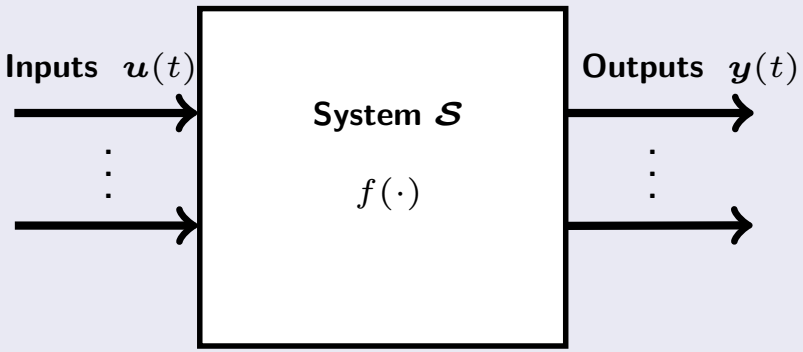
Lifting the curse of dimensionality in nonlinear system identification with tensor networks.

Kim Batselier, Zhongming Chen, Ching-Yun Ko, Ngai Wong

System identification



Main Problem

Given measured inputs $\{\boldsymbol{u}(t) \in \mathbb{R}^p\}_{t=1}^N$, outputs $\{\boldsymbol{y}(t) \in \mathbb{R}^l\}_{t=1}^N$, and a parametric input-output mapping $f(\cdot)$ for the system $\boldsymbol{\mathcal{S}}$, estimate the parameters of $f(\cdot)$.

NFIR black box model

- $\boldsymbol{y}(t) = f(\boldsymbol{u}(t), \boldsymbol{u}(t-1), \ldots, \boldsymbol{u}(t-M+1))$ with nonlinear function $f(\cdot)$

NFIR black box model

- $\boldsymbol{y}(t) = f(\boldsymbol{u}(t), \boldsymbol{u}(t-1), \ldots, \boldsymbol{u}(t-M+1))$ with nonlinear function $f(\cdot)$
- Taylor expansion of $f(\cdot)$: NFIR $\Rightarrow$ Volterra series

NFIR black box model

- $\boldsymbol{y}(t) = f(\boldsymbol{u}(t), \boldsymbol{u}(t-1), \ldots, \boldsymbol{u}(t-M+1))$ with nonlinear function $f(\cdot)$
- Taylor expansion of $f(\cdot)$: NFIR $\Rightarrow$ Volterra series
- SISO truncated Volterra series:

$$y(t) = h_0 + \sum_{i=1}^{d} \sum_{k_1,\ldots,k_i=0}^{M-1} h_i(k_1, \ldots, k_i) \prod_{j=1}^{i} u(t-k_j),$$

NFIR black box model

- $\boldsymbol{y}(t) = f(\boldsymbol{u}(t), \boldsymbol{u}(t-1), \ldots, \boldsymbol{u}(t-M+1))$ with nonlinear function $f(\cdot)$
- Taylor expansion of $f(\cdot)$: NFIR $\Rightarrow$ Volterra series
- SISO truncated Volterra series:

$$y(t) = h_0 + \sum_{i=1}^{d} \sum_{k_1,\ldots,k_i=0}^{M-1} h_i(k_1, \ldots, k_i) \prod_{j=1}^{i} u(t-k_j),$$

- MIMO truncated Volterra series: each of the l outputs in $\boldsymbol{y}(t) \in \mathbb{R}^l$ is a multivariate polynomial in $\boldsymbol{u}(t), \boldsymbol{u}(t-1), \ldots, \boldsymbol{u}(t-M+1)$.

Multivariate polynomials

- n-variate polynomial of total degree d
- Number of coefficients is $\binom{d+n}{n} \approx n^d$, e.g. $\binom{7+20}{20} = 888030$

Multivariate polynomials

- n-variate polynomial of total degree d
- Number of coefficients is $\binom{d+n}{n} \approx n^d$, e.g. $\binom{7+20}{20} = 888030$

Curse of dimensionality

Truncated Volterra series are described by an exponential amount of unknown coefficients $\Rightarrow$ need to estimate an exponential amount of coefficients.

Multivariate polynomials

- n-variate polynomial of total degree d
- Number of coefficients is $\binom{d+n}{n} \approx n^d$, e.g. $\binom{7+20}{20} = 888030$

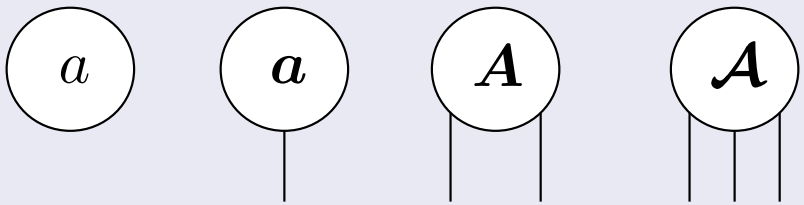
Curse of dimensionality

Truncated Volterra series are described by an exponential amount of unknown coefficients $\Rightarrow$ need to estimate an exponential amount of coefficients.

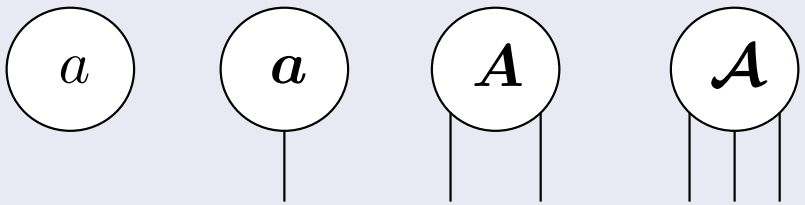
Main message of this talk

Lifting the curse of dimensionality in this system identification problem with tensor networks.

Tensor network building blocks



Tensor network building blocks



Three tensor network diagram rules

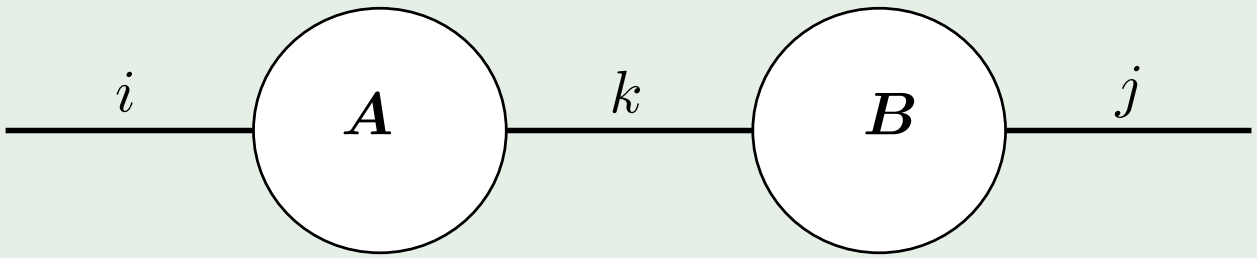
1. Nodes in the network are tensors
2. Each edge of each node is a particular index of the tensor
3. Connected edges refer to summation over that particular index

Tensor network example 1

$$\boldsymbol{C}(i,j) = \sum_k \boldsymbol{A}(i,k)\ \boldsymbol{B}(k,j)$$

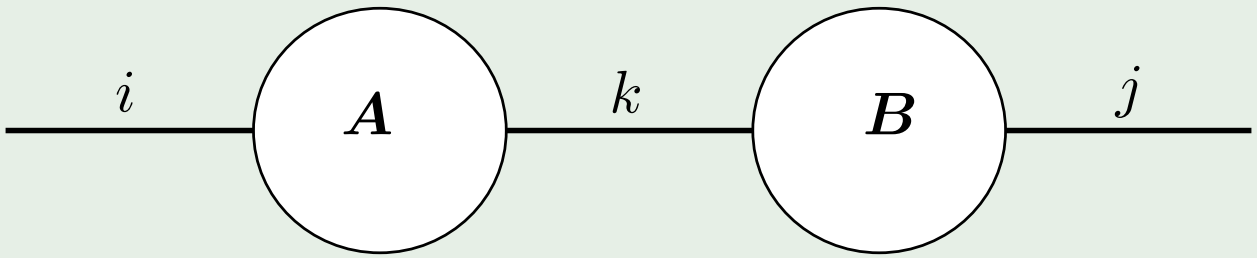
Tensor network example 1

$\boldsymbol{C}(i,j) = \sum_k \boldsymbol{A}(i,k)\, \boldsymbol{B}(k,j)$ is visually represented by

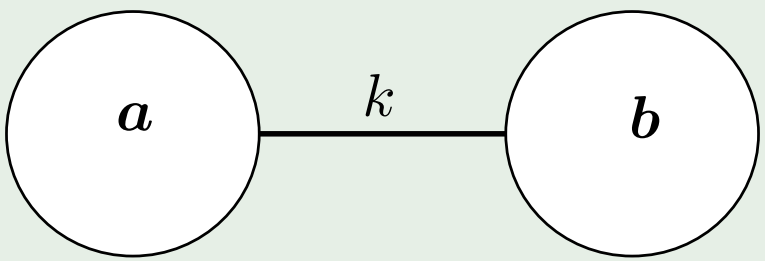


Tensor network example 1

$\boldsymbol{C}(i,j) = \sum_k \boldsymbol{A}(i,k)\, \boldsymbol{B}(k,j)$ is visually represented by

