Time-Varying Systems and Computations

Klaus Diepold

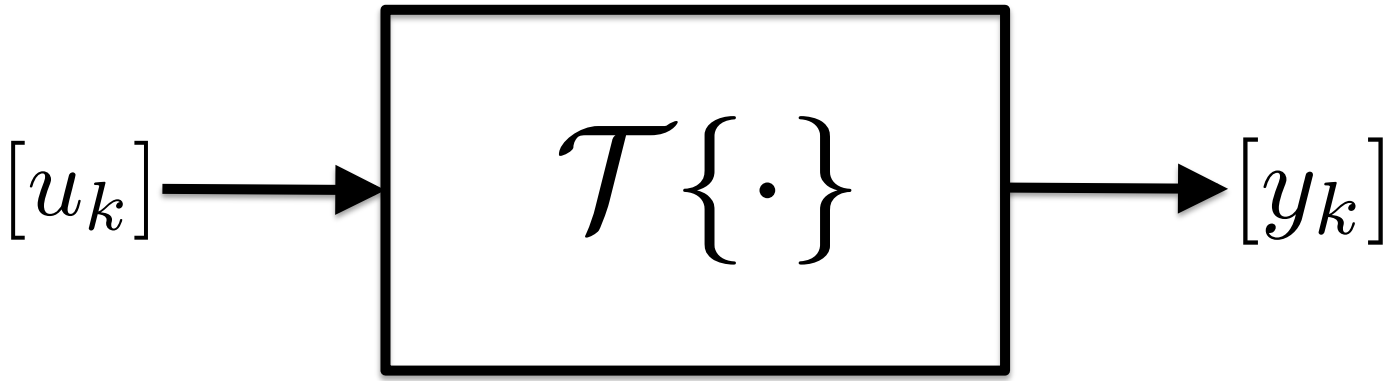
01.1 WS 2024

Computations

1. Numerical Simulations – Bottom Up
2. Data Analysis – Top Down
3. Cyber-Physical Systems - Embedded

Linear Systems

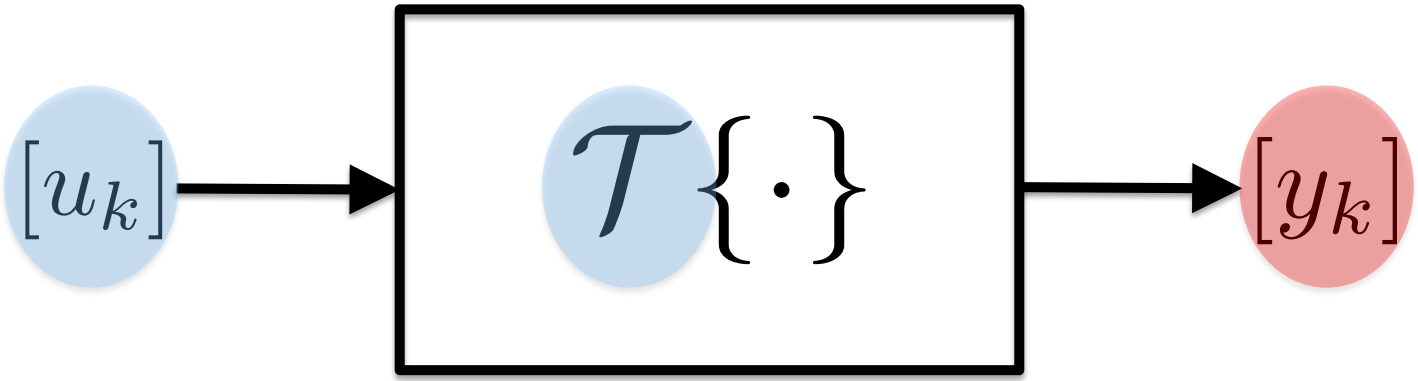
Standard Representation



$$T \cdot u = y$$

Standard Problems

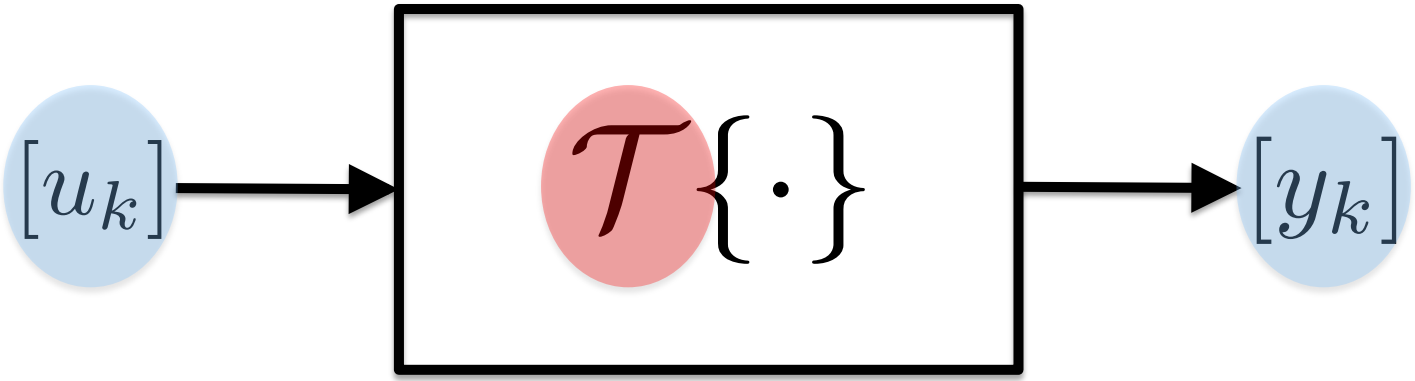
Filtering



$$(T, u) \mapsto y$$

Standard Problems

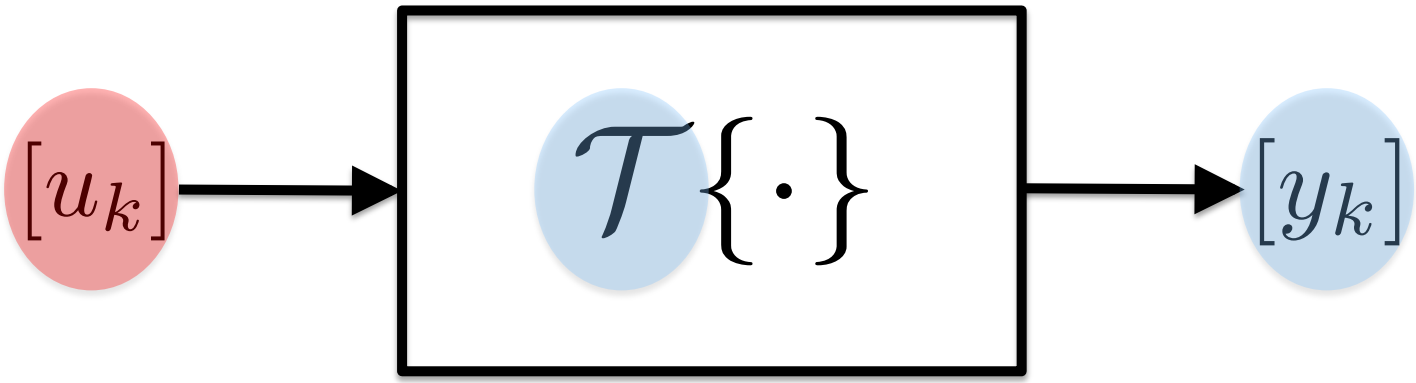
System Identification



$$(u, y) \mapsto T$$

Standard Problems

Inversion



$$(T, y) \mapsto u$$

Cost of Computations

Example: inner product

$$x^T y = [x_1, x_2, x_3, \dots x_n] \cdot \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} =$$

$$= x_1 y_1 + x_2 y_2 + x_3 y_3 + \cdots + x_n y_n$$

$$n \text{ Multiplications} + (n-1) \text{ Additions}$$

Typical Computations

- Inner product $x^T y$ $\mathcal{O}(n)$
- Matrix-Vector-Product $Tu = y$ $\mathcal{O}(n^2)$
- Matrix Addition $T_1 + T_2$ $\mathcal{O}(n^2)$
- Matrix-Matrix-Product $T_1 T_2$ $\mathcal{O}(n^3)$
- Matrix Inversion T^{-1} $\mathcal{O}(n^3)$

