Two-layer decoupling of multivariate polynomials with coupled ParaTuck and CP decompositions

Konstantin Usevich, Yassine Zniyed, Mariya Ishteva, Philippe Dreesen, André L F de Almeida

Two-layer decoupling of multivariate polynomials with coupled ParaTuck and CP decompositions*

Konstantin Usevich[b], Yassine Zniyed[a], Mariya Ishteva[c], Philippe Dreesen[d], André L. F. de Almeida[e]

[a] *Univ. de Toulon, Aix Marseille Univ., CNRS, Seatech, LIS UMR 7020, France.*
[b] *Université de Lorraine, CNRS, CRAN UMR 7039, France.*
[c] *KU Leuven, Dept. Computer Science, ADVISE-NUMA, Belgium.*
[d] *Maastricht University, Faculty of Science and Engineering, Department of Advanced Computing Sciences, The Netherlands.*
[e] *Department of Teleinformatics Engineering, Federal University of Fortaleza, Brazil.*

Abstract

In this paper, we propose a new method for multivariate function approximation that generalized the classical decoupling problem. In the context of neural network, this can be seen as a two-layer feedforward network learning problem. In this work, we make use of both first and second-order information of the original function, modeled through paratuck and canonical polyadic (CP) decompositions, respectively. However, it is currently a challenge in the literature to handle the paratuck decomposition effectively. Our approach is a methodological work that demonstrates how the paratuck and CP decompositions can be combined in a coupled manner to achieve function decoupling according to the new model. Numerical simulations show the effectiveness of the proposed method on a simple synthetic example, demonstrating its ability to approximate multivariate functions accurately.

Keywords: paratuck decomposition, cp decomposition, polynomial decoupling, neural network, coupled/structured tensor decomposition

1. Introduction

The problem of learning to imitate and approximate complex nonlinear functions is crucial for solving many scientific challenges, including neural network learning [18] and system identification [8]. Neural networks have become a

*This research was partially supported by the ANR (Agence Nationale de Recherche) grant LeaFleT (ANR-19-CE23-0021).

Email addresses: konstantin.usevich@cnrs.fr (Konstantin Usevich), zniyed@univ-tln.fr (Yassine Zniyed), mariya.ishteva@kuleuven.be (Mariya Ishteva), philippe.dreesen@kuleuven.be (Philippe Dreesen), andre@gtel.ufc.br (André L. F. de Almeida)

widespread tool for various applications in areas such as computer vision [16], natural language processing [6], and predictive modeling [26]. Their ability to learn complex relationships between inputs and outputs has resulted in impressive performance on a wide range of tasks. One of the key components of neural networks is the activation function, which determines the output of a neuron given its input. The choice of activation function can have a significant impact on network performance [2]. Currently, activation functions are typically fixed prior to training, limiting the expressiveness of the network. This is why researchers have proposed various shapes of activation functions in the literature, such as step, sigmoid [7], tanh [5], and ReLU [9], to improve the representational capacity of neural networks.

Flexible activation functions [2] are a key factor in enabling neural networks to approximate complex input-output relationships. Traditional activation functions are fixed prior to learning and do not adapt to the specific input-output relationship being modeled. Flexible activation functions, on the other hand, are learned during the training process, allowing them to adapt to the underlying data and capture complex non-linear dependencies. This results in a more expressive and powerful model of the system dynamics. Some examples of flexible activation functions include the Generalized sigmoid [21], the switch [23], the paratemeric ReLU [13], to mention a few.

In the context of neural networks, the Universal Approximation Theorem [15] states that a feedforward neural network with a single hidden layer, using a non-constant, bounded, and continuous activation function, can approximate any continuous function to an arbitrary degree of accuracy. However, this theorem does not hold for discontinuous functions. A one-layer network with a continuous activation function is not capable of approximating functions with discontinuities, as it can only model continuous relationships. This means that a one-layer network may not be suitable for certain applications where discontinuous relationships are present, such as in certain control problems [22] or digital signal processing tasks [29]. To model these types of relationships, a multi-layer neural network or an alternative approach may be necessary. The use of a two-layer neural network provides a more expressive and powerful model of the system dynamics compared to a one-layer network. This implies that a two-layer network has the ability to model a wider range of input-output relationships compared to a one-layer network. The added complexity of a second layer combined with the flexible activation functions allows the network to learn a more complex mapping between inputs and outputs, capturing non-linear dependencies that are not possible with a one-layer network.

In the field of system identification, the classical decoupling problem [8] refers to the task of decomposing a multivariate polynomial function as linear combinations of univariate polynomials in linear forms of the input variables. Such an approach has applications in system identification, approximation theory, neural networks, *etc.* The decoupling problem is equivalent to modeling the system as a one-layer feedforward neural network with polynomial activation functions.

Several tensor-based solutions [8, 24, 20] have been proposed to find the decoupled representation. Each solution uses a different tensor representation,

but they all lead to a canonical polyadic decomposition (CPD) [14, 11, 3]. In this paper, we propose to use the first-order information of the polynomials for a new decoupled representation. Such a solution has been used in [8] for the classical decoupling problem and it leads to a CPD Jacobian tensor. For the new proposed representation here, the Jacobian tensor will follow a ParaTuck (PT) decomposition [12, 10, 4]. To the best of the authors' knowledge, there is no existing algorithm that can solve the PT decomposition when no prior knowledge is given. The ALS-type algorithm, introduced in [4], has been shown to have convergence issues. In most cases where the PT decomposition is used [28], some of the parameters are assumed to be known, which simplifies the optimization problem.