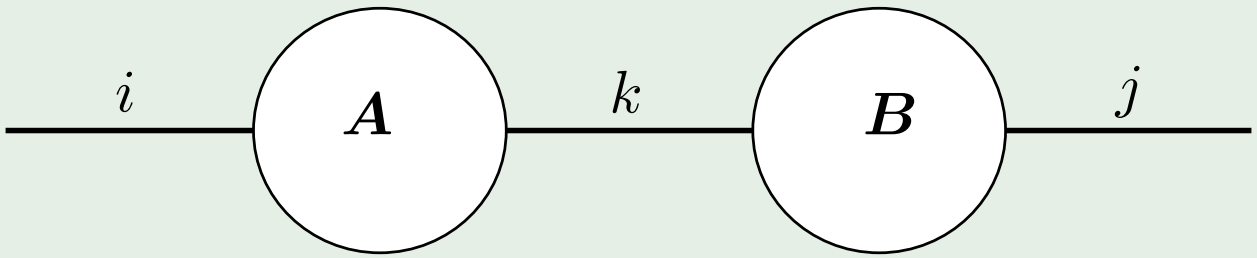


Tensor network example 2

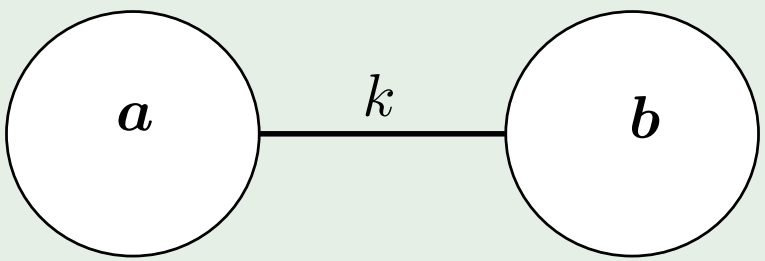


Tensor network example 1

$\boldsymbol{C}(i,j) = \sum_k \boldsymbol{A}(i,k)\ \boldsymbol{B}(k,j)$ is visually represented by

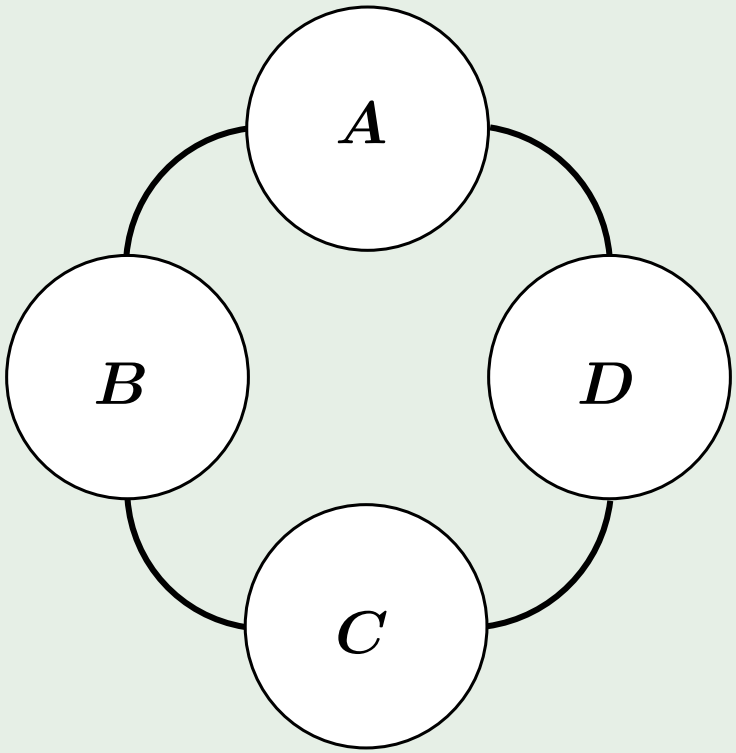


Tensor network example 2

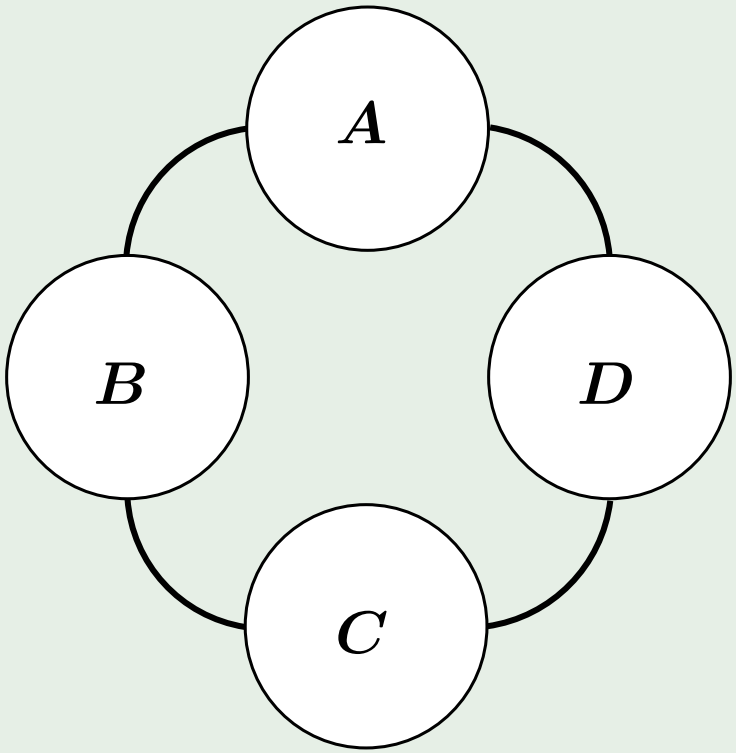


$\sum_k \boldsymbol{a}(k)\ \boldsymbol{b}(k) = \boldsymbol{a}^T\ \boldsymbol{b}$

Tensor network example 3



Tensor network example 3



$$\text{Trace}(\boldsymbol{ABCD}) = \text{Trace}(\boldsymbol{BCDA}) = \cdots = \text{Trace}(\boldsymbol{DABC})$$

Tensor network example 4

Singular value decomposition of a rank-r matrix $\boldsymbol{A} \in \mathbb{R}^{p \times q}$:

$$\boldsymbol{A} = \boldsymbol{U}\ \boldsymbol{S}\ \boldsymbol{V}^T$$

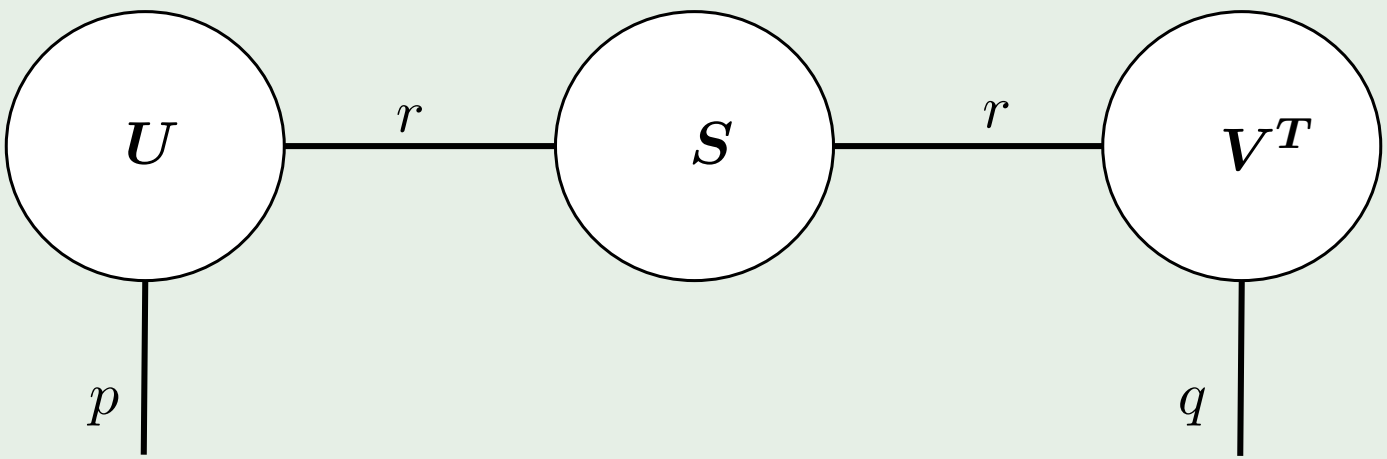
with $\boldsymbol{U} \in \mathbb{R}^{p \times r}, \boldsymbol{V} \in \mathbb{R}^{q \times r}$ orthogonal and $\boldsymbol{S} \in \mathbb{R}^{r \times r}$ diagonal.

Tensor network example 4

Singular value decomposition of a rank-r matrix $\boldsymbol{A} \in \mathbb{R}^{p \times q}$:

$$\boldsymbol{A} = \boldsymbol{U}\ \boldsymbol{S}\ \boldsymbol{V}^T$$

with $\boldsymbol{U} \in \mathbb{R}^{p \times r}, \boldsymbol{V} \in \mathbb{R}^{q \times r}$ orthogonal and $\boldsymbol{S} \in \mathbb{R}^{r \times r}$ diagonal.



