# Efficient Linearization for Explicit Multilinear Models in Tensor Train Representation*

Jithin Cherian[1] and Gerwald Lichtenberg[2]

***Abstract*— This paper presents a novel linearization method for tensor train (TT) based explicit multilinear time-invariant (eMTI) models. The proposed algorithm enables a fast computation of the Jacobian matrix to obtain the linearized model. The proposed method is compared to the standard finite differences method by their computation times for different model sizes. The linearization technique can be used e.g. for multilinear model-predictive control (MMPC) to cope with real-time requirements, or for small-signal analysis, where efficient linearization algorithms have been available only for canonical polyadic (CP) tensor decompositions so far.**

***Keywords:*** **Multilinear Models, Tensor Train Decomposition, Linearization**

## I. INTRODUCTION

Linearization of nonlinear models is used in many engineering tasks for the local approximations of complex system dynamics, to apply well-established methods from linear system theory. Thus, performant linearization algorithms for large-scale nonlinear systems are of interest for various applications. Conventional numeric methods, for e.g. finite differences, work for all kinds of nonlinearties, but are not as efficient as symbolic methods based on, for e.g. computer algebra or automatic differentiation. While these methods are not restricted to special classes of nonlinearities, many applications, such as HVAC systems, exist where the main phenomena can be described by multilinear polynomials.

For representing the nonlinear thermal dynamics of buildings, a suitable approach, hence, is the multilinear time-invariant (MTI) framework, which was first introduced in explicit format (eMTI) [1]. Many application driven methods have been developed since for parameter identification, e.g. the implicit (iMTI) formulation [2]. Compared to linear models, MTI models can represent a wider class of nonlinear dynamical systems [3]. Moreover, due to their inherent structural properties and tensor-based representations, they enable the use of efficient tensor decomposition methods such as the Canonical Polyadic (CP) and the Tensor Train (TT) decompostions among others [4], [5]. For more information on the tensor decomposition-based multilinear modelling, see [6].

MTI state space models are a subclass of nonlinear models where the right hand sides (RHS) are multilinear functions of the states and inputs, i.e. polynomials with maximum exponents of one in their monomials. Recall that multilinear functions become affine, if all but one variable are held constant. As the parameters of MTI models are tensors, the nonlinear dynamics of large-scale complex systems can be represented in various decomposed tensor formats.

The TT decomposition method, in particular, is widely known for its computational advantages. It helps to compute the parameter tensor of an eMTI model with the help of the pseudoinverse of the data tensor. Evaluation of the data tensor is done with the help of basis functions from measurement data as proposed in [7] with tools in [8], thus paving the way for data-driven multilinear parameter identification. After parameter identification, a fundamental approach in the multilinear system optimization framework is the linearization of the system model which in turn supports linear system theory for various optimal control methods. The motivation to develop algorithms for efficient computation of the Jacobian matrix contributing to symbolic linearization of eMTI models in TT representation is application-driven. Thus, comparing its performance to the conventional finite differences method is of interest for implementation in real-world HVAC systems.

The paper is organized as follows: Section II describes the eMTI model and the TT decomposition method in detail. Section III focuses on the proposed linearization algorithm and illustrates the algorithm using a simple example, while Section IV demonstrates its performance and accuarcy by simulation results. Final conclusions are drawn in Section V.

## II. TT REPRESENTATION OF MTI MODELS

This section introduces eMTI models with their parameter tensors, the tensor train decomposition, and the usage of TT decomposed tensors as parameters of eMTI models.

### A. Tensors

Tensors are basically multidimensional array structures over a field $\mathbb{R}$ represented by $\mathsf{T} \in \mathbb{R}^{N_1 \times N_2 \times \cdots \times N_d}$ where $d$ is the order of the tensor $\mathsf{T}$ and the tuple $(N_1, N_2, \cdots, N_d)$ represents its dimensions (or mode sizes). For visualization purposes, a third-order tensor is shown in Figure 1, as higher-order tensors cannot be represented adequately in two dimensions.

