly:

$$X_1 = \begin{bmatrix} B & A \end{bmatrix} \begin{bmatrix} U_0 \\ X_0 \end{bmatrix}; \qquad \begin{bmatrix} B & A \end{bmatrix} = X_1 \begin{bmatrix} U_0 \\ X_0 \end{bmatrix}^{\dagger}.$$

Case 1: Linear Systems

- System model: $x(k+1) = Ax(k) + Bu(k)$
- Given the measured data for $k = 0, \cdots, T$, define

$$U_0 = [u_d(0), \cdots, u_d(T-1)],$$
$$X_0 = [x_d(0), \cdots, x_d(T-1)],$$
$$X_1 = [x_d(1), \cdots, x_d(T)],$$

- The system matrices can then be identified directly:

$$X_1 = \begin{bmatrix} B & A \end{bmatrix} \begin{bmatrix} U_0 \\ X_0 \end{bmatrix}; \qquad \begin{bmatrix} B & A \end{bmatrix} = X_1 \begin{bmatrix} U_0 \\ X_0 \end{bmatrix}^{\dagger}.$$

- Require full row rank of $\begin{bmatrix} U_0 \\ X_0 \end{bmatrix}$ *(translated to the persistently exciting condition in the context of behavioral system theory)*

Case 2: Switched Linear Systems

- System model: $x(k+1) = A_{\sigma(k)}x(k) + B_{\sigma(k)}u(k), \quad \sigma(k) \in \Gamma$

Case 2: Switched Linear Systems

- System model: $x(k+1) = A_{\sigma(k)}x(k) + B_{\sigma(k)}u(k), \quad \sigma(k) \in \Gamma$
- Given the measured data for $k = 0, \cdots, T$, define $U_{i,0}$ and $X_{i,0}$ for each i as:

$$U_{i,0} = [u_{i,\sigma}(0), \cdots, u_{i,\sigma}(T-1)],$$
$$X_{i,0} = [x_{i,\sigma}(0), \cdots, x_{i,\sigma}(T-1)],$$

where for every $k = 0, \cdots, T$,

$$u_{i,\sigma}(k) = \begin{cases} u_\sigma(k) & \text{if } \sigma(k) = i \\ 0 & \text{otherwise} \end{cases}, \qquad x_{i,\sigma}(k) = \begin{cases} x_\sigma(k) & \text{if } \sigma(k) = i \\ 0 & \text{otherwise} \end{cases}.$$

Case 2: Switched Linear Systems

- System model: $x(k+1) = A_{\sigma(k)}x(k) + B_{\sigma(k)}u(k), \quad \sigma(k) \in \Gamma$
- Given the measured data for $k = 0, \cdots, T$, define $U_{i,0}$ and $X_{i,0}$ for each i as:

$$U_{i,0} = [u_{i,\sigma}(0), \cdots, u_{i,\sigma}(T-1)],$$
$$X_{i,0} = [x_{i,\sigma}(0), \cdots, x_{i,\sigma}(T-1)],$$

where for every $k = 0, \cdots, T$,

$$u_{i,\sigma}(k) = \begin{cases} u_\sigma(k) & \text{if } \sigma(k) = i \\ 0 & \text{otherwise} \end{cases}, \qquad x_{i,\sigma}(k) = \begin{cases} x_\sigma(k) & \text{if } \sigma(k) = i \\ 0 & \text{otherwise} \end{cases}.$$

- It is still about solving $X_1 = \sum_{i\in\Gamma}[B_i \ A_i] \begin{bmatrix} U_{i,0} \\ X_{i,0} \end{bmatrix}$

Case 2: Switched Linear Systems

- System model: $x(k+1) = A_{\sigma(k)}x(k) + B_{\sigma(k)}u(k), \quad \sigma(k) \in \Gamma$
- Given the measured data for $k = 0, \cdots, T$, define $U_{i,0}$ and $X_{i,0}$ for each i as:

$$U_{i,0} = [u_{i,\sigma}(0), \cdots, u_{i,\sigma}(T-1)],$$
$$X_{i,0} = [x_{i,\sigma}(0), \cdots, x_{i,\sigma}(T-1)],$$

where for every $k = 0, \cdots, T$,

$$u_{i,\sigma}(k) = \begin{cases} u_\sigma(k) & \text{if } \sigma(k) = i \\ 0 & \text{otherwise} \end{cases}, \qquad x_{i,\sigma}(k) = \begin{cases} x_\sigma(k) & \text{if } \sigma(k) = i \\ 0 & \text{otherwise} \end{cases}.$$