Outer product

$$\boldsymbol{C} = \boldsymbol{a}\,\boldsymbol{b}^T = \boldsymbol{a} \circ \boldsymbol{b}$$

Outer product

$$\boldsymbol{C} = \boldsymbol{a}\,\boldsymbol{b}^T = \boldsymbol{a} \circ \boldsymbol{b}$$

$$\boldsymbol{C}(i,j) = \boldsymbol{a}(i)\,\boldsymbol{b}(j)$$

Outer product

$$\boldsymbol{C} = \boldsymbol{a}\,\boldsymbol{b}^T = \boldsymbol{a} \circ \boldsymbol{b}$$

$$\boldsymbol{C}(i,j) = \boldsymbol{a}(i)\,\boldsymbol{b}(j)$$

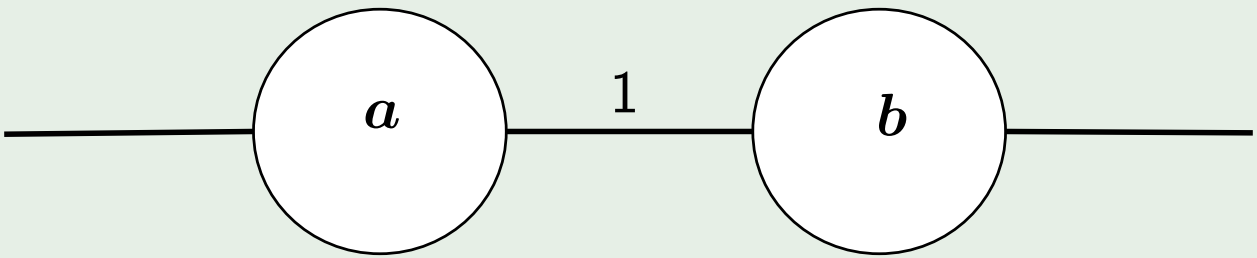
$$\boldsymbol{C}(i,j) = \sum_{k=1}^{1} \boldsymbol{a}(i,k)\,\boldsymbol{b}(k,j)$$

Outer product

$$\boldsymbol{C} = \boldsymbol{a}\,\boldsymbol{b}^T = \boldsymbol{a} \circ \boldsymbol{b}$$

$$\boldsymbol{C}(i,j) = \boldsymbol{a}(i)\,\boldsymbol{b}(j)$$

$$\boldsymbol{C}(i,j) = \sum_{k=1}^{1} \boldsymbol{a}(i,k)\,\boldsymbol{b}(k,j)$$

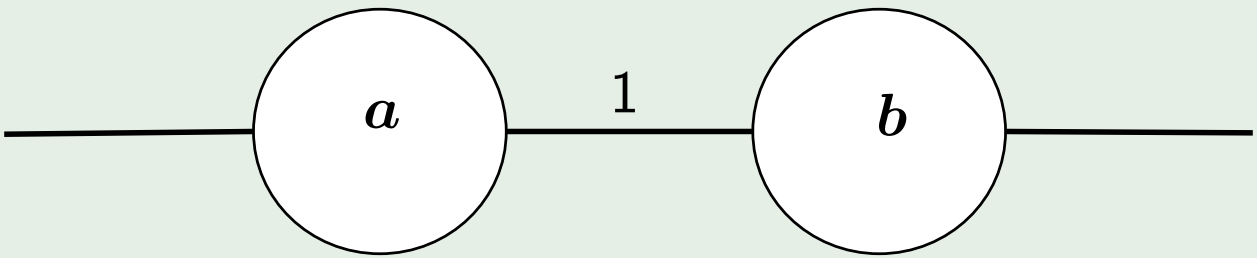


Outer product

$$\boldsymbol{C} = \boldsymbol{a}\,\boldsymbol{b}^T = \boldsymbol{a} \circ \boldsymbol{b}$$

$$\boldsymbol{C}(i,j) = \boldsymbol{a}(i)\,\boldsymbol{b}(j)$$

$$\boldsymbol{C}(i,j) = \sum_{k=1}^{1} \boldsymbol{a}(i,k)\,\boldsymbol{b}(k,j)$$



Important take-home message

Tensor networks with small interconnection dimensions represent tensors with interrelated entries.

MIMO Volterra model

Each of the l outputs in $\boldsymbol{y}(t) \in \mathbb{R}^l$ is a multivariate polynomial in $\boldsymbol{u}(t), \boldsymbol{u}(t-1), \ldots, \boldsymbol{u}(t-M+1)$.

MIMO Volterra model

Each of the l outputs in $\boldsymbol{y}(t) \in \mathbb{R}^l$ is a multivariate polynomial in $\boldsymbol{u}(t), \boldsymbol{u}(t-1), \ldots, \boldsymbol{u}(t-M+1)$.

Define input and output vectors

- $\boldsymbol{y}(t) := \begin{pmatrix} y_1(t) \\ \vdots \\ y_l(t) \end{pmatrix} \in \mathbb{R}^l$
- $\boldsymbol{u}_t := \begin{pmatrix} 1 & \boldsymbol{u}(t)^T & \cdots & \boldsymbol{u}(t-M+1)^T \end{pmatrix}^T \in \mathbb{R}^{(pM+1)}$
- All input monomials are contained in

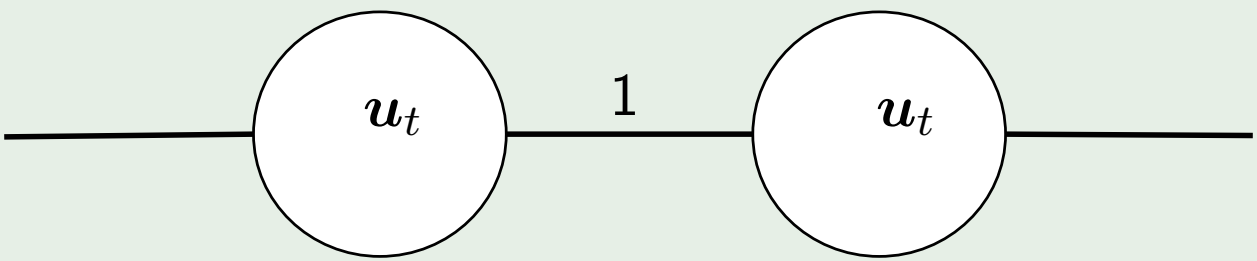
$$\overbrace{\boldsymbol{u}_t \circ \boldsymbol{u}_t \circ \cdots \circ \boldsymbol{u}_t}^{d} \in \mathbb{R}^{(pM+1)\times\cdots\times(pM+1)}$$

Small example: $p = 1$, $l = 1$, $M = 2$ and $d = 2$

$$\boldsymbol{u}_t = \begin{pmatrix} 1 \\ u(t) \\ u(t-1) \end{pmatrix}$$

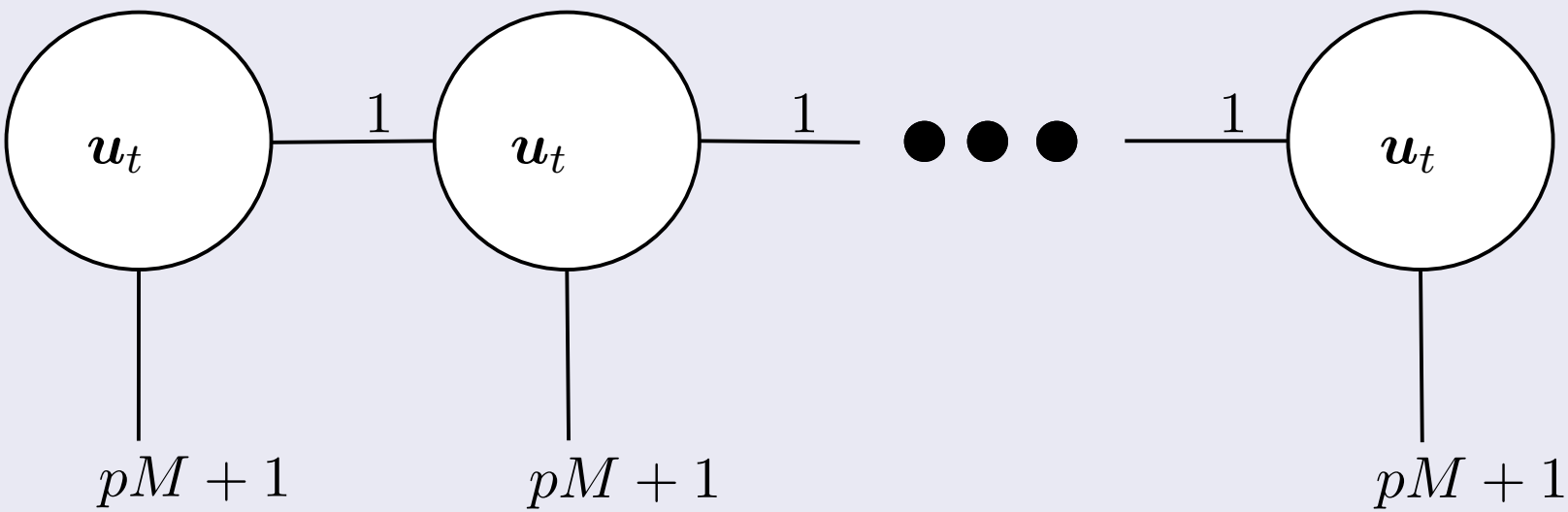
$$\boldsymbol{u}_t \circ \boldsymbol{u}_t = \begin{pmatrix} 1 \\ u(t) \\ u(t-1) \end{pmatrix} \begin{pmatrix} 1 & u(t) & u(t-1) \end{pmatrix}$$

$$= \begin{pmatrix} 1 & u(t) & u(t-1) \\ u(t) & u(t)^2 & u(t)\,u(t-1) \\ u(t-1) & u(t)\,u(t-1) & u(t-1)^2 \end{pmatrix}$$