Data Sparse Matrix

- Diagonal Matrix $\mathcal{O}(n)$ Matrix Entries

$$D = \begin{bmatrix} d_1 & & & & \\ & d_2 & & & \\ & & d_3 & & \\ & & & \ddots & \\ & & & & d_n \end{bmatrix}$$

Data Sparse Matrix

- Sparse Matrix $\mathcal{O}(n)$ Matrix Entries

$$D = \begin{bmatrix} d_0 & & d_1 & & d_2 \\ & & & d_3 & \\ d_4 & & & & \\ & d_5 & d_6 & & d_7 \\ d_8 & & & & d_9 \end{bmatrix}$$

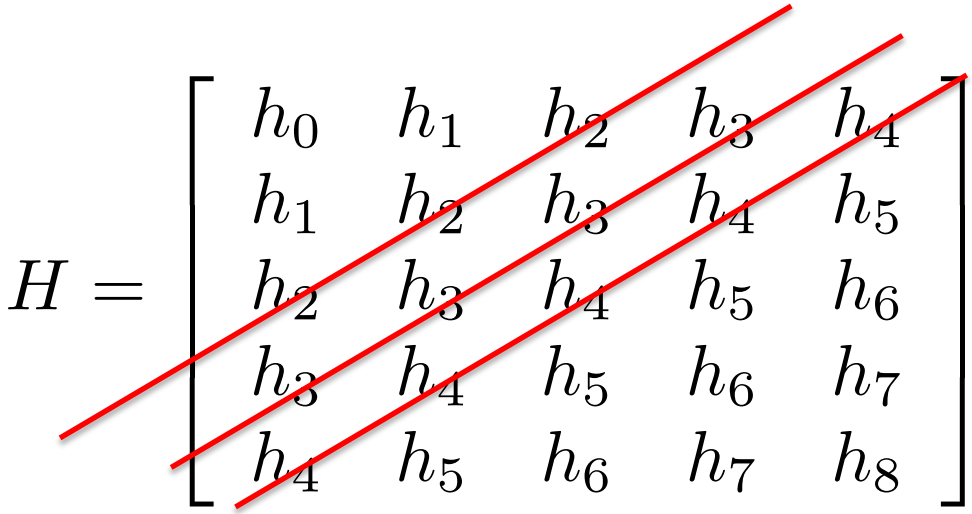
Data Sparse Matrix

- Toeplitz Matrix $\mathcal{O}(n)$ Matrix Entries

$$T = \begin{bmatrix} t_0 & t_1 & t_2 & t_3 & t_4 \\ t_{-1} & t_0 & t_1 & t_2 & t_3 \\ t_{-2} & t_{-1} & t_0 & t_1 & t_2 \\ t_{-3} & t_{-2} & t_{-1} & t_0 & t_1 \\ t_{-4} & t_{-3} & t_{-2} & t_{-1} & t_0 \end{bmatrix}$$

Data Sparse Matrix

- Hankel Matrix $\mathcal{O}(n)$ Matrix Entries



$$H = \begin{bmatrix} h_0 & h_1 & h_2 & h_3 & h_4 \\ h_1 & h_2 & h_3 & h_4 & h_5 \\ h_2 & h_3 & h_4 & h_5 & h_6 \\ h_3 & h_4 & h_5 & h_6 & h_7 \\ h_4 & h_5 & h_6 & h_7 & h_8 \end{bmatrix}$$