- It is still about solving $X_1 = \sum_{i\in\Gamma}[B_i \; A_i] \begin{bmatrix} U_{i,0} \\ X_{i,0} \end{bmatrix}$
- For each mode i, we can find a $\begin{bmatrix} U_{i,0} \\ X_{i,0} \end{bmatrix}^\dagger$ such that $[B_i \; A_i] = X_1 \begin{bmatrix} U_{i,0} \\ X_{i,0} \end{bmatrix}^\dagger$

Case 3: Nonlinear System with Known Basis

- System model: $y = f(u) = A\phi(u)$, where ϕ is a basis

Case 3: Nonlinear System with Known Basis

- System model: $y = f(u) = A\phi(u)$, where ϕ is a basis
- Arrange the measured data by

$$\Phi_0 = [\phi(u_d(0)), \cdots, \phi(u_d(T-1))],$$
$$Y_0 = [y_d(0), \cdots, y_d(T-1)],$$

- It is still about solving

$$Y_0 = A\Phi_0 \qquad \text{or} \qquad A = Y_0{\Phi_0}^\dagger$$

Case 3: Nonlinear System with Known Basis

- System model: $y = f(u) = A\phi(u)$, where ϕ is a basis
- Arrange the measured data by

$$\Phi_0 = [\phi(u_d(0)), \cdots, \phi(u_d(T-1))],$$
$$Y_0 = [y_d(0), \cdots, y_d(T-1)],$$

- It is still about solving

$$Y_0 = A\Phi_0 \qquad \text{or} \qquad A = Y_0{\Phi_0}^{\dagger}$$

- Direct data representation applies to dynamical or static systems

Case 3: Nonlinear System with Known Basis

- System model: $y = f(u) = A\phi(u)$, where ϕ is a basis
- Arrange the measured data by

$$\Phi_0 = [\phi(u_d(0)), \cdots, \phi(u_d(T-1))],$$
$$Y_0 = [y_d(0), \cdots, y_d(T-1)],$$

- It is still about solving

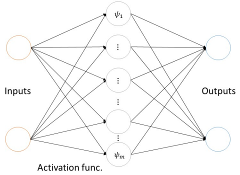
$$Y_0 = A\Phi_0 \qquad \text{or} \qquad A = Y_0{\Phi_0}^{\dagger}$$

- Direct data representation applies to dynamical or static systems
- *The full row rank condition can be understood as the condition for sufficient richness of the data for identifying the full underlying system*

Case 4: Supervised Learning

- Define $\psi(z)$ as the nonlinear activation function of the neurons, e.g., ReLU functions
- The "model" of a single layer ANN is then written as

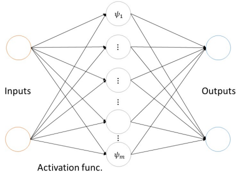
$$y = A_0 \psi(A_1 u + b_1) + b_0,$$



Case 4: Supervised Learning

- Define $\psi(z)$ as the nonlinear activation function of the neurons, e.g., ReLU functions
- The "model" of a single layer ANN is then written as

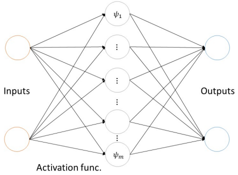
$$y = A_0 \psi(A_1 u + b_1) + b_0,$$



- The equation above is structurally identical to $y = A\phi(u)$

Case 4: Supervised Learning

- Define $\psi(z)$ as the nonlinear activation function of the neurons, e.g., ReLU functions
- The "model" of a single layer ANN is then written as

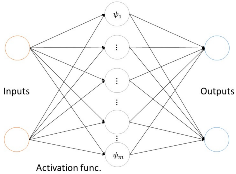


$$y = A_0 \psi(A_1 u + b_1) + b_0,$$

- The equation above is structurally identical to $y = A\phi(u)$
- Supervised learning is simultaneously identifying A_0 and finding the most proper basis functions through selections of A_1 and b_1