In summary, we introduce a novel decoupled representation that includes two layers instead of one in the classical representation. This work provides an expression for the first and second-order information of the new model and shows that the jacobian-based tensor follows a PT decomposition. Additionally, the paper provides an algorithm that computes a coupled factorization, allowing to retrieve the new decoupled representation (*i.e.,* the weights and the flexible activation functions in the context of neural networks), which combines the jacobian and hessian-based tensors.

2. Notation and tensor definitions

2.1. Notation

The symbols $(\cdot)^\dagger$ and $\text{rank}(\cdot)$ denote, respectively, the pseudo-inverse and the rank of a matrix. The outer, Hadamard and Khatri-Rao products are denoted to by $\otimes$, $\boxdot$, $\odot$, respectively. Tensors are represented by bold calligraphic capital letters, e.g., $\mathcal{X}$. $\mathcal{X}_{i,:,:}$, $\mathcal{X}_{:,j,:}$ and $\mathcal{X}_{:,:,k}$ are the i-th horizontal, j-th lateral and k-th frontal slices of sizes $n_2 \times n_3$, $n_1 \times n_3$ and $n_1 \times n_2$, respectively, of the tensor $\mathcal{X}$ of size $n_1 \times n_2 \times n_3$. The norm of a tensor $\mathcal{X}$ is the square root of the sum of the squares of all its elements, *i.e.,* $||\mathcal{X}|| = \sqrt{\sum_{i=1}^{n1}\sum_{j=1}^{n2}\sum_{k=1}^{n3}\mathcal{X}_{i,j,k}}$. $\mathcal{I}_{3,r}$ denotes the 3-order identity tensor of size $r \times r \times r$. The contraction on the kth index of a tensor is denoted as $\underset{k}{\bullet}$ [25]. The operator $\text{diag}(\cdot)$ forms a diagonal matrix from its vector argument, or captures the diagonal of a matrix if the argument is a matrix. $\text{unfold}_k\mathcal{X}$ refers to the unfolding of tensor $\mathcal{X}$ over its k-th mode [17].

2.2. CPD and matrix diagonalization

The CPD decomposes a tensor $\mathcal{X}$ into a sum of r rank-1 tensors. Following this definition, we can express a CPD tensor $\mathcal{X}$ of size $n_1 \times n_2 \times n_3$ as follows.

$$\mathcal{X} = [\![\lambda;\, \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \overset{\text{def}}{=} \sum_{k=1}^{r} \lambda_k \mathbf{a}_k \otimes \mathbf{b}_k \otimes \mathbf{c}_k,$$

with $\lambda \in \mathbb{R}^r$ and the factor matrices $\mathbf{A} \in \mathbb{R}^{n_1 \times r}, \mathbf{B} \in \mathbb{R}^{n_2 \times r}, \mathbf{C} \in \mathbb{R}^{n_3 \times r}$ are given by

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_r \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{b}_1 & \cdots & \mathbf{b}_r \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} \mathbf{c}_1 & \cdots & \mathbf{c}_r \end{bmatrix}.$$

For simplicity, and without loss of generality, we can omit the first factor, and denote $\mathcal{X}$ as

$$\mathcal{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] = \mathcal{I}_{3,r} \underset{1}{\bullet} \mathbf{A} \underset{2}{\bullet} \mathbf{B} \underset{3}{\bullet} \mathbf{C}. \tag{1}$$

Considering the definition of the CPD, we can observe that the frontal, vertical, and horizontal slices of the third-order tensor $\mathcal{X}$ in (1) can be viewed as three sets of matrices that can be jointly diagonalized by two factor matrices [1]. For example, for the frontal slices, we have

$$\mathcal{X}_{:,:,k} = \mathbf{A} \operatorname{diag}(\mathbf{C}_{k,:}) \mathbf{B}^{\mathsf{T}}. \tag{2}$$

2.3. ParaTuck decomposition

The paratuck decomposition can be seen as two level generalization of the CP decomposition. The PT model has been proposed in psychometric literature [12] in 1994, but has not been applied due to lack of published algorithm [4]. However, it has been utilized in telecommunication applications [28] with the presence of prior knowledge. The PT decomposition is defined as follows. Assume we have a $n_1 \times n_2 \times n_3$ tensor $\mathcal{X}$. The PT decomposition is defined throughout its slices, by a pair of ranks (r, s) and five factor matrices, as:

$$\mathcal{X}_{:,:,k} = \mathbf{A} \operatorname{diag}(\mathbf{g}_k) \mathbf{F} \operatorname{diag}(\mathbf{h}_k) \mathbf{B}^{\mathsf{T}}, \tag{3}$$

with $\mathbf{A} \in \mathbb{R}^{n_1 \times r}$, $\mathbf{B} \in \mathbb{R}^{n_2 \times s}$, $\mathbf{F} \in \mathbb{R}^{r \times s}$, $\mathbf{G} = \begin{bmatrix} \mathbf{g}_1 & \cdots & \mathbf{g}_{n_3} \end{bmatrix} \in \mathbb{R}^{r \times n_3}$, and $\mathbf{H} = \begin{bmatrix} \mathbf{h}_1 & \cdots & \mathbf{h}_{n_3} \end{bmatrix} \in \mathbb{R}^{s \times n_3}$. One can notice that (3) can be viewed as a multilevel version of (2). Alternatively, we can define the paratuck decomposition in the Tucker [27] form as:

$$\mathcal{T} = \mathcal{C} \underset{1}{\bullet} \mathbf{A} \underset{2}{\bullet} \mathbf{B},$$

where $\mathcal{C} \in \mathbb{R}^{r \times s \times n_3}$ is the paratuck core tensor given by

$$\mathcal{C}_{ijk} = F_{ij} G_{ik} H_{jk}, \tag{4}$$

or equivalently, using the Hadamard product, as:

$$\mathcal{C} = \mathbf{F} \boxdot_{\{r,s\}} \mathcal{S},$$

where $\mathcal{S}$, of size $r \times s \times n_3$, follows a CPD such that $\mathcal{S} = \mathcal{I}_{3,n_3} \bullet_1 \mathbf{G}^T \bullet_2 \mathbf{H}^T$. It is worth mentioning that if the factor matrices $\mathbf{A}$ and $\mathbf{B}$ are full column rank and are known, then we can recover the paratuck core $\mathcal{C}$. One should also notice that if $\mathcal{C}$ is known, the factor matrices $\mathbf{F}$, $\mathbf{G}$ and $\mathbf{H}$ can be easily retrieved. Indeed, an elementwise division of two slices is a rank one matrix

$$\frac{\mathcal{C}_{ijk}}{\mathcal{C}_{ijk'}} = \frac{G_{ik}}{G_{ik'}} \frac{H_{jk}}{H_{jk'}}.$$

To sum, the main difficulty for computing a PT decomposition is to find the factor matrices $\mathbf{A}$ and $\mathbf{B}$ from $\mathcal{X}$.

2.4. Paratuck ambiguities

The PT decomposition can be unique, in the same sense as the CPD, under some mild conditions [10]. This means that PT is prone to scaling and permutation ambiguities, *i.e.,* multiple solutions can exist for the same decomposition problem. It has be shown in [10], that if the 3-order tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ with paratuck decomposition (3), then there exists an alternative decomposition

$$\mathcal{T}_{:,:,k} = \tilde{\mathbf{A}} \operatorname{diag}(\tilde{\mathbf{g}_k}) \tilde{\mathbf{F}} \operatorname{diag}(\tilde{\mathbf{h}_k}) \tilde{\mathbf{B}}^{\mathsf{T}}, \tag{5}$$

where

$$\mathbf{A} = \tilde{\mathbf{A}} \cdot (\mathbf{\Pi_A} \cdot \mathbf{\Lambda_A}), \tag{6}$$

$$\mathbf{B} = \tilde{\mathbf{B}} \cdot (\mathbf{\Pi_B} \cdot \mathbf{\Lambda_B}), \tag{7}$$

$$\mathbf{F} = (\bar{\mathbf{\Lambda}}_\mathbf{A} \cdot \mathbf{\Lambda}_\mathbf{A}^{-1} \cdot \mathbf{\Pi}_\mathbf{A}^{\mathsf{T}}) \cdot \tilde{\mathbf{F}} \cdot (\mathbf{\Pi_B} \cdot \mathbf{\Lambda}_\mathbf{B}^{-1} \cdot \bar{\mathbf{\Lambda}}_\mathbf{B}), \tag{8}$$

$$\mathbf{g}_k = (\alpha_k \cdot \bar{\mathbf{\Lambda}}_\mathbf{A}^{-1} \mathbf{\Pi}_\mathbf{A}^{\mathsf{T}}) \cdot \tilde{\mathbf{g}}_k, \tag{9}$$

$$\mathbf{h}_k = (\alpha_k^{-1} \cdot \bar{\mathbf{\Lambda}}_\mathbf{B}^{-1} \mathbf{\Pi}_\mathbf{B}^{\mathsf{T}}) \cdot \tilde{\mathbf{h}}_k, \tag{10}$$

where $\mathbf{\Lambda_A}$, $\mathbf{\Lambda_B}$, $\bar{\mathbf{\Lambda}}_\mathbf{A}$ and $\bar{\mathbf{\Lambda}}_\mathbf{B}$ are diagonal matrices, $\mathbf{\Pi_A}$ and $\mathbf{\Pi_B}$ are permutation matrices, and α_k are nonzero scalars. The distinction between the CPD and PT ambiguities lies in the slice-wise ambiguities (coefficients α_k), making the ambiguity management task harder than the CPD case. However, we will show how to handle this later.

3. Polynomial functions decoupling

The concept of decoupling is widely recognized in the field of mathematics, particularly in the realm of polynomial functions. Classically, the decoupling problem refers to the challenge of expressing a multivariate polynomial function as a linear combination of univariate polynomials in terms of the input variables. In this paper, we take the classical decoupling problem one step further and generalize the representation to a two-layer model. By doing so, we aim to enhance the versatility and expressive power of the decoupling technique and extend its range of applications.

