Decoupling multivariate functions using tensors

Mariya Ishteva

November 26, 2024

KU LEUVEN

A toy example

$x_1 \longrightarrow$ $\longrightarrow y_1$

$$f_1(x_1, x_2) = 54x_1^3 - 54x_1^2x_2 + 8x_1^2 + 18x_1x_2^2 + 16x_1x_2 - 2x_2^3 + 8x_2^2 + 8x_2 + 1,$$

$$f_2(x_1, x_2) = -27x_1^3 + 27x_1^2x_2 - 24x_1^2 - 9x_1x_2^2 - 48x_1x_2 - 15x_1 + x_2^3 - 24x_2^2 - 19x_2 - 3$$

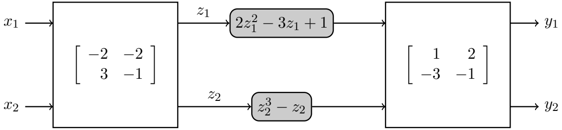
$x_2 \longrightarrow$ $\longrightarrow y_2$

A toy example

$$f_1(x_1, x_2) = 54x_1^3 - 54x_1^2x_2 + 8x_1^2 + 18x_1x_2^2 + 16x_1x_2 - 2x_2^3 + 8x_2^2 + 8x_2 + 1,$$

$$f_2(x_1, x_2) = -27x_1^3 + 27x_1^2x_2 - 24x_1^2 - 9x_1x_2^2 - 48x_1x_2 - 15x_1 + x_2^3 - 24x_2^2 - 19x_2 - 3$$

$$\updownarrow$$



Based on joint work with

Philippe Dreesen

Konstantin Usevich

Joppe De Jonghe

Johan Schoukens

Koen Tiels, Maarten Schoukens, David Westwick,
Ivan Markovsky, Gabriel Hollander, Thomas Goossens,
Yassine Zniyed, André de Almeida, ...

Decoupling multivariate functions using tensors

History

Computation

Applications

Waring's problem (1770)

Every natural number can be represented as
the sum of at most 4 squares, 9 cubes, or 19 fourth powers.

$$30 = 5^2 + 2^2 + 1^2$$
$$30 = 3^3 + 1^3 + 1^3 + 1^3$$
$$30 = 2^4 + 1^4 + \ldots + 1^4$$

Hilbert–Waring theorem, 1909:

$$\forall k \in \mathbb{N}, \exists s \in \mathbb{N} : \quad \forall n \in \mathbb{N} : \quad n = \sum_{i \leq s} a_i^k, \quad a_i \in \mathbb{N}, i = 1, \ldots, s.$$

Waring's problem for homogeneous polynomials

Decompose a homogeneous multivariate polynomial $f(x_1, \ldots, x_m)$ of degree d as

$$f(x_1, \ldots, x_m) = \sum_{i=1}^{r} u_i (v_{1i} x_1 + \cdots + v_{mi} x_m)^d;$$

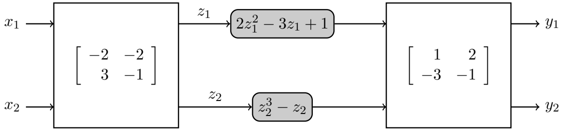
r is called the Waring rank.

Waring's problem for a set of non-homogeneous polynomials

$$f_1(x_1, x_2) = 54x_1^3 - 54x_1^2x_2 + 8x_1^2 + 18x_1x_2^2 + 16x_1x_2 - 2x_2^3 + 8x_2^2 + 8x_2 + 1,$$

$$f_2(x_1, x_2) = -27x_1^3 + 27x_1^2x_2 - 24x_1^2 - 9x_1x_2^2 - 48x_1x_2 - 15x_1 + x_2^3 - 24x_2^2 - 19x_2 - 3$$