Case 4: Supervised Learning

- Define $\psi(z)$ as the nonlinear activation function of the neurons, e.g., ReLU functions
- The "model" of a single layer ANN is then written as

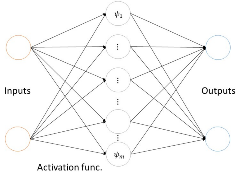


$$y = A_0\psi(A_1 u + b_1) + b_0,$$

- The equation above is structurally identical to $y = A\phi(u)$
- Supervised learning is simultaneously identifying A_0 and finding the most proper basis functions through selections of A_1 and b_1
- Choosing the number of neurons, activation functions, the number of hidden layers (for non-smooth problems) are effectively guessing a proper structure of the basis functions

Case 4: Supervised Learning

- Define $\psi(z)$ as the nonlinear activation function of the neurons, e.g., ReLU functions
- The "model" of a single layer ANN is then written as



$$y = A_0 \psi(A_1 u + b_1) + b_0,$$

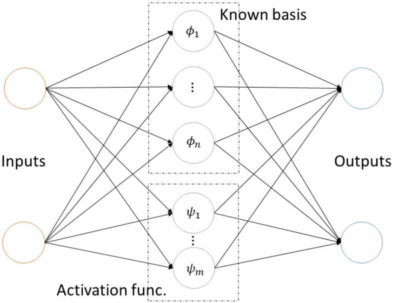
- The equation above is structurally identical to $y = A\phi(u)$
- Supervised learning is simultaneously identifying A_0 and finding the most proper basis functions through selections of A_1 and b_1
- Choosing the number of neurons, activation functions, the number of hidden layers (for non-smooth problems) are effectively guessing a proper structure of the basis functions
- *If the basis functions are known, then there is no benefit of applying supervised learning because SL requires a lot more data and computational complexity compared to straight solving linear equations*

Physics Aware ANN

For cases with known basis functions, combining the earlier results of switched linear and nonlinear systems would probably be sufficient

Physics Aware ANN

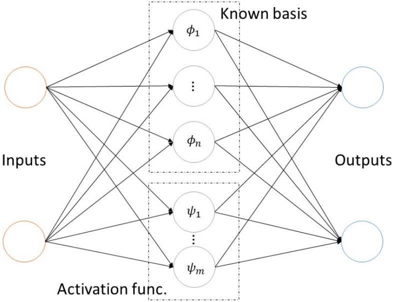
For cases with known basis functions, combining the earlier results of switched linear and nonlinear systems would probably be sufficient



- If only part of the basis functions are known, we can formulate physics aware ANN as shown in the figure

Physics Aware ANN

For cases with known basis functions, combining the earlier results of switched linear and nonlinear systems would probably be sufficient

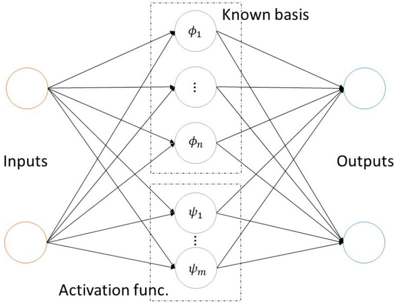


- If only part of the basis functions are known, we can formulate physics aware ANN as shown in the figure
- E.g., a system that involves power flow equations

$$\begin{bmatrix} P \\ Q \end{bmatrix} = A \begin{bmatrix} |V_{l_1}||V_{l_2}|\sin(\Delta\theta_l) \\ |V_{l_1}||V_{l_2}|\cos(\Delta\theta_l) \end{bmatrix}$$

Physics Aware ANN

For cases with known basis functions, combining the earlier results of switched linear and nonlinear systems would probably be sufficient



- If only part of the basis functions are known, we can formulate physics aware ANN as shown in the figure
- E.g., a system that involves power flow equations

$$\begin{bmatrix} P \\ Q \end{bmatrix} = A \begin{bmatrix} |V_{l_1}||V_{l_2}|\sin(\Delta\theta_l) \\ |V_{l_1}||V_{l_2}|\cos(\Delta\theta_l) \end{bmatrix}$$