3.1. Reminder: one-layer structure

Let $\mathbf{f} : \mathbb{R}^m \to \mathbb{R}^n$ be a multivariate polynomial map, with

$$\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}) \cdots f_n(\mathbf{x})]^{\mathsf{T}}, \tag{11}$$

$$\text{and} \qquad \mathbf{x} = [x_1 \cdots x_m]^{\mathsf{T}}. \tag{12}$$

It is said that $\mathbf{f}$ has a decoupled representation, if we have

$$\mathbf{f}(\mathbf{x}) = \mathbf{W}\mathbf{g}(\mathbf{V}^{\mathsf{T}}\mathbf{x}), \tag{13}$$

where $\mathbf{W} \in \mathbb{R}^{n\times r}$, $\mathbf{V} \in \mathbb{R}^{m\times s}$ are transformation matrices,$\mathbf{w}_k$ and $\mathbf{v}_k$ and are respectively their columns, and $\mathbf{g} : \mathbb{R}^r \to \mathbb{R}^r$ follows

$$\mathbf{g}(z_1, \cdots, z_r) = \begin{bmatrix} g_1(z_1) & \cdots & g_r(z_r) \end{bmatrix}^\mathsf{T}. \tag{14}$$

with $g_k : \mathbb{R} \to \mathbb{R}$ is a univariate function. In Fig. 1, we give a graphic representation of the decoupled representation in (13). We can see that this decomposition

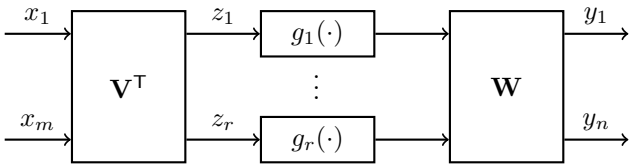

Figure 1: Decoupled representation of $\mathbf{f}$ as in (13).

is composed of a single block/layer, containing a transformation matrix $\mathbf{V}$ and a set of univariate functions g_k, followed by a second transformation matrix $\mathbf{W}$.

3.2. Proposed two-layer structure

In this paper, we propose to extend the decoupled representation in (13) to a representation with two blocks/layers as follows:

$$\mathbf{f}(\mathbf{x}) = \mathbf{W}\mathbf{g}\big(\mathbf{V}^\mathsf{T} \cdot \mathbf{h}(\mathbf{U}^\mathsf{T}\mathbf{x})\big), \tag{15}$$

where $\mathbf{W} \in \mathbb{R}^{n\times r}$, $\mathbf{V} \in \mathbb{R}^{s\times r}$, $\mathbf{U} \in \mathbb{R}^{m\times r}$, are transformation matrices, $\mathbf{h} : \mathbb{R}^s \to \mathbb{R}^s$ and $\mathbf{g} : \mathbb{R}^r \to \mathbb{R}^r$ follow

$$\mathbf{g}(z_1, \cdots, z_r) = [g_1(z_1) \cdots g_r(z_r)]^\mathsf{T}, \tag{16}$$

and

$$\mathbf{h}(t_1, \cdots, t_s) = [h_1(t_1) \cdots h_s(t_s)]^\mathsf{T}. \tag{17}$$

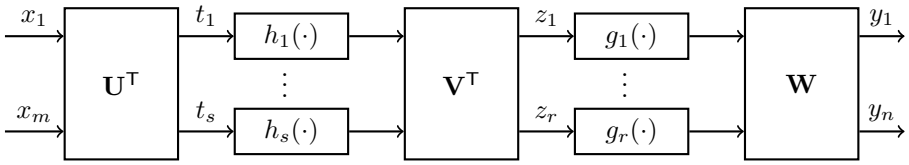

Figure 2: Decoupling representation in (15).

Graphically, the decoupled representation in (15) is given in Fig. 2. We can see that this decomposition has two blocks/layers compared to Fig. 1. This generalization allows to have more flexibility in the decoupling of multivariate

nonlinear functions. In theory, a single layer network may suffice for approximating continuous functions. However, in practical settings, not all functions are continuous and even for those that are, one-layer networks may struggle to fit and generalize well for complex functions. Additionally, a two-layer network has a larger capacity for representation, allowing it to store more information. Intuitively, the increased complexity of a two-layer network leads to improved expressiveness, approximation, generalization, and representation compared to a one-layer network.

4. Tensor-based function decomposition

4.1. Jacobian and paratuck decomposition

The main idea to find the decomposition (15) of a nonlinear function $\mathbf{f}$ relies on the evaluation of the jacobian matrix in different points $\mathbf{x}^{(p)}$, for $p = 1, \dots, P$. This idea mirrors [8], where it has been applied for the classical decoupling model. In the sequel, we will replicate the procedure with the new proposed structure in (15), and will derive the new expression of the Jacobian tensor.

Lemma 1. *The first-order derivatives of the parameterization* (15) *are given by*

$$\mathbf{J_f}(\mathbf{x}) := \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{x}) & \cdots & \frac{\partial f_1}{\partial x_m}(\mathbf{x}) \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1}(\mathbf{x}) & \cdots & \frac{\partial f_n}{\partial x_m}(\mathbf{x}) \end{bmatrix} \tag{18}$$

$$= \mathbf{W} \cdot \operatorname{diag}\Big(\big[g_1'(z_1) \cdots g_r'(z_r)\big]\Big) \cdot \mathbf{V}^\mathsf{T} \cdot \operatorname{diag}\Big(\big[h_1'(t_1) \cdots h_s'(t_s)\big]\Big) \cdot \mathbf{U}^\mathsf{T}. \tag{19}$$

Proof. The proof follows by applying the chain rule to (15).