$\updownarrow$



$$\mathbf{f}(\mathbf{x}) = \mathbf{U}\mathbf{g}(\mathbf{V}^T\mathbf{x})$$

Decoupling multivariate functions using tensors

History

Computation

Applications

We compute the decomposition by a first-order approach

$$\mathbf{f}(\mathbf{x}) = \mathbf{U}\mathbf{g}(\mathbf{V}^T\mathbf{x})$$

We compute the decomposition by a first-order approach

If $$\mathbf{f}(\mathbf{x}) = \mathbf{U}\mathbf{g}(\mathbf{V}^T\mathbf{x}) \ ,$$

then $$\underbrace{\left[\frac{\partial f_i(\mathbf{x})}{\partial x_j}\right]}_{\mathbf{J_f}(\mathbf{x})} = \mathbf{U}\begin{bmatrix} g_1'(\mathbf{v}_1^T\mathbf{x}) & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & g_r'(\mathbf{v}_r^T\mathbf{x}) \end{bmatrix}\mathbf{V}^T.$$

We compute the decomposition by a first-order approach

If $$\mathbf{f}(\mathbf{x}) = \mathbf{U}\mathbf{g}(\mathbf{V}^T\mathbf{x}) \ ,$$

then $$\underbrace{\left[\frac{\partial f_i(\mathbf{x})}{\partial x_j}\right]}_{\mathbf{J}_\mathbf{f}(\mathbf{x})} = \mathbf{U} \begin{bmatrix} g_1'(\mathbf{v}_1^T\mathbf{x}) & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & g_r'(\mathbf{v}_r^T\mathbf{x}) \end{bmatrix} \mathbf{V}^T.$$

- Collect Jacobian matrices $\mathbf{J}_\mathbf{f}^{(1)}, \mathbf{J}_\mathbf{f}^{(2)}, \mathbf{J}_\mathbf{f}^{(3)}, \mathbf{J}_\mathbf{f}^{(4)}, \mathbf{J}_\mathbf{f}^{(5)}, \ldots$ and diagonalize them simultaneously

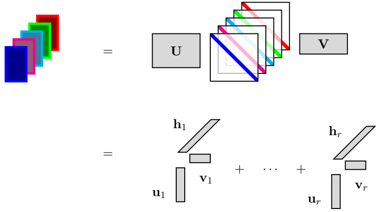
We compute the decomposition by a first-order approach

If $$\mathbf{f}(\mathbf{x}) = \mathbf{U}\mathbf{g}(\mathbf{V}^T\mathbf{x}) \;,$$

then $$\underbrace{\left[\frac{\partial f_i(\mathbf{x})}{\partial x_j}\right]}_{\mathbf{J}_\mathbf{f}(\mathbf{x})} = \mathbf{U}\begin{bmatrix} g_1'(\mathbf{v}_1^T\mathbf{x}) & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & g_r'(\mathbf{v}_r^T\mathbf{x}) \end{bmatrix}\mathbf{V}^T.$$

- Collect Jacobian matrices $\mathbf{J}_\mathbf{f}^{(1)}, \mathbf{J}_\mathbf{f}^{(2)}, \mathbf{J}_\mathbf{f}^{(3)}, \mathbf{J}_\mathbf{f}^{(4)}, \mathbf{J}_\mathbf{f}^{(5)}, \ldots$ and diagonalize them simultaneously

  Tool: Canonical Polyadic Decomposition (CPD)

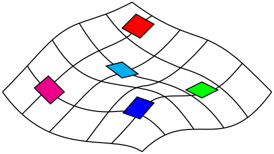


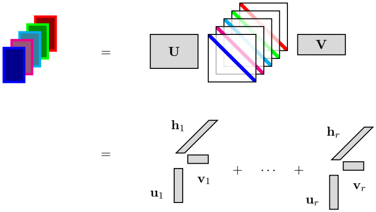
Algorithm

1. Construct tensor of Jacobians
   $\mathcal{J}_{\mathbf{f}} = \left\{ \mathbf{J}_{\mathbf{f}}^{(1)}, \mathbf{J}_{\mathbf{f}}^{(2)}, \mathbf{J}_{\mathbf{f}}^{(3)}, \mathbf{J}_{\mathbf{f}}^{(4)}, \mathbf{J}_{\mathbf{f}}^{(5)}, \ldots \right\}$