Symmetric rank-1 input tensor

$$\overbrace{\boldsymbol{u}_t \circ \boldsymbol{u}_t \circ \cdots \circ \boldsymbol{u}_t}^{d} \in \mathbb{R}^{(pM+1)\times\cdots\times(pM+1)}$$



Small example: $p = 1$, $l = 1$, $M = 2$ and $d = 2$

$$y(t) = f(u(t), u(t-1))$$

$$= \boldsymbol{v}^T \operatorname{vec}(\boldsymbol{u}_t \circ \boldsymbol{u}_t)$$

$$= \begin{pmatrix} v_1 & v_2 & v_3 & v_4 & \cdots & v_9 \end{pmatrix} \begin{pmatrix} 1 \\ u(t) \\ u(t-1) \\ u(t) \\ \vdots \\ u(t-1)^2 \end{pmatrix}$$

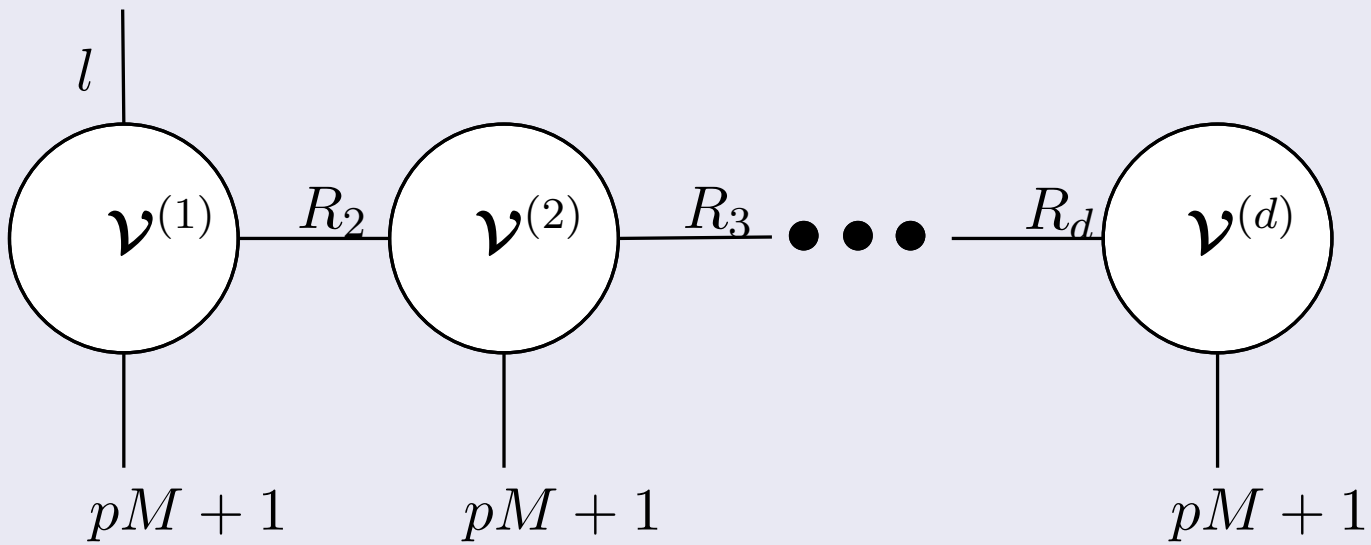
MIMO Volterra system

- $\boldsymbol{y}(t) = \boldsymbol{V}\,\mathrm{vec}\left(\boldsymbol{u}_t \circ \boldsymbol{u}_t \circ \cdots \circ \boldsymbol{u}_t\right)$
- $\boldsymbol{V} \in \mathbb{R}^{l \times (pM+1)^d}$ contains coefficients of l polynomials

MIMO Volterra system

- $\boldsymbol{y}(t) = \boldsymbol{V}\,\mathrm{vec}\left(\boldsymbol{u}_t \circ \boldsymbol{u}_t \circ \cdots \circ \boldsymbol{u}_t\right)$
- $\boldsymbol{V} \in \mathbb{R}^{l \times (pM+1)^d}$ contains coefficients of l polynomials

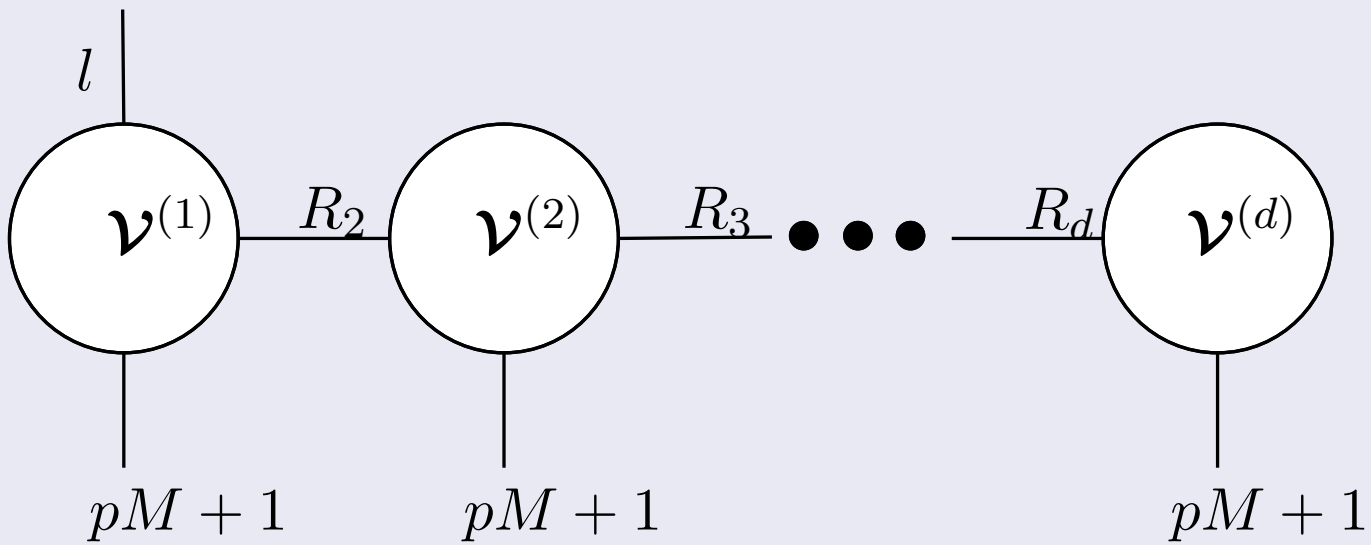
V matrix as a tensor network



MIMO Volterra system

- $\boldsymbol{y}(t) = \boldsymbol{V}\,\mathrm{vec}\left(\boldsymbol{u}_t \circ \boldsymbol{u}_t \circ \cdots \circ \boldsymbol{u}_t\right)$
- $\boldsymbol{V} \in \mathbb{R}^{l \times (pM+1)^d}$ contains coefficients of l polynomials