Based on Lemma 1, we can see that jacobian of (15) evaluated at the points $\mathbf{x}^{(p)}$ follows

$$\mathbf{J_f}(\mathbf{x}^{(p)}) = \mathbf{W} \cdot diag\Big(\mathbf{g}'(\mathbf{z}^{(p)})\Big) \cdot \mathbf{V}^\mathsf{T} \cdot diag\Big(\mathbf{h}'(\mathbf{t}^{(p)})\Big) \cdot \mathbf{U}^\mathsf{T}, \tag{20}$$

$$= \mathbf{W} \cdot \mathbf{D}_\mathbf{G}^p \cdot \mathbf{V}^\mathsf{T} \cdot \mathbf{D}_\mathbf{H}^p \cdot \mathbf{U}^\mathsf{T}, \tag{21}$$

where $\mathbf{t}^{(p)} = \begin{bmatrix} t_1^{(p)} & \cdots & t_s^{(p)} \end{bmatrix} = \mathbf{U}^\mathsf{T}\mathbf{x}^{(p)}$, $\mathbf{z}^{(p)} = \begin{bmatrix} z_1^{(p)} & \cdots & z_r^{(p)} \end{bmatrix} = \mathbf{V}^\mathsf{T}\mathbf{h}(\mathbf{U}^\mathsf{T}\mathbf{x}^{(p)})$, and $\mathbf{D}_\mathbf{H}^p \in \mathbb{R}^{s\times s}$ and $\mathbf{D}_\mathbf{G}^p \in \mathbb{R}^{r\times r}$ are the diagonal matrices given by

$$\mathbf{D}_\mathbf{H}^p = \operatorname{diag}\Big(\big[g_1'(z_1^{(p)}) \cdots g_r'(z_r^{(p)})\big]\Big),$$

$$\mathbf{D}_\mathbf{G}^p = \operatorname{diag}\Big(\big[h_1'(t_1^{(p)}) \cdots h_s'(t_s^{(p)})\big]\Big).$$

We can then define the matrices $\mathbf{H} \in \mathbb{R}^{s\times P}$ and $\mathbf{G} \in \mathbb{R}^{r\times P}$, for $p = 1, \cdots, P$, as

$$\mathbf{H}_{:,p} = \operatorname{diag}\Big(\mathbf{D}_\mathbf{H}^p\Big), \quad \mathbf{G}_{:,p} = \operatorname{diag}\Big(\mathbf{D}_\mathbf{G}^p\Big),$$

or equivalently by their entries as

$$H_{jp} = h'_j(t_j^{(p)}), \quad G_{ip} = g'_i(z_i^{(p)}).$$

This result shows that expression of the jacobian of the new model corresponds to the frontal slices of a PT decomposition of rank (r, s) with factors $\mathbf{U}^\mathsf{T}$, $\mathbf{V}^\mathsf{T}$ and $\mathbf{W}$. It is worth noting that the factors $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{W}$ do not depend on the choice of the point $\mathbf{x}^{(p)}$. Following Lemma 1, a jacobian tensor $\mathcal{J}$ of size $n \times m \times P$ is constructed by stacking the jacobian evaluation at P different sampling points $\mathbf{x}^{(p)} \in \mathbb{R}^m$, for $p = 1, \cdots, P$, where

$$\mathcal{J}_{:,:,p} = \mathbf{J_f}(\mathbf{x}^{(p)}), \tag{22}$$

therefore, tensor $\mathcal{J}$ admits a paratuck decomposition.

4.2. Second-order information and structured CPD

To improve the usefulness of the paratuck formulation, we will examine the second-order information of (15), and show later how this can help in the decomposition, since the paratuck decomposition lacks a reliable algorithm for the moment. In this subsection, we derive an expression for the Hessian tensor at each point. The Hessian tensor $\mathcal{T}(\mathbf{x}) \in \mathbb{R}^{n\times m\times m}$ at a point $\mathbf{x}$ is defined as follows.

$$\mathcal{T}_{ijk}(\mathbf{x}) = \frac{\partial^2 f_i}{\partial x_j \partial x_k}(\mathbf{x}).$$

Next, we show that the Hessian tensor has a CP decomposition. But before that, we introduce some extra notation for the factors of the jacobians introduced in the previous subsection. We denote the matrices $\mathbf{A}(\mathbf{x}) \in \mathbb{R}^{n\times s}$ and $\mathbf{B}(\mathbf{x}) \in \mathbb{R}^{r\times m}$ such that

$$\mathbf{A}(\mathbf{x}) = \mathbf{W} \cdot diag\Big(\mathbf{g}'(\mathbf{z}(\mathbf{x}))\Big) \cdot \mathbf{V}^\mathsf{T}, \tag{23}$$

$$\mathbf{B}(\mathbf{x}) = \mathbf{V}^\mathsf{T} \cdot diag\Big(\mathbf{h}'(\mathbf{t}(\mathbf{x}))\Big) \cdot \mathbf{U}^\mathsf{T}; \tag{24}$$

so that with this notation, we can reformulate (20) as

$$\mathbf{J_f}(\mathbf{x}) = \mathbf{A}(\mathbf{x}) diag\Big(\mathbf{h}'(\mathbf{t}(\mathbf{x}))\Big) \cdot \mathbf{U}^\mathsf{T} = \mathbf{W} \cdot diag\Big(\mathbf{g}'(\mathbf{z}(\mathbf{x}))\Big)\mathbf{B}(\mathbf{x}). \tag{25}$$

Armed with this notation, we are ready to formulate the following result on the structure of the Hessian tensor.