2. CPD of $\mathcal{J}_{\mathbf{f}}$ gives $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{H}$

3. Retrieve coefficients of $g_i(\cdot)$ from $\mathbf{y}^{(k)} = \mathbf{U}\left[g_i(\mathbf{v}_i^T \mathbf{x}^{(k)})\right]$
   (solving linear system)





Variations of the main problem are useful in various contexts

- Scalar functions

Variations of the main problem are useful in various contexts

- Scalar functions
  $\rightarrow$ second-order (Hessian) approach

Variations of the main problem are useful in various contexts

- Scalar functions
  $\rightarrow$ second-order (Hessian) approach
- Uniqueness, noise reduction

Variations of the main problem are useful in various contexts

- Scalar functions
  $\rightarrow$ second-order (Hessian) approach

- Uniqueness, noise reduction
  $\rightarrow$ parametrization of the internal functions
  $\rightarrow$ joint decompositions (combining Jacobians and Hessians)

Variations of the main problem are useful in various contexts

- Scalar functions
  $\rightarrow$ second-order (Hessian) approach
- Uniqueness, noise reduction
  $\rightarrow$ parametrization of the internal functions
  $\rightarrow$ joint decompositions (combining Jacobians and Hessians)
- Meaningful multivariate internal functions

Variations of the main problem are useful in various contexts

- Scalar functions
  $\rightarrow$ second-order (Hessian) approach
- Uniqueness, noise reduction
  $\rightarrow$ parametrization of the internal functions
  $\rightarrow$ joint decompositions (combining Jacobians and Hessians)
- Meaningful multivariate internal functions
  $\rightarrow$ block-term decomposition

Variations of the main problem are useful in various contexts

- Scalar functions
  $\rightarrow$ second-order (Hessian) approach
- Uniqueness, noise reduction
  $\rightarrow$ parametrization of the internal functions
  $\rightarrow$ joint decompositions (combining Jacobians and Hessians)
- Meaningful multivariate internal functions
  $\rightarrow$ block-term decomposition
- Multiple layers

Variations of the main problem are useful in various contexts

- Scalar functions
  $\rightarrow$ second-order (Hessian) approach
- Uniqueness, noise reduction
  $\rightarrow$ parametrization of the internal functions
  $\rightarrow$ joint decompositions (combining Jacobians and Hessians)
- Meaningful multivariate internal functions
  $\rightarrow$ block-term decomposition
- Multiple layers
  $\rightarrow$ ParaTuck-Z decomposition

Decoupling multivariate functions using tensors

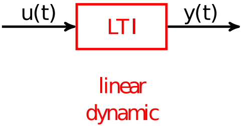
History

Computation

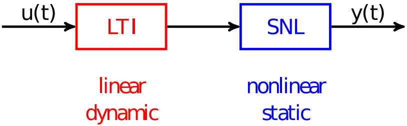
Applications

- System identification (2)
- Neural networks

Block-oriented models provide a good trade-off between simplicity and descriptive power

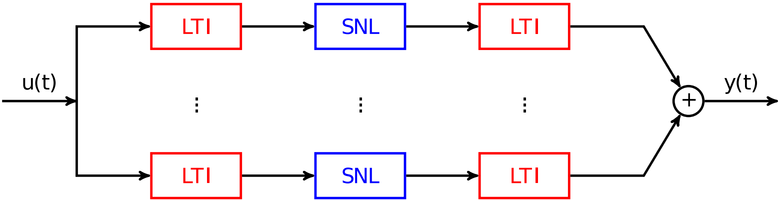


Block-oriented models provide a good trade-off between simplicity and descriptive power