A computational disadvantage of higher order tensors is the curse of dimensionality, because the number of elements of a tensor grows exponentially with its order, thus, requiring exponentially growing memory if tensors are stored element-wise. This can be overcome by tensor decomposition meth-

*This paper is funded by the Federal Ministry of Research, Technology and Space (MARGE project) and the Federal Ministry for Economic Affairs and Energy (ADAPTER project) in Germany

[1]Jithin Cherian is with the Faculty of Life Sciences, University of Applied Sciences, Hamburg jithin.cherian@haw-hamburg.de

[2]Gerwald Lichtenberg is with the Faculty of Life Sciences, University of Applied Sciences, Hamburg gerwald.lichtenberg@haw-hamburg.de

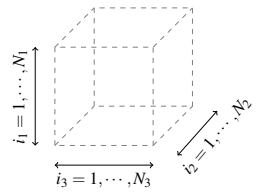


Fig. 1. Third-order tensor $\mathsf{B} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$.

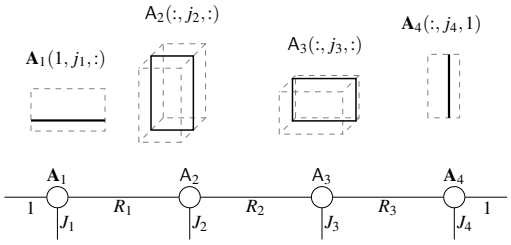


Fig. 2. TT Decomposition of a $4^{th}$ order tensor $\mathsf{A}$

ods - similar to the singular value decomposition (SVD), which enables reductions or approximations, expecially for rank-deficient matrices.

Whereas other decomposition methods like CP and Tucker are also possible, we focus in this paper on Tensor Trains, which are known to have advantages if approximations of an original tensor have to be found with a defined measure of accuracy, [5] .

### B. Multilinear Models

For the state vector $\mathbf{x} \in \mathbb{R}^n$, input vector $\mathbf{u} \in \mathbb{R}^m$, the monomial tensor

$$\mathsf{M}(\mathbf{x},\mathbf{u}) = \begin{pmatrix}1\\ x_1\end{pmatrix} \circ \begin{pmatrix}1\\ x_2\end{pmatrix} \circ \cdots \circ \begin{pmatrix}1\\ x_n\end{pmatrix} \circ \begin{pmatrix}1\\ u_1\end{pmatrix} \circ \cdots \circ \begin{pmatrix}1\\ u_m\end{pmatrix}, \quad (1)$$

of dimension $\mathbb{R}^{\overbrace{2 \times 2 \times \cdots \times 2}^{(n+m)}} \equiv \mathbb{R}^{\times^{(n+m)}2}$ is defined by the outer product (denoted by $\circ$). This is used in the contracted products $\langle | \rangle$ of a continuous-time eMTI model [3]

$$\begin{aligned} \dot{\mathbf{x}} &= \langle \mathsf{F} \mid \mathsf{M}(\mathbf{x},\mathbf{u}) \rangle \,, \\ \mathbf{y} &= \langle \mathsf{G} \mid \mathsf{M}(\mathbf{x},\mathbf{u}) \rangle \,, \end{aligned} \quad (2)$$

to define the vector $\dot{\mathbf{x}}$ of state derivatives and the output vector $\mathbf{y} \in \mathbb{R}^p$ by the transition tensor $\mathsf{F} \in \mathbb{R}^{\times^{(n+m)}2 \times n}$ and the output tensor $\mathsf{G} \in \mathbb{R}^{\times^{(n+m)}2 \times p}$. More information about tensor operations such as the contracted product $\langle | \rangle$ is given e.,g. in [9]. The state and output parameter tensors share the same first $n+m$ dimensions with the monomial tensor but have one more dimension of the size corresponding to the signal, such that their contracted products are computed as sums of the element-wise products resulting in vectors of dimensions $n$ for $\dot{\mathbf{x}}$ or $p$ for $\mathbf{y}$, respectively.

### C. Tensor Train Decomposition

A Tensor Train (TT) decomposition approximates a tensor of order $d$ by a product of $d+1$ third order tensors, known as *TT-cores*, where the first and the last TT-cores have a degenerated dimension of 1 and thus can be represented as matrices. The middle dimensions connecting the core tensors are known as the *TT-ranks* of the decomposition and if it is exact, they are the TT-ranks of the original tensor.

