City University of Hong Kong,
Nanjing University

Learning Based Intelligent Modeling of Distributed Parameter Systems

Reporter: Zhi Wang,
Supervisor: Han-Xiong Li,
Collaborator: Chunlin Chen

September 28, 2018

Self-introduction

王志：

- 2011-2015，南京大学工程管理学院自动化系
  - 工程学学士
  - 导师：陈春林
- 2015-至今，香港城市大学系统工程与工程管理系
  - 攻读博士学位
  - 导师：李涵雄
- 研究方向：机器学习及其应用，智能系统建模

Content

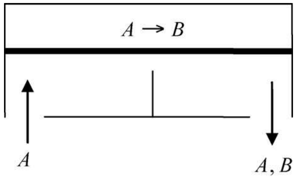
Distributed parameter systems (DPSs)

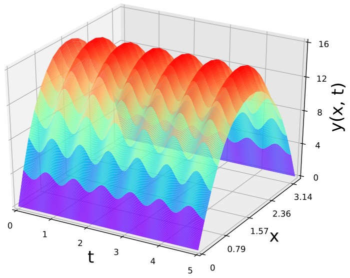
- A common kind of industrial processes where the input and output may vary in both time and space dimension
- Described by partial differential equations (PDEs):

$$\frac{\partial y(x,t)}{\partial t} = \mathcal{L}\left(y, \frac{\partial y}{\partial x}, \frac{\partial^2 y}{\partial x^2}, ..., \frac{\partial^{n_0} y}{\partial x^{n_0}}\right) + \bar{B}(x)u(t)$$

▶ A benchmark example: Catalytic rod system

$$\frac{\partial y(x,t)}{\partial t} = \frac{\partial^2 y(x,t)}{\partial x^2} + \beta_T\left(e^{-\frac{\gamma}{1+y}} - e^{-\gamma}\right) + \beta_u\left(b^T(x)u(t) - y(x,t)\right)$$





- Motivated by Fourier series:

$$y(x,t) = \sum_{i=1}^{\infty} \varphi_i(x) a_i(t)$$

- Reduced-order system, for practical use:

$$y_n(x,t) = \sum_{i=1}^{n} \varphi_i(x) a_i(t),$$

Modeling perspective

Approximating the original system

Control perspective

Developing control laws

Time-space separation
Karhunen-Loève decomposition (KLD)

Deriving the reduced-order basic functions (BFs) $\{\varphi_i(x)\}_{i=1}^{n}$:

- $\min_{\varphi_i(x)} \langle ||y(x,t) - y_n(x,t)||^2 \rangle$, subject to $(\varphi_i, \varphi_i) = 1$
- $J = \langle ||y(x,t) - y_n(x,t)||^2 \rangle + \sum_{i=1}^{n} \lambda_i((\varphi_i, \varphi_i) - 1)$
- ...
- Energy function $E_i = \frac{\lambda_i}{\sum_{j=1}^{K} \lambda_j}$

Determine the degree *n*

Capturing more than 99% of the system's *energy*

- Background:
  - $y_n(x,t) = \sum_{i=1}^{n} \varphi_i(x) a_i(t)$
  - Learning $\{\varphi_i(x)\}_{i=1}^{n}$ using KLD
- Time series data in the low-dimensional space: $a_i(t)$
- Identify the reduced-order temporal model: $F(a)$

Nonlinear autoregressive exogenous (NARX) model

$$a(t) = F(a(t-1), ..., a(t-d_a), u(t-1), ..., u(t-d_u)) + e(t)$$

Learning $\rightleftharpoons$ Modeling

The temporal model F can be approximated by:

- Polynomial functions
- Radial basis functions
- Neural networks
- Extreme learning machines
- Support vector machines
- ...

Purposes

Predicting future outputs, tracking the system's dynamics

Recall the modeling process: $y_n(x,t) = \sum_{i=1}^{n} \varphi_i(x) a_i(t)$

- The quality of $\varphi_i(x)$ is dependent on the precise information measured on the sampling spatial domain

A limited number of sensors in practice

- Expensive initial and maintenance costs

Problem Formulation

Choose the most informative m locations from the candidate n ones ($m < n$).