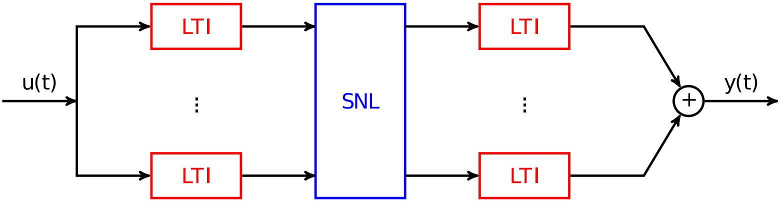


Block-oriented models provide a good trade-off between simplicity and descriptive power

Block-oriented models provide a good trade-off between simplicity and descriptive power



Block-oriented models: parameter estimation



Step 1: estimate the parameters of the LTI blocks
Step 2: fit a multivariate polynomial
Step 3: decouple the multivariate polynomial

Decoupling multivariate functions using tensors

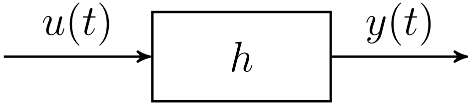
History

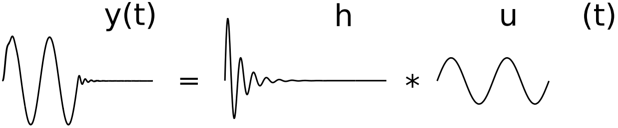
Computation

Applications

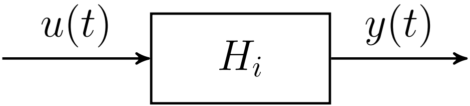
- System identification (2)
- Neural networks

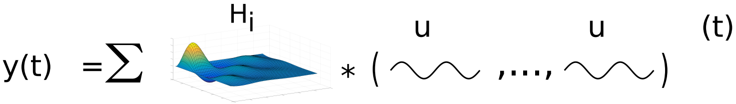
The impulse response completely characterizes *linear* dynamical systems



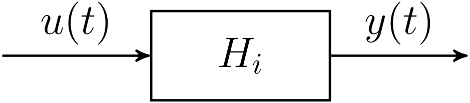


The Volterra kernels completely characterize *nonlinear* dynamical systems





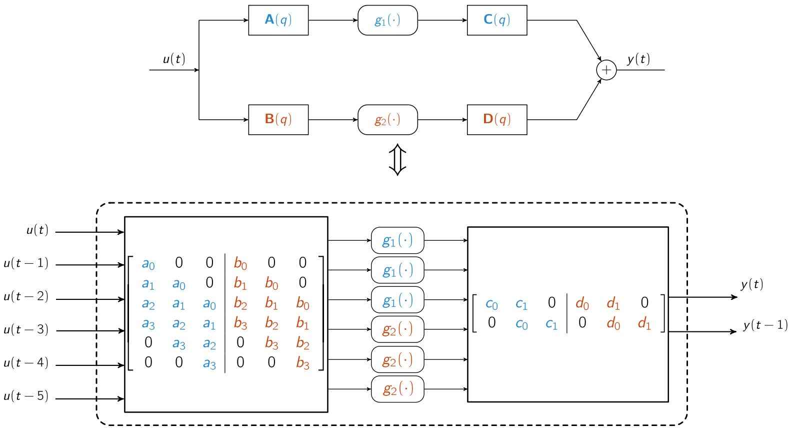
The Volterra kernels completely characterize *nonlinear* dynamical systems



Volterra series are polynomials
of time-shifted input signals

$$y(t) = \sum_i \sum_{\tau_1,\ldots,\tau_i} \underbrace{H_i(\tau_1,\ldots,\tau_i)}_{\text{kernels}} \underbrace{u(t-\tau_1)\cdots u(t-\tau_i)}_{\text{time-shifted inputs}}$$