The visual and graphical representations of a fourth order TT decomposed tensor is shown in Figure 2 where each TT-core $\mathsf{A}_i \in \mathbb{R}^{R_{k-1} \times J_k \times R_k}$ is represented as dotted lines and $i = 1,2,3,4$, and the lower part of the figure is called the tensor network graph. Here, $R_k$ represents the TT-ranks for $k = 1,2,3$. From Figure 2, it can be seen that the first and the last TT-cores, $\mathbf{A_1}$ and $\mathbf{A_4}$, are matrices while all other cores - here $\mathsf{A}_2, \mathsf{A}_3$ are 3D tensors.

Also, each element in the TT decomposed tensor can be calculated by a series of matrix multiplications between the slices of the TT-cores. For example, the elements of $\mathsf{A}$ can be evaluated by the combined product between a row vector, two matrices and a column vector which are represented as thick lines and indexed accordingly in Figure 2. The TT decomposition format also benefits from SVD-based truncation, while avoiding the exponential growth in memory requirements which highlights its potential for eMTI modelling. For more information on the global SVD of TT decomposed tensors and other TT decomposition operations, see [7], [9], [10].

*Definition 1 (TT tensor):* The decomposed tensor

$$\mathsf{A} = [\![ \mathbf{A}_1, \mathsf{A}_2, \cdots, \mathbf{A}_d ]\!] \quad (3)$$

is given by its cores in the brackets $[\![, \ldots, ]\!]$, where $\mathsf{A}_1$ and $\mathsf{A}_d$ are matrices, $(\mathsf{A}_2, \cdots, \mathsf{A}_{(d-1)})$ are 3D tensors and $d$ denotes the order of the tensor.

Elements of $\mathsf{A}$ are computed by the so-called contraction

$$a(j_1, j_2, \cdots, j_d) = \mathsf{A}_1(1, j_1, :)\mathsf{A}_2(:, j_2, :) \cdots \mathsf{A}_d(:, j_d, 1) \quad (4)$$

of the TT cores along the TT ranks. This representation can then be used to define the state and output tensors $\mathsf{F}$ and $\mathsf{G}$ of the eMTI model (5) in TT decomposed formats.

*Definition 2 (TT representation of eMTI models):*

$$\begin{aligned} \dot{\mathbf{x}} &= \langle [\![ \mathsf{F}_{x_1}, \ldots, \mathsf{F}_{x_n}, \mathsf{F}_{u_1}, \ldots, \mathsf{F}_{u_m}, \mathbf{F}_\phi ]\!] \mid \mathsf{M}(\mathbf{x},\mathbf{u}) \rangle \,, \\ \mathbf{y} &= \langle [\![ \mathsf{G}_{x_1}, \ldots, \mathsf{G}_{x_n}, \mathsf{G}_{u_1}, \ldots, \mathsf{G}_{u_m}, \mathbf{G}_\phi ]\!] \mid \mathsf{M}(\mathbf{x},\mathbf{u}) \rangle \,, \end{aligned} \quad (5)$$

where each TT core is a 3D tensor - except for the first and the last TT cores which are considered as matrices [10].

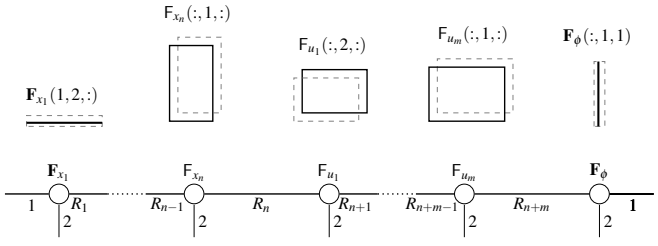


Fig. 3. Parameter tensor $\mathsf{F}$ in TT representation

The structural representation of the $\mathsf{F}$ tensor in TT format is shown in Figure 3 - similar to the one of $\mathsf{G}$.