Lemma 2. *The Hessian tensor has the following rank $(r + s)$ CP (polyadic) decomposition:*

$$\mathcal{T}(\mathbf{x}) = [\![\mathbf{g}''(\mathbf{z}(\mathbf{x}))^\mathsf{T};\, \mathbf{W}, \mathbf{B}^\mathsf{T}(\mathbf{x}), \mathbf{B}^\mathsf{T}(\mathbf{x})]\!] + [\![\mathbf{h}''(\mathbf{t}(\mathbf{x}))^\mathsf{T};\, \mathbf{A}(\mathbf{x}), \mathbf{U}, \mathbf{U}]\!]. \tag{26}$$

Proof. The proof follows by applying the Leibniz rule to one of the formulations in (25).

It is worth noting that (i) the Hessian tensor is partially symmetric, *i.e.*, $\mathcal{T}_{ijk} = \mathcal{T}_{ikj}$, and (ii) the rank is too high so that the decomposition cannot be obtained from a CPD due to loss of uniqueness.

5. A constrained coupled tensors decomposition approach

In this section, we propose to tackle the previously mentioned problems by formulating the new decoupling problem as a constrained coupled tensors decomposition, using both the first and second-order information. Before that, we specify the assumptions considered in our approach:

1. $m \geq r$ and $n \geq s$,
2. $\mathbf{W}$ is known and has full column rank r,
3. $\mathbf{U}$ has full column rank s and $\text{unfold}_2\mathcal{J}$ is also of rank s.

Under the aforementioned conditions, we can always reduce the problem to the following case: $n = r$, $m = s$ and $\mathbf{W} = \mathbf{I}_r$. It is important to mention that (i) despite the considered assumptions, the PT decomposition remains a challenging problem that cannot be solved by ALS-type algorithms, such as the one proposed in [4], and (ii) these assumptions are not overly restrictive and can easily be met in practical applications, especially in the training of neural networks, such as autoencoders [19].

5.1. Reformulation as a constrained CPD

We assume that we are given both jacobians and hessians $\mathbf{J_f}(\mathbf{x}^{(p)})$, and $\mathcal{T}(\mathbf{x}^{(p)})$ at P evaluation points. Additionally, we impose that all matrices $\mathbf{D}^p_\mathbf{G}$ are nonsingular to simplify the formulation of the problem. Based on (25), we remark that the matrices $\mathbf{B}(\mathbf{x}^{(p)})$ can be expressed through the known jacobians as

$$\mathbf{B}(\mathbf{x}^{(p)}) = (\mathbf{D}^p_\mathbf{G})^{-1}\mathbf{J}^{(p)}, \tag{27}$$

where $\mathbf{J}^{(p)} = \mathbf{J_f}(\mathbf{x}^{(p)})$. In the same way, and based on (23), matrix $\mathbf{A}(\mathbf{x}^{(p)})$ can be expressed as

$$\mathbf{A}(\mathbf{x}^{(p)}) = \mathbf{D}^p_\mathbf{G}\mathbf{V}^\mathsf{T}. \tag{28}$$

Substituting (27) and (28) in (26), the hessian at the p-th sampling point has the following decomposition.

$$\mathcal{T}(\mathbf{x}^{(p)}) = [\![((\mathbf{D}^p_\mathbf{G})^{-2}\mathbf{g}''(\mathbf{z}(\mathbf{x}^{(p)})))^\mathsf{T}; \mathbf{I}_r, (\mathbf{J}^{(p)})^\mathsf{T}, (\mathbf{J}^{(p)})^\mathsf{T}]\!] + [\![\mathbf{h}''(\mathbf{t}(\mathbf{x}^{(p)})); \mathbf{D}^p_\mathbf{G}\mathbf{V}^\mathsf{T}, \mathbf{U}, \mathbf{U}]\!].$$

In what follows, we take all the hessians at P points and stack them into the following third-order $Pn \times m \times m$ tensor $\mathcal{T}^{all}$:

$$(\mathcal{T}^{all})_{1+(p-1)n:pn,:,:} = \mathcal{T}(\mathbf{x}^{(p)}).$$

We also do the same for the jacobians, which we stack into Jacobian matrix $\mathbf{J}^{all} \in \mathbb{R}^{Pn\times m}$:

$$\mathbf{J}^{all}_{1+(p-1)n:pn,:} = \mathbf{J_f}(\mathbf{x}^{(p)}).$$

Then, from the previous derivations, the tensor had the following CPD with structured factors:

$$\mathcal{T}^{all} = [\![diag(\mathbf{c}), (\mathbf{J}^{all})^\mathsf{T}, (\mathbf{J}^{all})^\mathsf{T}]\!] + [\![\mathbf{E}, \mathbf{U}, \mathbf{U}]\!] \tag{29}$$

where

$$\mathbf{c} = \begin{bmatrix} (\mathbf{D}_{\mathbf{G}}^{1})^{-2}\mathbf{g}''(\mathbf{z}(\mathbf{x}^{(1)})) \\ \vdots \\ (\mathbf{D}_{\mathbf{G}}^{P})^{-2}\mathbf{g}''(\mathbf{z}(\mathbf{x}^{(p)})) \end{bmatrix}, \quad \mathbf{E} = \begin{bmatrix} (\mathbf{D}_{\mathbf{G}}^{1})\mathbf{V}^{\mathsf{T}} diag(\mathbf{h}''(\mathbf{t}(\mathbf{x}^{(1)}))) \\ \vdots \\ (\mathbf{D}_{\mathbf{G}}^{P})\mathbf{V}^{\mathsf{T}} diag(\mathbf{h}''(\mathbf{t}(\mathbf{x}^{(p)}))) \end{bmatrix}.$$

We see that the latter problem in (29) is a CPD where the first term has two known factors $\mathbf{J}^{all}$ and one diagonal factor, and a second term with unknown factors. Note that the second CPD has very low-rank s and can be retrieved with matrix methods as we explain in the next subsection.