Decoupling Volterra representations



Decoupling multivariate functions using tensors

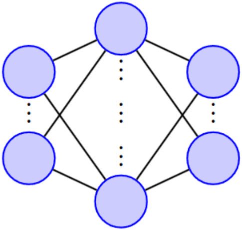
History

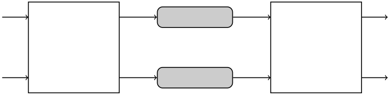
Computation

Applications

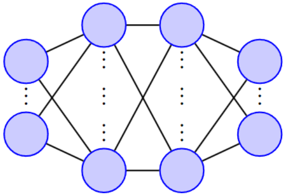
- System identification
- Neural networks

Neural networks with one layer are decoupled functions but can be compressed using flexible activation functions

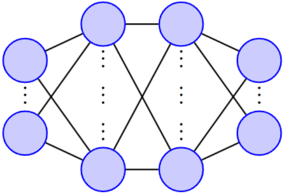


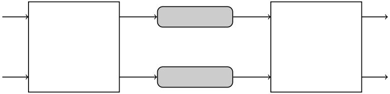


Neural networks with two layers can be compressed to one layer with flexible activation functions

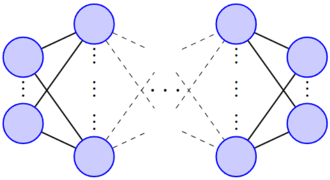


Neural networks with two layers can be compressed to one layer with flexible activation functions

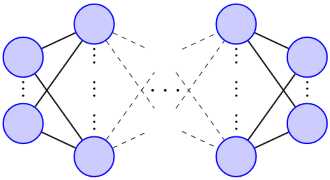


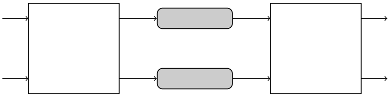


Neural networks with multiple layers can be compressed to one layer with flexible activation functions

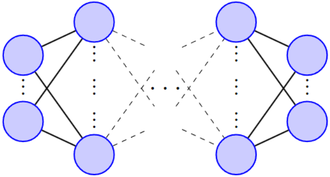


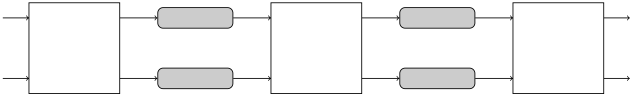
Neural networks with multiple layers can be compressed to one layer with flexible activation functions





Neural networks with multiple layers can be compressed to two layers with flexible activation functions using ParaTuck-2





- Multilayer Tensor-based Neural Network Compression with Flexible Activation Functions
- A lifting approach to ParaTuck-2 tensor decompositions

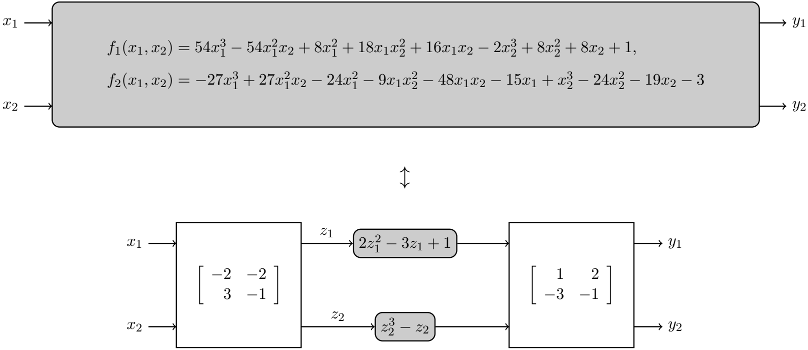
Decoupling multivariate functions using tensors

History

Computation

Applications

A toy example



Decoupling multivariate functions using tensors

Mariya Ishteva

November 26, 2024

KU LEUVEN