- It is still about solving $Y = A\Psi(U)$

Bounds of Modeling Errors

Data are most likely noisy with presence of measurement errors, latency, and sometimes only pseudo-measurements are available

- Focus on the analysis on $Y = A\Psi$ (or $Y = A\Psi(U)$)

Bounds of Modeling Errors

Data are most likely noisy with presence of measurement errors, latency, and sometimes only pseudo-measurements are available

- Focus on the analysis on $Y = A\Psi$ (or $Y = A\Psi(U)$)
- The estimated model straight from the data is given as $A^e = Y\Psi^\dagger$

Bounds of Modeling Errors

Data are most likely noisy with presence of measurement errors, latency, and sometimes only pseudo-measurements are available

- Focus on the analysis on $Y = A\Psi$ (or $Y = A\Psi(U)$)
- The estimated model straight from the data is given as $A^e = Y\Psi^\dagger$
- The accurate $Y^\star$ and $\Psi^\star$ that generate the actual model: $A = Y^\star {\Psi^\star}^\dagger$

Bounds of Modeling Errors

Data are most likely noisy with presence of measurement errors, latency, and sometimes only pseudo-measurements are available

- Focus on the analysis on $Y = A\Psi$ (or $Y = A\Psi(U)$)
- The estimated model straight from the data is given as $A^e = Y\Psi^\dagger$
- The accurate $Y^\star$ and $\Psi^\star$ that generate the actual model: $A = Y^\star {\Psi^\star}^\dagger$
- Define $\delta Y = Y^\star - Y$ and $\delta\Psi = \Psi^\star - \Psi$ and assume

$$\frac{\|\delta Y\|}{\|Y^\star\|} \leq r_Y, \quad \frac{\|\delta\Psi\|}{\|\Psi^\star\|} \leq r_\Psi < 1.$$

Bounds of Modeling Errors

Data are most likely noisy with presence of measurement errors, latency, and sometimes only pseudo-measurements are available

- Focus on the analysis on $Y = A\Psi$ (or $Y = A\Psi(U)$)
- The estimated model straight from the data is given as $A^e = Y\Psi^\dagger$
- The accurate $Y^\star$ and $\Psi^\star$ that generate the actual model: $A = Y^\star {\Psi^\star}^\dagger$
- Define $\delta Y = Y^\star - Y$ and $\delta\Psi = \Psi^\star - \Psi$ and assume

$$\frac{\|\delta Y\|}{\|Y^\star\|} \leq r_Y, \quad \frac{\|\delta \Psi\|}{\|\Psi^\star\|} \leq r_\Psi < 1.$$

- **Goal**:
  bound $\|\delta A\| = \|A - A^e\|$ originated from the non-zero δY and $\delta\Psi$

Bounds of Modeling Errors

Data are most likely noisy with presence of measurement errors, latency, and sometimes only pseudo-measurements are available

- Focus on the analysis on $Y = A\Psi$ (or $Y = A\Psi(U)$)
- The estimated model straight from the data is given as $A^e = Y\Psi^\dagger$
- The accurate $Y^\star$ and $\Psi^\star$ that generate the actual model: $A = Y^\star \Psi^{\star\dagger}$
- Define $\delta Y = Y^\star - Y$ and $\delta\Psi = \Psi^\star - \Psi$ and assume

$$\frac{\|\delta Y\|}{\|Y^\star\|} \leq r_Y, \quad \frac{\|\delta\Psi\|}{\|\Psi^\star\|} \leq r_\Psi < 1.$$

- **Goal**:
  bound $\|\delta A\| = \|A - A^e\|$ *originated from the non-zero* δY *and* $\delta\Psi$
- The key is essentially characterizing the sensitivity of the pseudo-inverse of $\Psi^\star$ with respect to the perturbation $\delta\Psi$

Pre-Processing the Data Matrices

Theorem: Bound of the Estimation Error

If the assumptions hold, then $\frac{\|\delta A\|}{\|A\|} \leq c_\Psi \frac{r_Y + r_\Psi}{1 - r_\Psi}$.