Combinatorial complexity: $\binom{n}{m} = \frac{n!}{m!(n-m)!}$

Optimal sensor placement
Existing methods

POD-based methods:

- Heuristically arranged at the extrema of the POD modes
- Sensitive to experimental settings; no performance guarantee

Convec optimization:

- Relaxing the constraints $\{0, 1\}^n$ to the convex set $[0, 1]^n$
- Model-based, requiring convexity

Greedy methods:

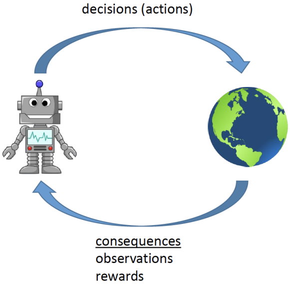
- Using a series of optimal local steps instead of global ones
- Guarantee on local optimality, requiring submodularity

Genetic algorithms:

- Valid alternatives to reduce brute-force search
- An optimization perspective

Expected properties:

- Model-free
- Computationally efficient
- A performance guarantee



Reinforcement Learning

How an autonomous active agent learns the optimal policies while interacting with an initially unknown environment

An RL perspective

Sensor placement configuration

Objective function

$$\text{maximize}_{\boldsymbol{P}_m} \quad \mathcal{F}(\boldsymbol{P}_m) = \log\det(\boldsymbol{\Phi}^T \boldsymbol{P}_m^T \boldsymbol{P}_m \boldsymbol{\Phi})$$

Formulation of MDP	
state s	$\boldsymbol{P}_m$
action a	$\boldsymbol{\Psi}$
next state s'	$\boldsymbol{P}'_m = \boldsymbol{P}_m\boldsymbol{\Psi}$
reward $r(s,a)$	$\mathcal{F}(\boldsymbol{P}'_m)$
state value function	$v_\pi(s)$

Temporal difference learning

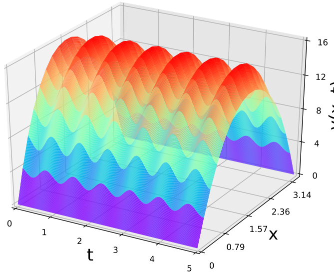
$$v(s) \leftarrow v(s) + \alpha(r_{t+1} + \gamma v(s_{t+1}) - v(s_t))$$

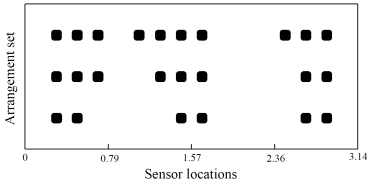
An RL perspective
Advantages

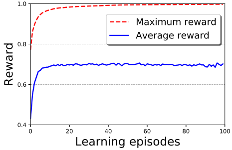
Main advantages of RL-based sensor placement for modeling:

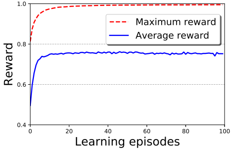
- Guarantee on convergence to the global optimum
- Model-free, compared to analytic methods
- Efficient for NP-hard problems by trade-off between exploration and exploitation
- Implemented in an online, incremental way
- Potential for high-dimensional problems (DRL)

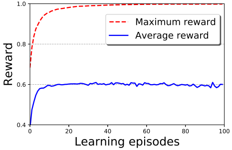
RL-based sensor placement
Benchmark: Catalytic rod





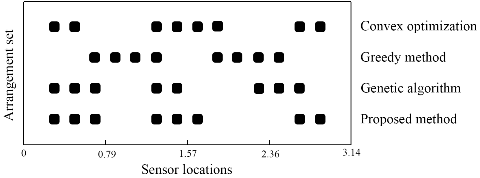






RL-based sensor placement

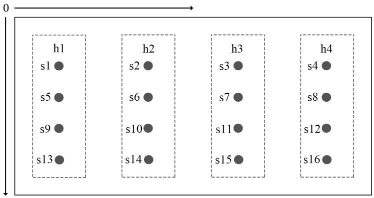
Benchmark: Catalytic rod

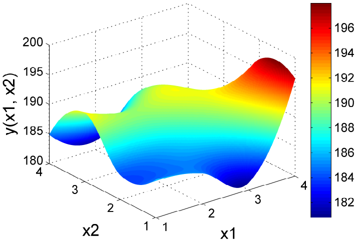


	Sensors	Convex	Greedy	GA	RL
RMSE$(c, \hat{c})$	m=6	3.51E-05	8.22E-05	5.96E-05	**3.24E-05**
	m=8	3.16E-05	6.68E-05	4.37E-05	**3.15E-05**
	m=10	2.95E-05	4.88E-05	3.30E-05	**2.52E-05**
RMSE$(y, \hat{y})$	m=6	4.40E-04	4.84E-04	4.86E-04	**4.12E-04**
	m=8	4.28E-04	4.68E-04	4.74E-04	**4.18E-04**
	m=10	4.15E-04	4.26E-04	4.12E-04	**4.09E-04**