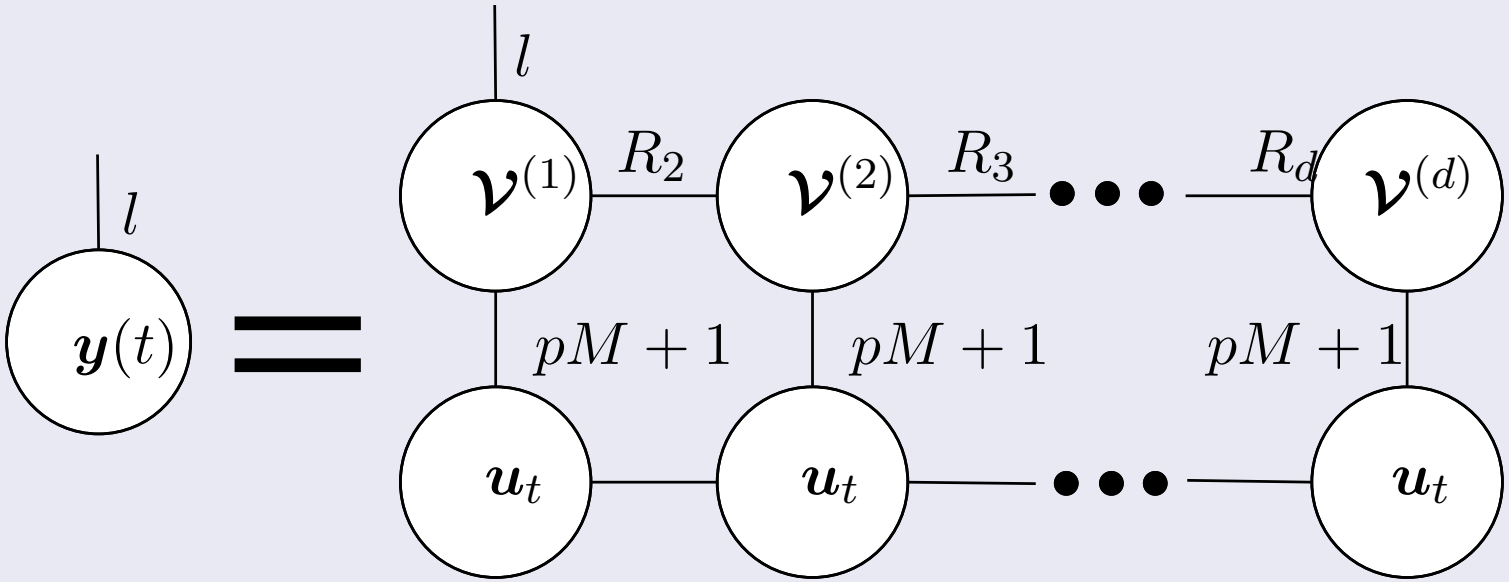
V matrix as a tensor network



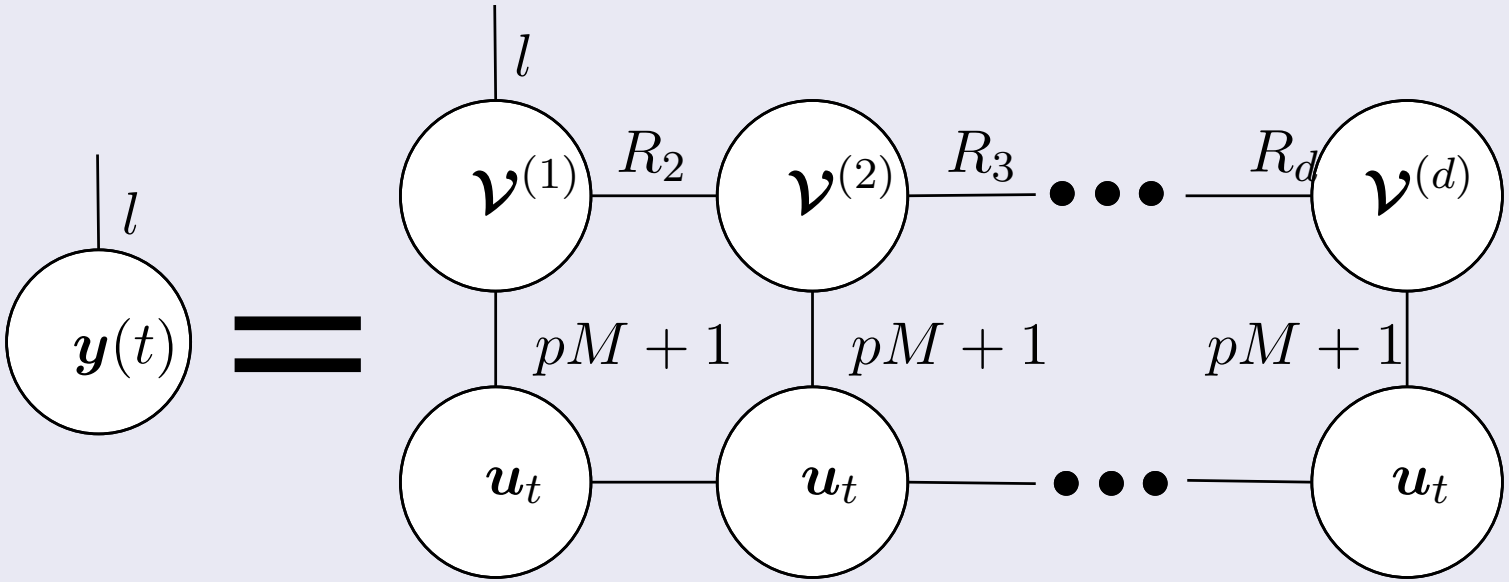
Lifting the curse of dimensionality

Storage of tensor network requires $\approx O(d(pM+1)R^2)$ elements

MIMO Volterra tensor network



MIMO Volterra tensor network



Tensor Network math

$$\sum_{r_2=1}^{R_2} \cdots \sum_{r_d=1}^{R_d} \sum_{i_1=1}^{pn_u+1} \cdots \sum_{i_d=1}^{pn_u+1} \sum_{j_2=1}^{1} \cdots \sum_{j_d=1}^{1} \boldsymbol{\nu}^{(1)}(l, i_1, r_2)\boldsymbol{\nu}^{(2)}(r_2, i_2, r_3)$$

$$\boldsymbol{\nu}^{(d-1)}(r_{d-1}, i_d, r_d)\, \boldsymbol{\nu}^{(d)}(r_d, i_d)\, \boldsymbol{u_t}(i_1, j_2)\, \boldsymbol{u_t}(j_2, i_2, j_3) \cdots \boldsymbol{u_t}(j_d, i_d).$$

MIMO Volterra system identification problem

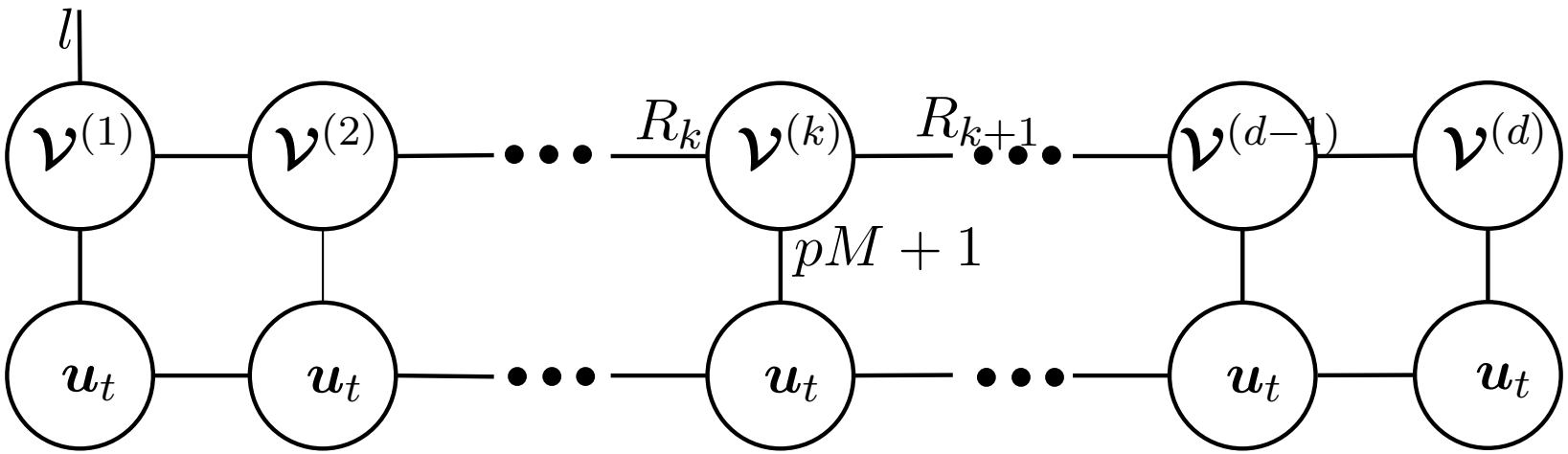
Given measured inputs $\{\boldsymbol{u}(t) \in \mathbb{R}^p\}_{t=1}^N$, outputs $\{\boldsymbol{y}(t) \in \mathbb{R}^l\}_{t=1}^N$, and an unknown MIMO Volterra model $\boldsymbol{V}$, estimate the tensor network tensors $\boldsymbol{\mathcal{V}}^{(1)}, \boldsymbol{\mathcal{V}}^{(2)}, \ldots, \boldsymbol{\mathcal{V}}^{(d)}$.