- $c_\Psi = \|\Psi\| \left\|\Psi^\dagger\right\|$ is known as the condition number of Ψ
- Probably no analytical bound that does not involve c_Ψ
- The value of c_Ψ is determined by the data and there is no much control over it

Pre-Processing the Data Matrices

Theorem: Bound of the Estimation Error

If the assumptions hold, then $\frac{\|\delta A\|}{\|A\|} \leq c_\Psi \frac{r_Y + r_\Psi}{1 - r_\Psi}$.

- $c_\Psi = \|\Psi\| \left\|\Psi^\dagger\right\|$ is known as the condition number of Ψ
- Probably no analytical bound that does not involve c_Ψ
- The value of c_Ψ is determined by the data and there is no much control over it

Several directions of tightening the bound:

1. The concept of effective condition number may help, but the concept is only used for positive definite matrices [F. Chan and D. E. Foulser 1988], [Z.-C. Li *et al.* 2007]
2. Choose partial data points while retaining the full row rank of Ψ
3. Diagonal scaling of the data matrices

Pre-Processing - Selection of the Data Points

- Only choose certain columns (data points) of Ψ, indexed by τ and denoted by Ψ_τ

Theorem: Bougain-Tzafriri

Suppose matrix Ψ is standardized. Then there is a set τ of column indices for which

$$|\tau| \geq c \cdot \frac{\|\Psi\|_F}{\|\Psi\|}$$

such that Ψ_τ has the condition number less than or equal to $\sqrt{3}$

Pre-Processing - Selection of the Data Points

- Only choose certain columns (data points) of Ψ, indexed by τ and denoted by Ψ_τ

Theorem: Bougain-Tzafriri

Suppose matrix Ψ is standardized. Then there is a set τ of column indices for which

$$|\tau| \geq c \cdot \frac{\|\Psi\|_F}{\|\Psi\|}$$

such that Ψ_τ has the condition number less than or equal to $\sqrt{3}$

- $\sqrt{3}$ is an impressively tight bound given that condition numbers can easily go over hundreds. Recall $\frac{\|\delta A\|}{\|A\|} \leq c_\Psi \frac{r_Y + r_\Psi}{1 - r_\Psi}$

Pre-Processing - Selection of the Data Points

- Only choose certain columns (data points) of Ψ, indexed by τ and denoted by Ψ_τ

Theorem: Bougain-Tzafriri

Suppose matrix Ψ is standardized. Then there is a set τ of column indices for which

$$|\tau| \geq c \cdot \frac{\|\Psi\|_F}{\|\Psi\|}$$

such that Ψ_τ has the condition number less than or equal to $\sqrt{3}$

- $\sqrt{3}$ is an impressively tight bound given that condition numbers can easily go over hundreds. Recall $\frac{\|\delta A\|}{\|A\|} \leq c_\Psi \frac{r_Y + r_\Psi}{1 - r_\Psi}$
- **Catch**: the theorem accounts the option of non-full row rank selection of columns, or vertical matrices
- Algorithmization of the theorem is available [J. A. Tropp 2009]

Pre-Processing - Diagonal Scaling

The goal is finding diagonal matrices, D_L and D_R, such that the condition number of $\widehat{\Psi} := D_L \Psi D_R$ is smaller than Ψ.

Pre-Processing - Diagonal Scaling

The goal is finding diagonal matrices, D_L and D_R, such that the condition number of $\widehat{\Psi} := D_L \Psi D_R$ is smaller than Ψ.

- The structure of the linear equality remains unchanged

$$Y = A\Psi \quad \Longrightarrow YD_R = AD_L^{-1}\left(D_L \Psi D_R\right) \quad \Longrightarrow \widehat{Y} = \widehat{A}\widehat{\Psi},$$

where $\widehat{Y} = YD_R$ and $\widehat{A} = AD_L^{-1}$.

Pre-Processing - Diagonal Scaling

The goal is finding diagonal matrices, D_L and D_R, such that the condition number of $\widehat{\Psi} := D_L \Psi D_R$ is smaller than Ψ.

- The structure of the linear equality remains unchanged

$$Y = A\Psi \quad \Longrightarrow YD_R = AD_L^{-1}\left(D_L \Psi D_R\right) \quad \Longrightarrow \widehat{Y} = \widehat{A}\widehat{\Psi},$$

where $\widehat{Y} = YD_R$ and $\widehat{A} = AD_L^{-1}$.