In Figure 3, it is visualized, how a tensor element $F(2, \ldots, 1, 2, \ldots, 1, 1)$ is computed by vector times

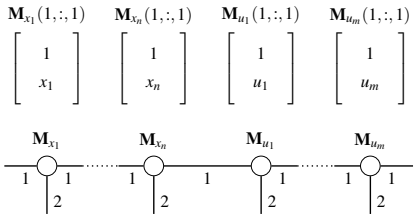

Fig. 4. Monomial tensor $\mathsf{M}(\mathbf{x},\mathbf{u})$ in TT representation

$(n+m-1)$ matrix times vector multiplications. Each TT core represents a state or input dimension of the $\mathsf{F}$ and $\mathsf{G}$ tensors.

Similarly, Figure 4 illustrates the TT representation

$$\mathsf{M}(\mathbf{x},\mathbf{u}) = \left[\!\!\left[ \begin{pmatrix}1\\x_1\end{pmatrix}, \cdots, \begin{pmatrix}1\\x_n\end{pmatrix}, \begin{pmatrix}1\\u_1\end{pmatrix}, \cdots, \begin{pmatrix}1\\u_m\end{pmatrix} \right]\!\!\right] \tag{6}$$

of the monomial tensor and its mathematical formulation is shown in (6), where each TT-core is 3D but the two outer dimensions are structurally 1, which enables each core to be essentially seen as a vector with only 2 elements. Moreover, $\mathsf{F}$ has an additional TT-core $\mathbf{F}_\phi$ compared to $\mathsf{M}$ due to the fact that the contracted tensor product $\langle|\rangle$ between the two tensors results in the state derivative vector shown in (5). Additionally, the row dimension of the last TT core of $\mathsf{F}$ is equal to the number of states of the model.

Note that the TT-cores of the parameter tensor $\mathsf{F}$ shown in Figure 3 have one of their dimensions always fixed at 2 compared to the cores of the general TT decomposed tensor illustrated in Figure 2. This ensures that the row dimensions of the TT-cores of $\mathsf{M}$ match with the vectors holding the constant 1 and one of the state or input variables.

Each element of $\mathsf{F}$ can be computed as explained in Definition 1 by selecting one of the row vectors from the first TT-core, one of the matrix slices from the remaining 3D TT-cores and one of the column vectors from the last TT-core which are all represented as thick lines and indexed accordingly in Figure 3. More information on the application of this parameter identification procedure can be found in [11].

The contraction between $\mathsf{F}$ and $\mathsf{M}$ produces, as a result, a vector of dimension $n$, which can be alternatively represented as tensor $\mathsf{H}$. The next proposition shows that the contraction operation can be done on the TT decomposed forms – without expanding them – and is recalled from [9].

*Proposition 1 (Contraction of* $\mathsf{F}$ *and* $\mathsf{M}$ *in TT format):* The contracted product

$$\langle \mathsf{F} \mid \mathsf{M}(\mathbf{x},\mathbf{u}) \rangle = \left[\!\!\left[ \mathbf{H}_{x_1}, \cdots, \mathsf{H}_{x_n}, \mathsf{H}_{u_1}, \cdots, \mathsf{H}_{u_m}, \mathbf{H}_\phi \right]\!\!\right] , \tag{7}$$

results in a vector

$$\begin{aligned} \mathsf{H}_{\mathsf{x}_\mathsf{i}} &= \mathsf{F}_{x_i}(:,1,:) + x_i \cdot \mathsf{F}_{x_i}(:,2,:) \ , \ \forall\, i = 1,\ldots,n \\ \mathsf{H}_{\mathsf{u}_\mathsf{j}} &= \mathsf{F}_{u_j}(:,1,:) + u_j \cdot \mathsf{F}_{u_j}(:,2,:) \ , \ \forall\, j = 1,\ldots,m \\ \mathbf{H}_\phi &= \mathbf{F}_\phi \ , \end{aligned} \tag{8}$$

formally given by a TT decomposition computed from the TT-cores of the tensor $\mathsf{F}$ and the signal vectors.

*Proof*: Standard rules for TT-contraction can be applied to see it directly.

As the middle dimension of all TT-core tensors $\mathsf{H}_{\mathsf{x}_\mathsf{i}}$ and $\mathsf{H}_{\mathsf{u}_\mathsf{j}}$ is one, they all can be represented as matrices. This contraction process is further explained by a simple example.