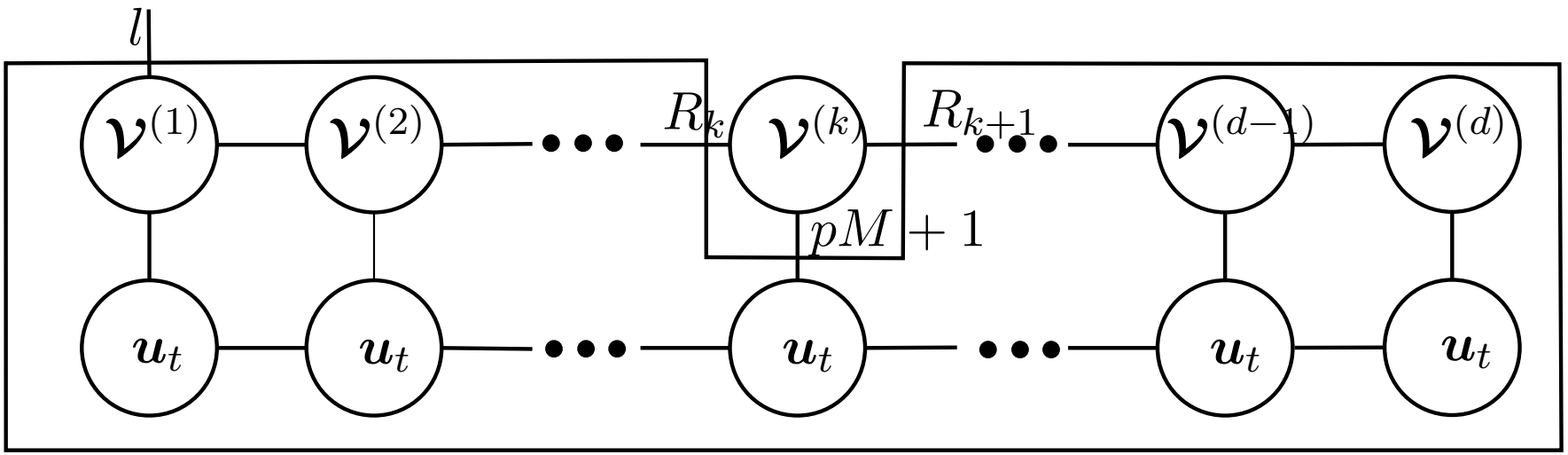
MIMO Volterra system identification problem

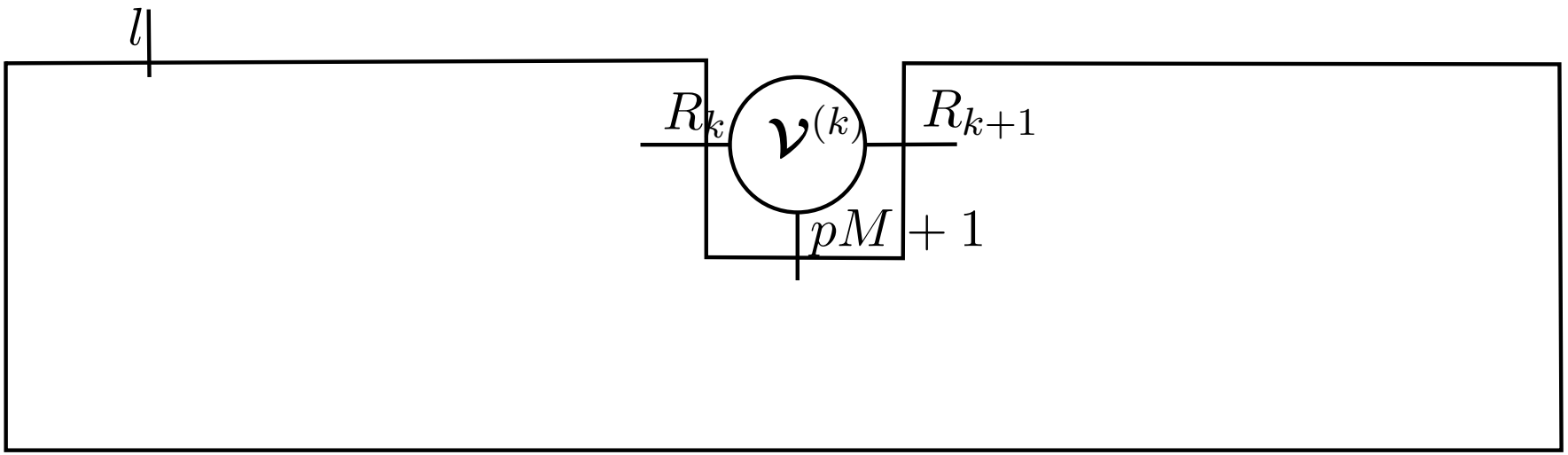
Given measured inputs $\{\boldsymbol{u}(t) \in \mathbb{R}^p\}_{t=1}^N$, outputs $\{\boldsymbol{y}(t) \in \mathbb{R}^l\}_{t=1}^N$, and an unknown MIMO Volterra model $\boldsymbol{V}$, estimate the tensor network tensors $\boldsymbol{\mathcal{V}}^{(1)}, \boldsymbol{\mathcal{V}}^{(2)}, \ldots, \boldsymbol{\mathcal{V}}^{(d)}$.

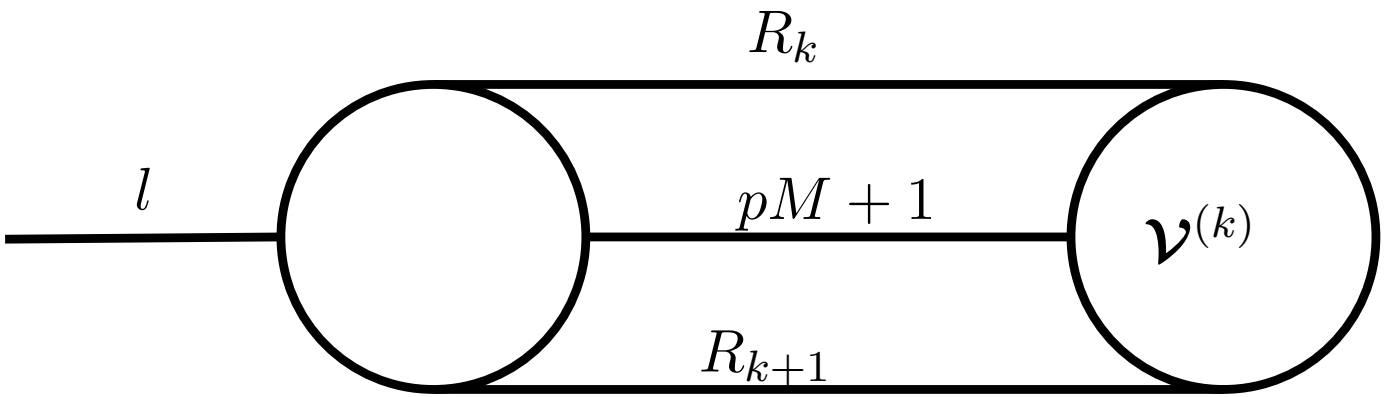
Alternating Linear Scheme (ALS)

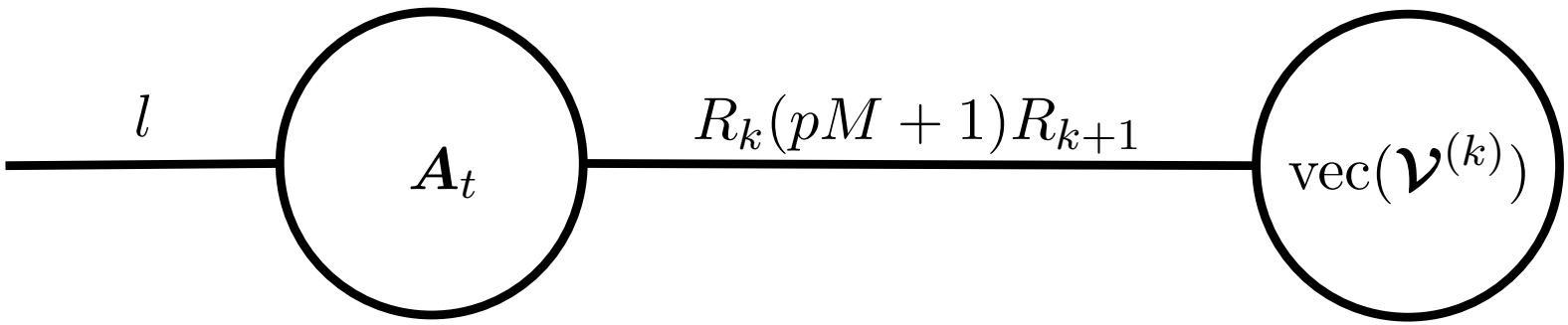
- initialize unknown tensors $\boldsymbol{\mathcal{V}}^{(1)}, \boldsymbol{\mathcal{V}}^{(2)}, \ldots, \boldsymbol{\mathcal{V}}^{(d)}$
- for $1 \leq k \leq d$
  - Assume $\boldsymbol{\mathcal{V}}^{(1)}, \ldots, \boldsymbol{\mathcal{V}}^{(k-1)}, \boldsymbol{\mathcal{V}}^{(k+1)}, \ldots, \boldsymbol{\mathcal{V}}^{(d)}$ known and update $\boldsymbol{\mathcal{V}}^{(k)}$.
  - Update $\boldsymbol{\mathcal{V}}^{(k)} \Rightarrow$ solve small linear system for $\mathrm{vec}(\boldsymbol{\mathcal{V}}^{(k)})$