- The bound of the error term $\widehat{\delta A}$ relative to $\widehat{A}$ is tighter than the original one in the sense that the condition number for $\widehat{\Psi}$ is smaller than Ψ

Original: $\dfrac{\|\delta A\|}{\|A\|} \leq c_\Psi \dfrac{r_Y + r_\Psi}{1 - r_\Psi}$ **Diag. scaling:** $\dfrac{\left\|\widehat{\delta A}\right\|}{\left\|\widehat{A}\right\|} \leq \widehat{c_\Psi} \dfrac{\widehat{r_Y} + \widehat{r_\Psi}}{1 - \widehat{r_\Psi}}$

Diagonal Scaling

- No analytical conclusion on actual reduction of the modeling error
- Diagonal scaling is non-convex in general. Some heuristics are available [A. M. Bradley 2010], [R. Takapoui and H. Javad 2016]
- The condition numbers are reduced by a factor of 10 in an example of a switched linear system with 5 modes

	without pre-processing	with pre-processing
Mode 1	199.1373	21.0689
Mode 2	136.7279	16.3103
Mode 3	160.5263	18.2697
Mode 4	173.2082	18.6434
Mode 5	170.2047	20.3172

Table: The condition number of a data matrix w/wo the diagonal scaling.

Diagonal Scaling

- The reduced condition number results in tighter upper bounds
- The actual modeling errors are also reduced with diagonal scaling

	without pre-processing	with pre-processing
Mode 1	4.0230	0.4256
Mode 2	2.7622	0.3295
Mode 3	3.2430	0.3691
Mode 4	3.4992	0.3766
Mode 5	3.4385	0.4104

Table: The **upper bounds of** $\frac{\|\delta A\|}{\|A\|}$ w/wo the diagonal scaling.

	without pre-processing	with pre-processing
Mode 1	0.0136	0.0115
Mode 2	0.0115	0.0095
Mode 3	0.0130	0.0125
Mode 4	0.0165	0.0155
Mode 5	0.0129	0.0125

Table: The **value of** $\frac{\|\delta A\|}{\|A\|}$ w/wo the diagonal scaling.

Robust Controller Design

- The model is written as $A = Y\Psi(U)^{\dagger}$ with δA characterized

Robust Controller Design

- The model is written as $A = Y\Psi(U)^\dagger$ with δA characterized
- For nonlinear systems, $x(k+1) = A\Psi(x(k), u(k))$, a common way for controller designs is the linearization given as

$$x(k+1) = A_0 x(k) + B_0 u(k) + f_0(x(k), u(k)),$$
$$\|f_0(x(k), u(k))\| \leq [x(k)^\top, u(k)^\top]^\top P_0 [x(k), u(k)],$$

- With A, bounds of δA and Ψ known, one can find good candidates of A_0, B_0 and f_0 for robust controller design for the nonlinear system

Robust Controller Design

- The model is written as $A = Y\Psi(U)^\dagger$ with δA characterized
- For nonlinear systems, $x(k+1) = A\Psi(x(k), u(k))$, a common way for controller designs is the linearization given as

$$x(k+1) = A_0 x(k) + B_0 u(k) + f_0(x(k), u(k)),$$
$$\|f_0(x(k), u(k))\| \leq [x(k)^\top, u(k)^\top]^\top P_0 [x(k), u(k)],$$

- With A, bounds of δA and Ψ known, one can find good candidates of A_0, B_0 and f_0 for robust controller design for the nonlinear system
- We will showcase the results with a robust state feedback controller for the following switched linear system:

$$x(k+1) = A_{\sigma(k)} x(k) + B_{\sigma(k)} u(k),$$
$$u(k) = K_{\sigma(k)} x(k),$$
$$\sigma(k) = f(x(k))$$

Robust Controller for Switched Linear Systems

- A set of control gains $K_i,\ i \in \Gamma$ satisfying the following common Lyapunov conditions guarantees the stability of switched linear system under random switching

$$\exists P \succeq 0 \quad \text{s.t.} \quad (A_i + B_i K_i) P (A_i + B_i K_i)^\top \preceq P,\ \forall i \in \Gamma.$$