RL-based sensor placement

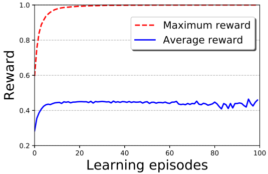
Real snap curing oven system





RL-based sensor placement

Real snap curing oven system



Static level / Dynamic level (E-03)				
Sensors	Convex	Greedy	GA	RL
m = 4	**3.3** / **13.2**	11.3 / 64.5	3.5 / 47.5	**3.3** / **13.2**
m = 6	**3.0** / **8.6**	10.1 / 38.5	5.0 / 40.2	**3.0** / **8.6**
m = 8	2.7 / 7.4	8.7 / 27.4	2.4 / 38.5	**2.2** / **6.4**
m = 10	2.3 / 5.8	3.0 / 20.0	2.8 / 33.0	**2.2** / **4.3**
m = 12	**0.8** / **2.8**	2.3 / 19.0	1.3 / 31.9	**0.8** / **2.8**

Zhi Wang, Han-Xiong Li, and Chunlin Chen, “Reinforcement learning based optimal sensor placement for distributed parameter systems", resubmitted to IEEE Transactions on Cybernetics.

Recall traditional modeling of DPSs:

- $y_n(x,t) = \sum_{i=1}^{n} \varphi_i(x) a_i(t), x \in [x_1, ..., x_N], t \in [1, ..., L]$
- Deriving BFs using KLD
- Identify the temporal model

Limitations:

- Offline implementations only
- Time-space synthesis is computed only once and remains fixed

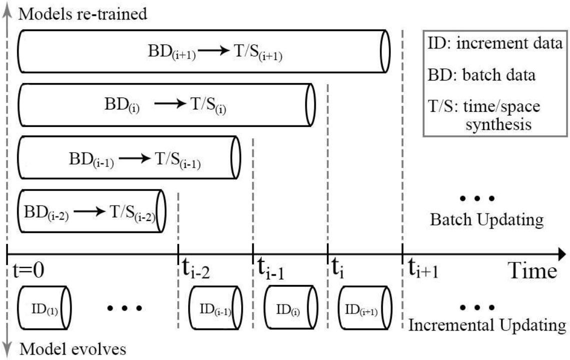
Canonical method for the online environment:

- Re-trained from scratch repeatedly when the new data comes
- Calculating the basis functions with L time steps of N spatial measurements requires $O(NL)$ memory units and $O(L^3)$ flops
- Time-consuming, a great storage burden

$$C\gamma_i = \lambda_i \gamma_i$$

$$C_{tk} = \frac{1}{L}\int_{\Omega} y(\zeta, t)y(\zeta, k)d\zeta$$

$C : L \times L$, L: increasing time steps



Incremental learning

Update the time-space synthesis in an incremental way when the new data comes

Original basis functions Φ:

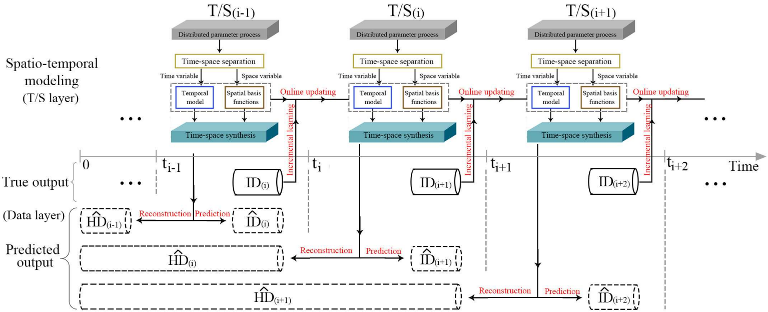
- $C = \frac{1}{L} Y_1^T Y_1$
- $Y_1^T = U \Sigma V^T$
- $C = \frac{1}{L} U \Sigma V^T V \Sigma U^T = U \Lambda U^T$
- $C_n = U_n \Lambda_n U_n^T$
- $\Phi = (U_n^T Y_1^T)^T = Y_1 U_n$

Updating the basis functions:

- $Y=[Y_1,Y_2]$, $\quad \bar{C}=\frac{1}{L+M}Y^TY=\frac{1}{L+M}\begin{bmatrix} Y_1^TY_1 & Y_1^TY_2 \\ Y_2^TY_1 & Y_2^TY_2 \end{bmatrix}$
- $(I-V_nV_n^T)Y_2=QR$
- $Y^T=\begin{bmatrix} (Y_1^T)_{L\times N} \\ (Y_2^T)_{M\times N} \end{bmatrix}=\begin{bmatrix} U_n & 0 \\ 0 & I_M \end{bmatrix}\begin{bmatrix} \Sigma_n & 0 \\ Y_2^TV_n & R^T \end{bmatrix}[V_n \quad Q]^T$
- $\begin{bmatrix} \Sigma_n & 0 \\ Y_2^TV_n & R^T \end{bmatrix}=\widetilde{U}\widetilde{\Sigma}\widetilde{V}^T$
- $\bar{C}=(\begin{bmatrix} U_n & 0 \\ 0 & I_M \end{bmatrix}\widetilde{U})(\frac{1}{L+M}\widetilde{\Sigma}\widetilde{\Sigma}^T)(\begin{bmatrix} U_n & 0 \\ 0 & I_M \end{bmatrix}\widetilde{U})^T=\bar{U}\bar{\Lambda}\bar{U}^T$
- $\bar{C}_{n'}=\bar{U}_{n'}\bar{\Lambda}_{n'}\bar{U}_{n'}^T$

$$\boxed{\bar{\Phi}=[\Phi A \quad Y_2](\begin{bmatrix} U_n & 0 \\ 0 & I_M \end{bmatrix}\widetilde{U}_{n'})}$$

Incremental modeling
Method



Total complexity

$O(NM^2)$, depending on the length M of the new data increment

Main operations:

- QR decomposition of $[(I - V_n V_n^T) Y_2]_{N \times M}$: $O(NM^2)$ flops
- SVD of $\begin{bmatrix} \Sigma_n & 0 \\ Y_2^T V_n & R^T \end{bmatrix}_{(n+M)\times(n+m)}$: $O((n+m)(n+M)^2)$ flops
- $n \ll \{m, N, M\}$, and $m \leq min\{N, M\}$

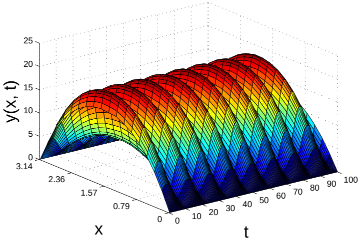
Complexity of canonical method:

- The new correlation matrix $\bar{C}_{(L+M)\times(L+M)}$
- $O((L+M)^3)$, $\{N, M\} \ll L$

- Online computation and database update
- Reduced complexity and memory requirements
- Track and adapt to the system's dynamics in real-time

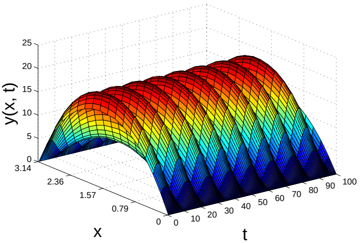
Benchmark: Catalytic rod

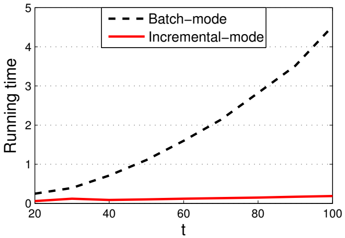
Modeling accuracy



Without updating / Batch updating / Incremental updating		
Time	RMSE training	RMSE testing
t = 20	0.0788 / 0.0433 / 0.0434	0.0844 / 0.0550 / 0.0549
t = 30	0.0807 / 0.0434 / 0.0435	0.0834 / 0.0555 / 0.0553
t = 40	0.0814 / 0.0435 / 0.0436	0.0834 / 0.0552 / 0.0551
t = 50	0.0818 / 0.0435 / 0.0436	0.0818 / 0.0552 / 0.0549
t = 60	0.0818 / 0.0435 / 0.0436	0.0850 / 0.0551 / 0.0550
t = 70	0.0823 / 0.0435 / 0.0437	0.0841 / 0.0555 / 0.0557
t = 80	0.0825 / 0.0435 / 0.0437	0.0832 / 0.0559 / 0.0555
t = 90	0.0826 / 0.0435 / 0.0437	0.0835 / 0.0556 / 0.0554
t = 100	0.0827 / 0.0435 / 0.0437	0.0832 / 0.0553 / 0.0549

Benchmark: Catalytic rod
Running time





Zhi Wang, and Han-Xiong Li, “Incremental spatiotemporal learning for online modeling of distributed parameter systems," IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2018.