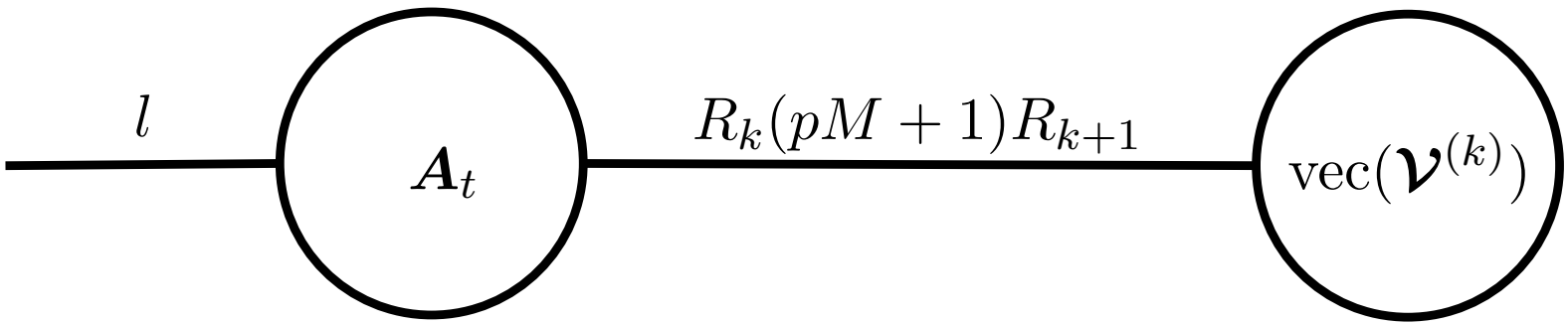

l
$\boldsymbol{\nu}^{(1)}$
$\boldsymbol{\nu}^{(2)}$
R_k
$\boldsymbol{\nu}^{(k)}$
R_{k+1}
$\boldsymbol{\nu}^{(d-1)}$
$\boldsymbol{\nu}^{(d)}$
$pM+1$
$\boldsymbol{u}_t$
$\boldsymbol{u}_t$
$\boldsymbol{u}_t$
$\boldsymbol{u}_t$
$\boldsymbol{u}_t$


l
$\boldsymbol{\nu}^{(1)}$
$\boldsymbol{\nu}^{(2)}$
R_k
$\boldsymbol{\nu}^{(k)}$
R_{k+1}
$\boldsymbol{\nu}^{(d-1)}$
$\boldsymbol{\nu}^{(d)}$
$pM+1$
$\boldsymbol{u}_t$
$\boldsymbol{u}_t$
$\boldsymbol{u}_t$
$\boldsymbol{u}_t$
$\boldsymbol{u}_t$


l
R_k
$\boldsymbol{\nu}^{(k)}$
R_{k+1}
$pM + 1$


R_k
l
$pM+1$
$\boldsymbol{\nu}^{(k)}$
R_{k+1}


l
$\boldsymbol{A}_t$
$R_k(pM+1)R_{k+1}$
$\mathrm{vec}(\boldsymbol{\mathcal{V}}^{(k)})$



$$\boldsymbol{y}(t) = \boldsymbol{A}_t \ \text{vec}(\boldsymbol{\mathcal{V}}^{(k)})$$

Updating $\boldsymbol{\mathcal{V}}^{(k)}$

$$\begin{pmatrix} \boldsymbol{y}(1) \\ \boldsymbol{y}(2) \\ \vdots \\ \boldsymbol{y}(N) \end{pmatrix} = \begin{pmatrix} \boldsymbol{A}_1 \\ \boldsymbol{A}_2 \\ \vdots \\ \boldsymbol{A}_N \end{pmatrix} \operatorname{vec}(\boldsymbol{\mathcal{V}}^{(k)})$$

MIMO Volterra ALS system identification algorithm

- Initialize unknown tensors $\boldsymbol{\mathcal{V}}^{(1)}, \boldsymbol{\mathcal{V}}^{(2)}, \ldots, \boldsymbol{\mathcal{V}}^{(d)}$
- While stopping criterion not satisfied
  - For $k = 1 : d$, Update $\boldsymbol{\mathcal{V}}^{(k)}$
  - For $k = d - 1 : -1 : 2$, Update $\boldsymbol{\mathcal{V}}^{(k)}$
- End while

MIMO Volterra ALS system identification algorithm

- Initialize unknown tensors $\boldsymbol{\mathcal{V}}^{(1)}, \boldsymbol{\mathcal{V}}^{(2)}, \ldots, \boldsymbol{\mathcal{V}}^{(d)}$
- While stopping criterion not satisfied
  - For $k = 1 : d$, Update $\boldsymbol{\mathcal{V}}^{(k)}$
  - For $k = d - 1 : -1 : 2$, Update $\boldsymbol{\mathcal{V}}^{(k)}$
- End while

Remarks

- Stopping criterion: fixed number of updates, residual on $||\boldsymbol{Y} - \hat{\boldsymbol{Y}}||_F / ||\boldsymbol{Y}||_F$, ...
- How to choose $R_2, \ldots, R_d$? ($\Rightarrow$ Modified ALS updates the ranks automatically)
- Orthogonalization step after solving the linear system ensures numerical stability.

Numerical Experiment

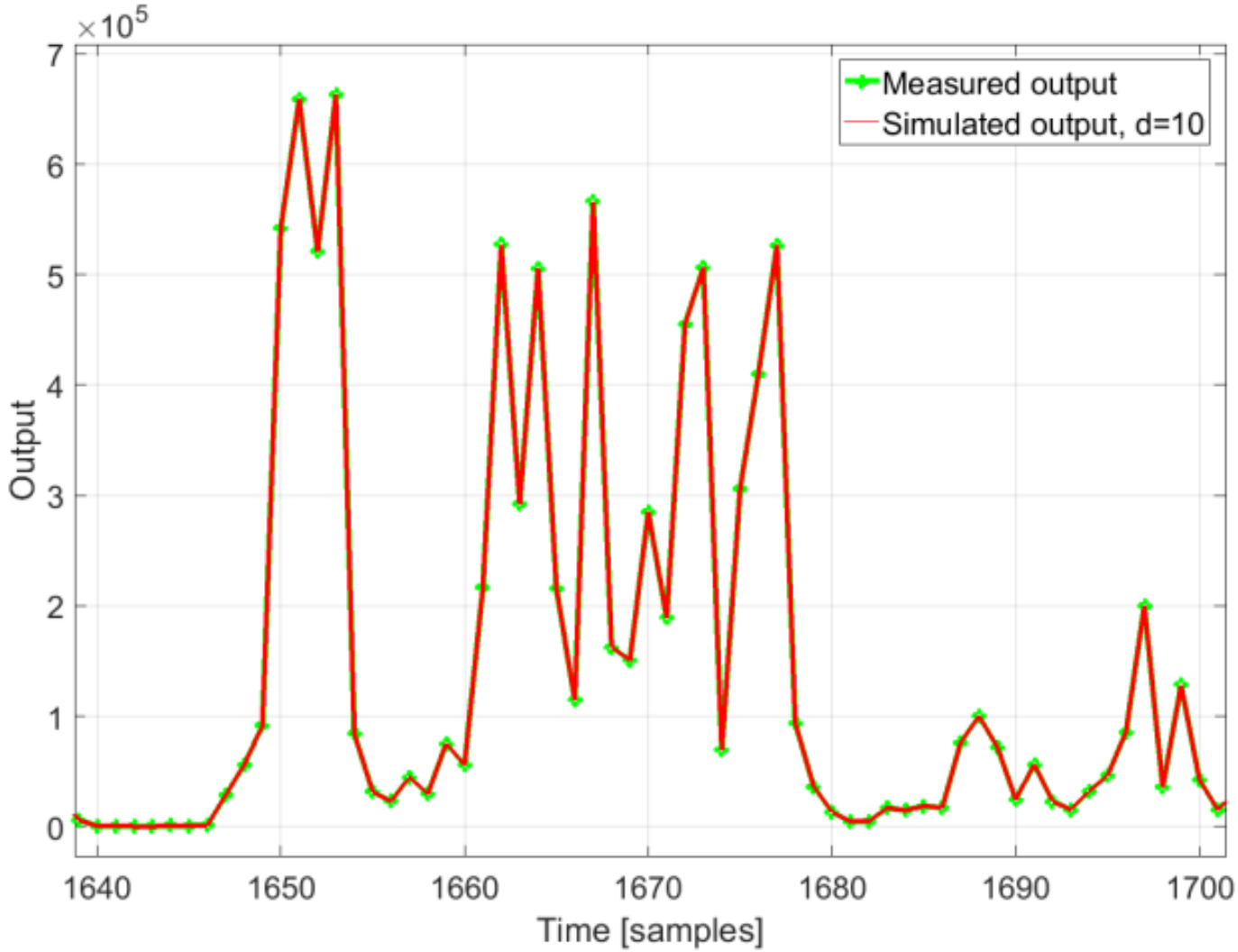
- Generated Volterra system with $d = 10$ and $M = 7$.
- The ith symmetric Volterra kernel contains entries

$$h_i(k_1, \ldots, k_i) = e^{-.1 \sum_{j=1}^{i} k_j^2}.$$

- white noise input: 700 samples for identification, 2000 samples for validation.
- relative approximation error $||\boldsymbol{Y} - \hat{\boldsymbol{Y}}||/||\boldsymbol{Y}|| \leq 10^{-4}$.