Robust Controller for Switched Linear Systems

- A set of control gains $K_i,\ i \in \Gamma$ satisfying the following common Lyapunov conditions guarantees the stability of switched linear system under random switching

$$\exists P \succeq 0 \quad \text{s.t.} \quad (A_i + B_i K_i) P (A_i + B_i K_i)^\top \preceq P,\ \forall i \in \Gamma.$$

- Similar to [C. De Persis and P. Tesi 2019], for each $i \in \Gamma$, define

$$\begin{bmatrix} K_i \\ I \end{bmatrix} = \begin{bmatrix} U_{i,0} \\ X_{i,0} \end{bmatrix} G_i,$$

Robust Controller for Switched Linear Systems

- A set of control gains $K_i,\ i \in \Gamma$ satisfying the following common Lyapunov conditions guarantees the stability of switched linear system under random switching

$$\exists P \succeq 0 \quad \text{s.t.} \quad (A_i + B_i K_i) P (A_i + B_i K_i)^\top \preceq P,\ \forall i \in \Gamma.$$

- Similar to [C. De Persis and P. Tesi 2019], for each $i \in \Gamma$, define

$$\begin{bmatrix} K_i \\ I \end{bmatrix} = \begin{bmatrix} U_{i,0} \\ X_{i,0} \end{bmatrix} G_i,$$

- Leading to a data representation of $A_i + B_i K_i$

$$\begin{aligned} A_i + B_i K_i &= [B_i \ \ A_i] \begin{bmatrix} K_i \\ I \end{bmatrix} = [B_i \ \ A_i] \begin{bmatrix} U_{i,0} \\ X_{i,0} \end{bmatrix} G_i \\ &= \left([B_i^e \ \ A_i^e] + \delta [B_i \ \ A_i] \right) \begin{bmatrix} U_{i,0} \\ X_{i,0} \end{bmatrix} G_i \\ &= \left(X_1 \left(I - \sum_{j \in \Gamma, j \neq i} \begin{bmatrix} U_{j,0} \\ X_{j,0} \end{bmatrix}^\dagger \begin{bmatrix} U_{j,0} \\ X_{j,0} \end{bmatrix} \right) + \delta [B_i \ \ A_i] \begin{bmatrix} U_{i,0} \\ X_{i,0} \end{bmatrix} \right) G_i. \end{aligned}$$

Robust Controller for Switched Linear Systems

$$A_i + B_i K_i = \Bigg(\underbrace{X_1 \Big(I - \sum_{j \in \Gamma, j \neq i} \begin{bmatrix} U_{j,0} \\ X_{j,0} \end{bmatrix}^{\dagger} \begin{bmatrix} U_{j,0} \\ X_{j,0} \end{bmatrix} \Big)}_{\text{the estimated system model}} + \underbrace{\delta [B_i \ \ A_i] \begin{bmatrix} U_{i,0} \\ X_{i,0} \end{bmatrix}}_{\text{the modeling error}} \Bigg) G_i.$$

- The estimated models are straight from the data; we can bound the second term by $\frac{\|\delta[B_i \ \ A_i]\|}{\|[B_i \ \ A_i]\|} \leq c_{\Psi} \frac{r_Y + r_{\Psi}}{1 - r_{\Psi}}$
- Some standard procedures (Schur complement, S-procedure, etc) are applied so that linear matrix inequalities (LMIs) conditions for stabilizing K_i, $i \in \Gamma$, are established

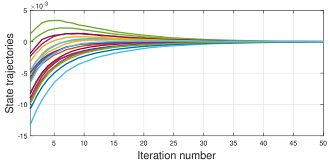


Figure: Trajectories of the system under the data-driven robust controller.

Summary

Conclusions

- Direct data representations of system modeling
- Insights on how noisy data propagate to inaccurate system modeling
- Some pre-processing methods are covered
- Robust controller design

Future Work

- Enhance the robustness and resiliency of the *real-time controllers* built around the data representations of system modeling
- Addressing the issue of the complexity involved in the controller design. *Reinforcement learning* may be justified in some applications such that controller design involves computationally intractable problems.