Data Sparse Matrix

- Vandermonde Matrix $\mathcal{O}(n)$ Matrix Entries

$$V = \begin{bmatrix} 1 & v_1 & v_1^2 & v_1^3 & v_1^4 \\ 1 & v_2 & v_2^2 & v_2^3 & v_2^4 \\ 1 & v_3 & v_3^2 & v_3^3 & v_3^4 \\ 1 & v_4 & v_4^2 & v_4^3 & v_4^4 \\ 1 & v_5 & v_5^2 & v_5^3 & v_5^4 \end{bmatrix}$$

Data Sparse Matrix

- Low Rank Matrix $\mathcal{O}(n)$ Matrix Entries

$$A = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} \cdot \begin{bmatrix} v_1 & v_2 & v_3 & v_4 & v_5 \end{bmatrix} =$$

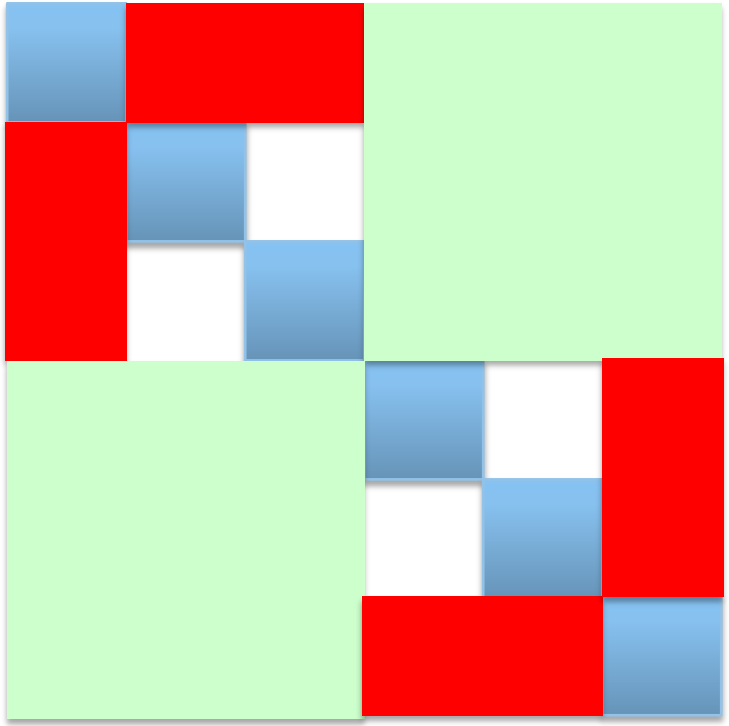
$$= \begin{bmatrix} u_1v_1 & u_1v_2 & u_1v_3 & u_1v_4 & u_1v_5 \\ u_2v_1 & u_2v_2 & u_2v_3 & u_2v_4 & u_2v_5 \\ u_3v_1 & u_3v_2 & u_3v_3 & u_3v_4 & u_3v_5 \\ u_4v_1 & u_4v_2 & u_4v_3 & u_4v_4 & u_4v_5 \\ u_5v_1 & u_5v_2 & u_5v_3 & u_5v_4 & u_5v_5 \end{bmatrix}$$

Data Sparse Matrix

- Band Matrices $\mathcal{O}(n)$ Matrix Entries

$$B = \begin{bmatrix} b_{11} & b_{12} & & & \\ b_{21} & b_{22} & b_{23} & & \\ & b_{32} & b_{33} & b_{34} & \\ & & b_{43} & b_{44} & b_{45} \\ & & & b_{54} & b_{55} \end{bmatrix}$$

Rank Structured Matrices



Evaluation Criteria

- Runtime of a program
- Parallelism of algorithm
- Accuracy and precision
- Robustness and Stability
- Match to Computing Architecture

Mission Statement

- Methodology for Designing Numerical Algos
- Efficient Algorithms for Embedded Computing
- Exploiting Structure in the Data
- Using Engineering Intuition
- → Time-Varying System Theory
- Considering the Computing Architecture
- Reformulating Computational Challenges