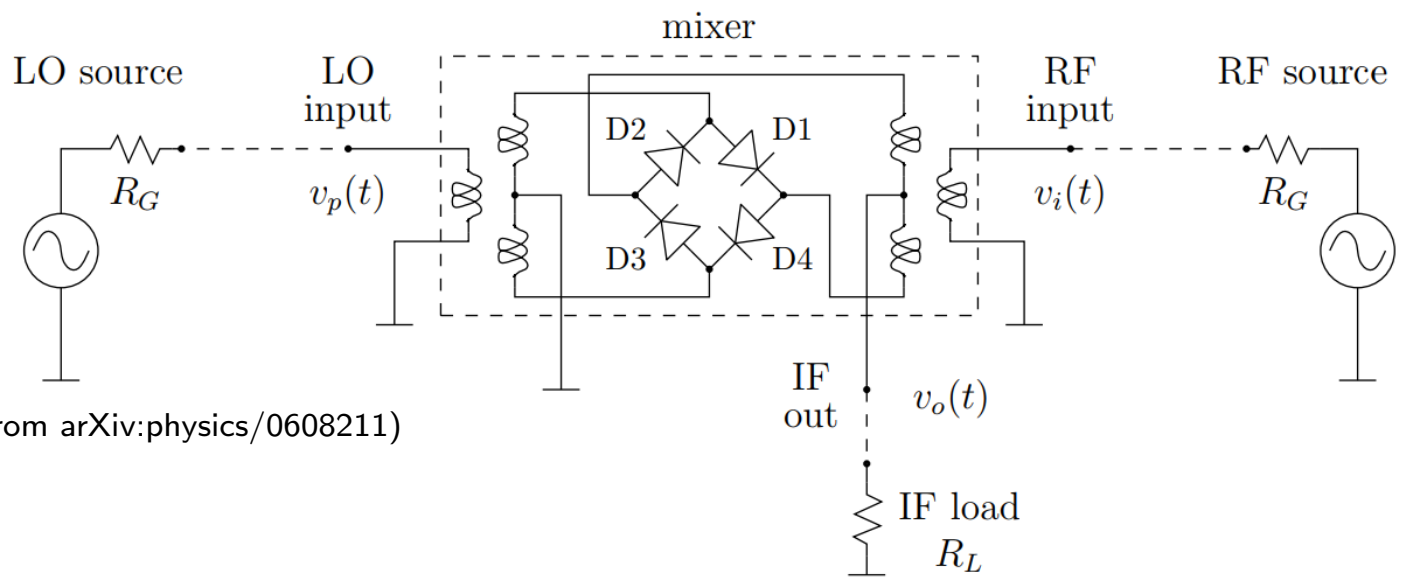
Table 1
Run times, maximal TT-rank and number of estimated Volterra tensor elements for an increasing degree d.

d	Run time [seconds]			max TT-rank	$(M+1)^d$
	$\boldsymbol{U}^\dagger \boldsymbol{y}$	ALS	MALS		
2	0.017	0.240	0.077	6	64
3	0.253	0.223	0.251	8	512
4	39.36	1.771	0.415	8	4096
5	NA	4.288	0.587	8	32768
6	NA	3.065	0.791	8	262144
7	NA	6.783	0.961	8	2097152
8	NA	13.11	1.199	8	16777216
9	NA	14.42	1.384	8	134217728
10	NA	18.37	1.576	8	1.0737e9


×10^5
7
6
5
4
3
2
1
0
Output
1640
1650
1660
1670
1680
1690
1700
Time [samples]
Measured output
Simulated output, d=10

Experiment: double balanced mixer for upconversion

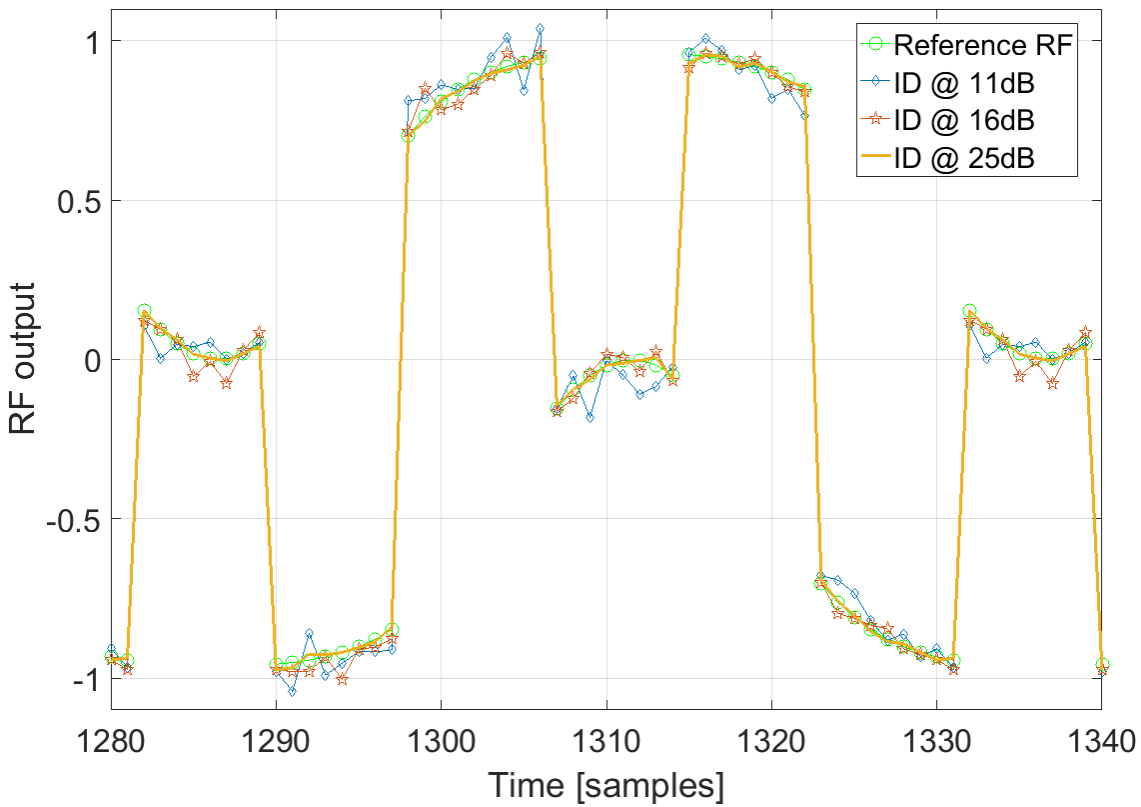
- 2 inputs: LO = 100Hz sine, IF= 300Hz square-wave.
- output= RF.
- Sampling frequency: 5kHz for 1 second
- Five different noise levels added to RF output: SNR=11dB,13dB,16dB,19dB,25dB.



(image from arXiv:physics/0608211)

Experiment

- $M = 2, d = 11 \Rightarrow (pM + 1)^d = 5^{11} = 48828125$
- $R_2 = \cdots = R_{11} = 5 \Rightarrow$ Each TN node has 125 unknowns
- Identification takes around 2 seconds on standard desktop pc



Time-varying MIMO Volterra identification

$$\boldsymbol{V}(t+1)^T = \boldsymbol{V}(t)^T + \Delta\boldsymbol{V}(t)^T$$
$$\boldsymbol{y}(t)^T = \mathrm{vec}(\boldsymbol{u}_t \circ \cdots \circ \boldsymbol{u}_t)^T \, \boldsymbol{V}(t)^T + \boldsymbol{e}(t)$$

- Time-varying linear state space
- "State" is a matrix $\boldsymbol{V}^T \in \mathbb{R}^{(pM+1)^d \times l}$
- Tensor network Kalman filter for the recursive estimation of $\boldsymbol{V}$

Time-varying MIMO Volterra identification

$$\boldsymbol{V}(t+1)^T = \boldsymbol{V}(t)^T + \Delta\boldsymbol{V}(t)^T$$
$$\boldsymbol{y}(t)^T = \mathrm{vec}(\boldsymbol{u}_t \circ \cdots \circ \boldsymbol{u}_t)^T \, \boldsymbol{V}(t)^T + \boldsymbol{e}(t)$$

- Time-varying linear state space
- "State" is a matrix $\boldsymbol{V}^T \in \mathbb{R}^{(pM+1)^d \times l}$
- Tensor network Kalman filter for the recursive estimation of $\boldsymbol{V}$

"Polynomial in the input" state space

$$\boldsymbol{x}(t+1) = \boldsymbol{A}\,\boldsymbol{x}(t) + \boldsymbol{B}\,\mathrm{vec}(\boldsymbol{u}_t \circ \cdots \circ \boldsymbol{u}_t)$$
$$\boldsymbol{y}(t) = \boldsymbol{C}\,\boldsymbol{x}(t) + \boldsymbol{D}\,\mathrm{vec}(\boldsymbol{u}_t \circ \cdots \circ \boldsymbol{u}_t).$$

- MOESP subspace identification with Tensor Networks

Open source MATLAB/Octave implementations

`https://github.com/kbatseli/`

References

- Batselier K., Chen Z., Wong N., Tensor Network alternating linear scheme for MIMO Volterra system identification, Automatica, vol. 84, 2017, pp. 26-35
- Batselier K., Chen Z., Wong N., A Tensor Network Kalman filter with an application in recursive MIMO Volterra system identification, Automatica, vol. 84, 2017, pp. 17-25
- Batselier K., Wong N., Matrix output extension of the tensor network Kalman filter with an application in MIMO Volterra system identification, Automatica, vol. 95, 2018, pp. 413-418
- Batselier K., Ko C.-Y., Wong N., Tensor network subspace identification of polynomial state space models, Automatica, vol. 95, 2018, pp. 187-196.