*Example 1:* Consider an eMTI state space model having two states $(x_1, x_2)$ and one input $u$ and the corresponding TT tensors $\mathsf{F}$ and $\mathsf{M}$ are shown in Figures 5 and 6.

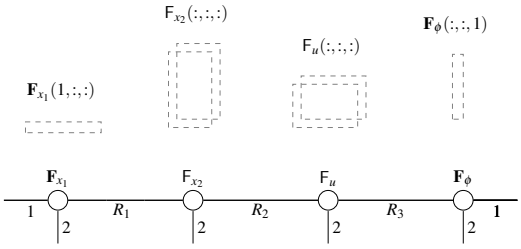

Fig. 5. Representation of the TT parameter tensor $\mathsf{F}$ of Example 1

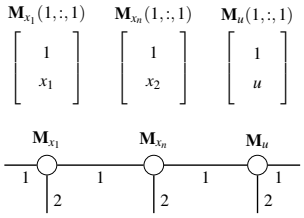

Fig. 6. Representation of the TT monomial tensor $\mathsf{M}(\mathbf{x},\mathbf{u})$ of Example 1

Now, the TT contraction between the first TT cores of $\mathsf{F}$ and $\mathsf{M}$ is executed by the scalar multiplication of the elements 1 and $x_1$ with the corresponding row vectors of $\mathsf{F}_{\mathsf{x}_1}(1,:,:)$ matrix and subsequently adding them up to obtain a resultant TT core which is a row vector.

Similarly, the TT contraction between the second TT cores is implemented by the scalar multiplication of the elements 1 and $x_2$ with the corresponding matrix slices of $\mathsf{F}_{\mathsf{x}_2}(:,:,:)$ tensor and adding them up to produce a resultant TT core which is a matrix. This process continues until the final TT core of $\mathsf{M}$ and the additional $\mathsf{F}_\phi(:,:,1)$ tensor is appended

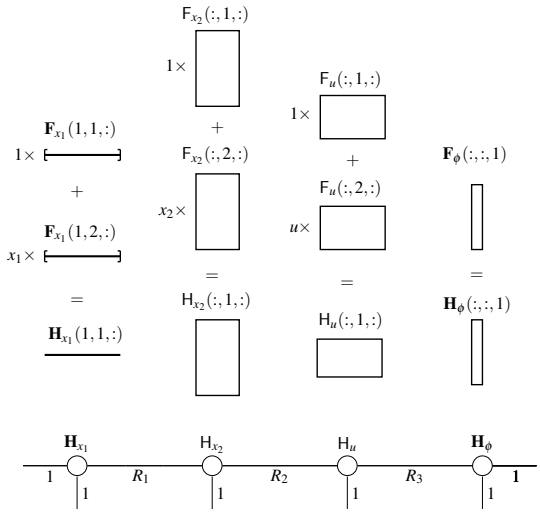

Fig. 7. Representation of the resultant TT contracted tensor $\mathsf{H}$ of Example 1

at the end of the final result. This contraction process is shown in detail in Figure 7 and the mathematical expression of the TT cores of $\mathsf{H}$ is given in (8). Note that the tensor contraction between the TT cores of $\mathsf{H}$ along the TT ranks explained in Proposition 1 provides the state derivative vector $\dot{\mathbf{x}}$ in (5).

In the next section, this is used to elaborate the concept of linearization of eMTI models in TT representation, which reduces computational times compared to standard finite differences methods.

## III. Linearization of TT based eMTI model

The proposed linearization technique produces the Jacobian matrix of the multilinear system in TT representation. Also, within the interior-point optimizer (IPOPT), the Jacobian matrix of the nonlinear constraints, which includes the eMTI model is required to construct the linearized subproblems of the optimal control problem (OCP) at each iteration of the closed loop model simulation as explained in [12].