5.2. Reformulation as structured low-rank matrix completion

Instead of the tensor $\mathcal{T}^{all}$, we consider its transposed first unfolding $\mathbf{T}^{all} \in \mathbb{R}^{m^2 \times nP}$, which has the following factorization.

$$\mathbf{T}^{all} = \left((\mathbf{J}^{all})^{\mathsf{T}} \odot (\mathbf{J}^{all})^{\mathsf{T}}\right) \operatorname{diag}(\mathbf{c}) + (\mathbf{U} \odot \mathbf{U})\mathbf{E}^{\mathsf{T}}.$$

Assume we are in the exact case, then we just need to find vector $\mathbf{c}$ so that the matrix

$$\mathscr{S}(\mathbf{c}) = \mathbf{T}^{all} - \left((\mathbf{J}^{all})^{\mathsf{T}} \odot (\mathbf{J}^{all})^{\mathsf{T}}\right) \operatorname{diag}(\mathbf{c})$$

has rank s. We propose to pose this problem as rank minimization over the set of structured matrices $\mathscr{S}(\mathbf{c})$, which can be solve as the following minimization problem over the low-rank manifold:

$$\min_{PL} \|\Pi_{\mathscr{S}}(PL - \mathbf{T}^{all})\|_F, \tag{30}$$

where $\Pi_{\mathscr{S}}$ denotes the projection on the set of structured matrices.

5.3. Ambiguities in the problem and recovering the functions

After estimation of $\mathbf{U}$, the matrices $\mathbf{V}$, $\mathbf{G}$ and $\mathbf{H}$ can be easily found, as noted before. The problem which remains for decoupling approach is the reconstruction of functions. The issue is that even plotting the $\mathbf{H}_{k,:}$ with versus

$$(t_1, \ldots, t_P) = (\mathbf{u}^{\mathsf{T}}\mathbf{x}^{(1)}, \ldots, \mathbf{u}^{\mathsf{T}}\mathbf{x}^{(P)})$$

does not work, as in the CPD case, see Fig. 5.3.

This is due to the presence of the ambiguities α_p for each of the columns of $\mathbf{G}$. In order to recover nice plots of functions, we need more assumptions (for example, impose that g_k can be polynomials of low order. To estimate the ambiguities, we need to solve the following system of equations

$$\mathbf{a}_k^{\mathsf{T}}\mathbf{X}(\mathbf{t}, d) = \mathbf{H}_{k,:} \operatorname{diag}(\alpha), \quad k = 1, \ldots, s$$

where d is the degree of the polynomial $\mathbf{X}(\mathbf{t}, d)$ is the Vandermonde matrix (for points t and up to degree d), and we solve for $\mathbf{a}_k^{\mathsf{T}}$ (coefficients of polynomials) and α (scalings). This is a problem of intersection of linear subspaces and can be solved with alternating least squares, see Fig. 5.3.

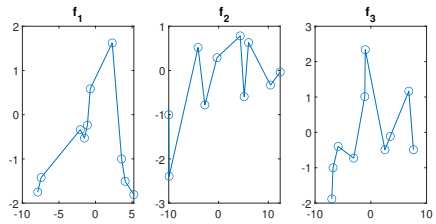

Figure 3: Estimation of activation function without correction for slice ambiguities.

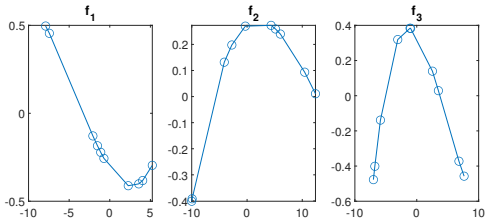

Figure 4: Estimation of activation function with correction for slice ambiguities.

6. Conclusion

In conclusion, this paper presents a new method for multivariate function approximation that combines paratuck and CP decompositions in a coupled manner. Our approach utilizes both first and second-order information of the original function and has been shown to be effective through numerical simulations on a simple synthetic example. Although the paratuck decomposition remains a challenging problem, our results demonstrate the potential of the proposed method for addressing this issue and provide a promising direction for future work in the field of multivariate nonlinear function approximation.

References

[1] R. André, X. Luciani, and E. Moreau. Joint eigenvalue decomposition algorithms based on first-order taylor expansion. *IEEE Transactions on Signal Processing*, 68:1716–1727, 2020. doi: 10.1109/TSP.2020.2976580.

[2] A. Apicella, F. Donnarumma, F. Isgrò, and R. Prevete. A survey on modern trainable activation functions. *Neural Networks*, 138:14–32, 2021.

[3] R. Bro. Parafac. tutorial and applications. *Chemometrics and Intelligent Laboratory Systems*, 38(2):149–171, 1997.