Recall traditional modeling of DPSs:

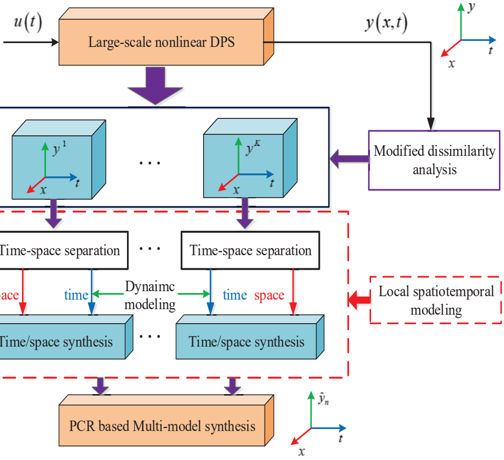
- $y_n(x,t) = \sum_{i=1}^{n} \varphi_i(x) a_i(t), x \in [x_1, ..., x_N], t \in [1, ..., L]$
- KLD relies on the Euclidean distance as the metric to minimize
- Assumption: the process data belongs to a linear space

Limitations:

- Fails to capture the nonlinear degrees of freedom in complex nonlinear systems

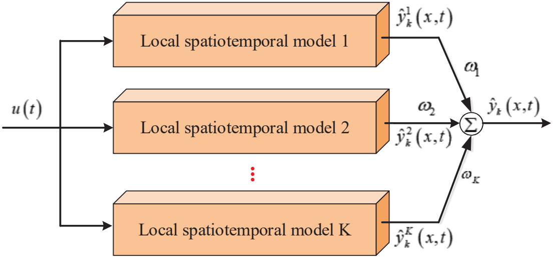
Multi-mode modeling

Motivation



- $Y_1, Y_2, R_i = \frac{1}{N_i} Y_i Y_i^T$
- $R = \frac{N_1}{N_1+N_2} R_1 + \frac{N_2}{N_1+N_2} R_2, \; P_0^T R P_0 = \Lambda$
- $Z_i = \sqrt{\frac{N_i}{N_1+N_2}} \Lambda^{-\frac{1}{2}} P_0^T Y_i = \sqrt{\frac{N_i}{N_1+N_2}} P^T Y_i$
- $S_i = \frac{1}{N_i} Z_i Z_i^T = \frac{N_i}{N_1+N_2} P^T \frac{Y_i Y_i^T}{N_i} P = \frac{N_i}{N_1+N_2} P^T R_i P$
- $S_1 + S_2 = I, \; S_i w_i^j = \lambda_i^j w_i^j$

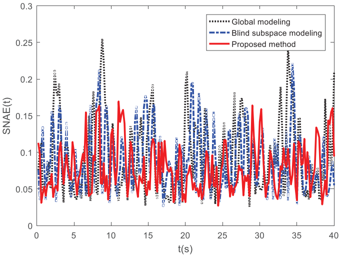
$$\boxed{Dis(Y_1, Y_2) = \frac{4}{n} \sum_{j=1}^{n} (\lambda_j - 0.5)^2}$$

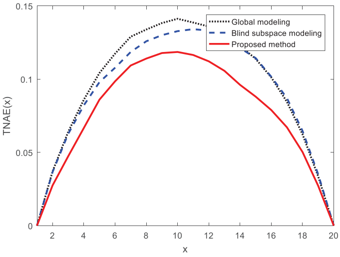


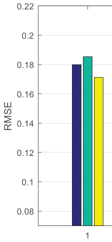
Principle component regression:

- $\widehat{y}_k(x_i,t) = w_{i,1}\widehat{y}_k^1(x_i,t) + w_{i,2}\widehat{y}_k^2(x_i,t) + \ldots + w_{i,K}\widehat{y}_k^K(x_i,t)$
- $W_i = (H_i^T H_i)^{-1} H_i^T Y_i, i = 1, \ldots, n$
- $\bar{H} = c_1 d_1^T + c_2 d_2^T + \ldots + c_K d_K^T,\ \bar{Y} = \bar{H}\bar{W} = CD^T\bar{W}$
- $\bar{W} = D\bar{W}_q = D(C^T C)^{-1} C^T \bar{Y}$

Benchmark: Catalytic rod
Modeling error

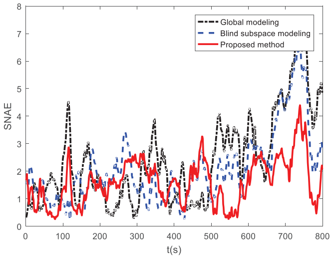


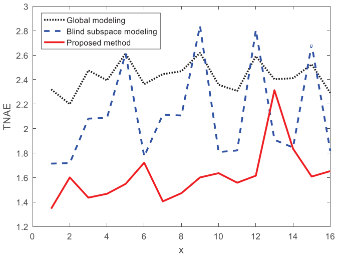




Real snap curing oven system

Modeling error





Zhi Wang, and Han-Xiong Li, “Modified Dissimilarity Analysis based spatiotemporal multi-modeling for complex distributed parameter processes," submitted to IEEE Transactions on Industrial Electronics.