[4] R. Bro. Multi-way analysis in the food industry models, algorithms & applications. 1998.

[5] F.-C. Chen. Back-propagation neural networks for nonlinear self-tuning adaptive control. *IEEE Control Systems Magazine*, 10(3):44–48, 1990. doi: 10.1109/37.55123.

[6] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 160–167, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605582054. doi: 10.1145/1390156.1390177. URL https://doi.org/10.1145/1390156.1390177.

[7] G. Cybenko. Continuous valued neural networks with two hidden layers are sufficient, department of computer science. *Trfts. University*, 1988.

[8] P. Dreesen, M. Ishteva, and J. Schoukens. Decoupling multivariate polynomials using first-order information and tensor decompositions. *SIAM Journal on Matrix Analysis and Applications*, 36(2):864–879, 2015.

[9] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In G. Gordon, D. Dunson, and M. Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL http://proceedings.mlr.press/v15/glorot11a.html.

[10] R. Harshman and M. Lundy. Uniqueness proof for a family of models sharing features of Tucker's three-mode factor analysis and PARAFAC/candecomp. *Psychometrika*, 61(1):133–154, March 1996. doi: 10.1007/BF02296963. URL https://ideas.repec.org/a/spr/psycho/v61y1996i1p133-154.html.

[11] R. A. Harshman. Foundations of the parafac procedure: Models and conditions for an 'explanatory' multi-modal factor analysis. *UCLA Working Papers in Phonetics*, 16:1–84, 1970.

[12] R. A. Harshman and M. E. Lundy. Parafac: Parallel factor analysis. *Computational Statistics & Data Analysis*, 18(1):39–72, 1994. ISSN 0167-9473. doi: https://doi.org/10.1016/0167-9473(94)90132-5. URL https://www.sciencedirect.com/science/article/pii/0167947394901325.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, page 1026–1034, USA, 2015. IEEE Computer Society. ISBN 9781467383912. doi: 10.1109/ICCV.2015.123. URL https://doi.org/10.1109/ICCV.2015.123.

[14] F. L. Hitchcock. Multiple invariants and generalized rank of a p-way matrix or tensor. *Journal of Mathematics and Physics*, 7:39–79, 1927.

[15] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. ISSN 0893-6080. doi: https://doi.org/10.1016/0893-6080(89)90020-8. URL https://www.sciencedirect.com/science/article/pii/0893608089900208.

[16] S. Khan, H. Rahmani, S. Shah, and M. Bennamoun. *A Guide to Convolutional Neural Networks for Computer Vision*. Number 1 in Synthesis Lectures on Computer Vision. 2018. doi: 10.2200/S00822ED1V01Y201712COV015.

[17] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[18] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. doi: 10.1038/nature14539. URL https://doi.org/10.1038/nature14539.

[19] C.-Y. Liou, W.-C. Cheng, J.-W. Liou, and D.-R. Liou. Autoencoder for words. *Neurocomputing*, 139:84–96, 2014. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2013.09.055. URL https://www.sciencedirect.com/science/article/pii/S0925231214003658.

[20] A. V. Mulders, L. Vanbeylen, and K. Usevich. Identification of a block-structured model with several sources of nonlinearity. In *European Control Conference (ECC)*, Strasbourg, France, 2014.

[21] S. Narayan. The generalized sigmoid activation function: Competitive supervised learning. *Information Sciences*, 99(1):69–82, 1997. ISSN 0020-0255. doi: https://doi.org/10.1016/S0020-0255(96)00200-9. URL https://www.sciencedirect.com/science/article/pii/S0020025596002009.

[22] S. Pfrommer, M. Halm, and M. Posa. Contactnets: Learning discontinuous contact dynamics with smooth, implicit representations. In J. Kober, F. Ramos, and C. Tomlin, editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 2279–2291. PMLR, 16–18 Nov 2021. URL https://proceedings.mlr.press/v155/pfrommer21a.html.

[23] P. Ramachandran, B. Zoph, and Q. V. Le. Searching for activation functions. In *6th International Conference on Learning Representations, ICLR*, 2018.

[24] M. Schoukens and Y. Rolain. Cross-term elimination in parallel Wiener systems using a linear input transformation. *IEEE Transactions on Instrumentation and Measurement*, 61:845–847, 2012.

[25] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos. Tensor decomposition for signal processing and machine learning. *IEEE Transactions on Signal Processing*, 65(13):3551–3582, 2017.

[26] A. E. SMITH and A. K. MASON. Cost estimation predictive modeling: Regression versus neural network. *The Engineering Economist*, 42(2):137–161, 1997. doi: 10.1080/00137919708903174. URL https://doi.org/10.1080/00137919708903174.

[27] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[28] L. R. Ximenes, G. Favier, A. L. F. de Almeida, and Y. C. B. Silva. Parafac-paratuck semi-blind receivers for two-hop cooperative mimo relay systems. *IEEE Transactions on Signal Processing*, 62(14):3604–3615, 2014. doi: 10.1109/TSP.2014.2328323.

[29] S. İçer and Şerife Gengeç. Classification and analysis of non-stationary characteristics of crackle and rhonchus lung adventitious sounds. *Digital Signal Processing*, 28:18–27, 2014. ISSN 1051-2004. doi: https://doi.org/10.1016/j.dsp.2014.02.001. URL https://www.sciencedirect.com/science/article/pii/S1051200414000384.