For a vector function

$$\mathbf{f}(\mathbf{v}) = \begin{bmatrix} f_1(\mathbf{v}) \\ \vdots \\ f_s(\mathbf{v}) \end{bmatrix},$$

dependent on the variables vector

$$\mathbf{v} = \begin{bmatrix} v_1 \\ \vdots \\ v_q \end{bmatrix},$$

the Jacobian matrix is given as

$$\mathbf{J}(\mathbf{v}) = \frac{\partial \mathbf{f}}{\partial \mathbf{v}} = \begin{bmatrix} \frac{\partial f_1}{\partial v_1} & \cdots & \frac{\partial f_1}{\partial v_q} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_s}{\partial v_1} & \cdots & \frac{\partial f_s}{\partial v_q} \end{bmatrix} \tag{9}$$

From Proposition 1 and Example 1, the formulation of a TT contracted tensor can be clearly understood. After obtaining the TT contracted tensor, the structure of the tensor has to be changed with respect to the system variable for computing the Jacobian matrix. In order to establish the novel linearization algorithm involving the Jacobian matrix computation for the TT-based eMTI model, a special type of TT contracted tensor is explained in Definition 3.

*Definition 3 (Specialized TT tensors):* The special class of TT tensors are formed by replacing the row vector or the matrix (depending on the TT core position) in a specified TT core of $\mathsf{H}$ explained in Proposition 1 by the corresponding row vector or matrix slice related to the system variable in $\mathsf{F}$. Also, the nomenclature of this special TT tensor changes by adding a superscript to $\mathsf{H}$ which indicates the respective TT core where the modifications occured. Examples are shown in Figure 8 where the modification occurs at the first TT core of $\mathsf{H}$, therefore, replacing the row vector and in Figure 9 where the modification occurs at the second TT core of $\mathsf{H}$, thereby, substituting the matrix.

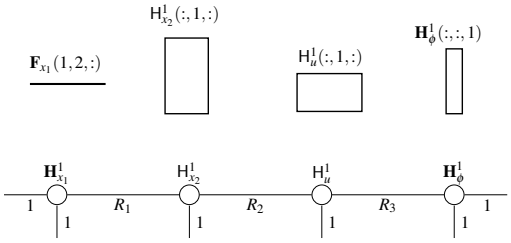


Fig. 8. Representation of special TT contracted tensor $\mathsf{H}^1$ in TT format

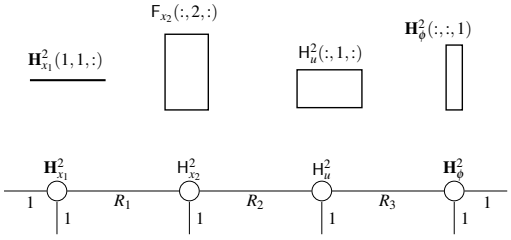


Fig. 9. Representation of special TT contracted tensor $\mathsf{H}^2$ in TT format

*Proposition 2 (Jacobian of the TT based eMTI model):* For an eMTI model with the parameter tensor $\mathsf{F}$ in TT format and the monomial tensor $\mathsf{M}$ shown in Figures 3 and 4, the Jacobian matrix

$$\mathbf{J} = \begin{bmatrix} \mathsf{H}^1 & \cdots & \mathsf{H}^{n+m} \end{bmatrix} = \begin{bmatrix} \mathsf{H}^i \end{bmatrix}_{i=1}^{n+m} \tag{10}$$

can be expressed by the special TT-tensors $(\mathsf{H}^1, \cdots, \mathsf{H}^{n+m})$ as described in Definition 3.

*Proof:* From Proposition 1, the TT contraction between the parameter tensor $\mathsf{F}$ and the monomial tensor $\mathsf{M}$ of the TT based eMTI system can be written as a mathematical operation shown in Eq.(11) where $\mathbf{z} = (z_1 \cdots z_{n+m}) = (\mathbf{x}^T \mathbf{u}^T)^T$.

$$\dot{\mathbf{x}} = \mathbf{F}_{\phi} \cdot (\prod_{i=1}^{n+m} (\mathsf{F}_{z_i}(:,1,:) + \mathsf{F}_{z_i}(:,2,:) z_i))^T \tag{11}$$

We will be constructing the Jacobian vector of $\dot{\mathbf{x}}$ with respect to the individual state $z_j$ where $\mathbf{x} = (z_j)_{j=1}^{n}$. Modifying Eq.(11) by splitting the product from $i = 1$ to $n+m$ in three parts, we get

$$\begin{aligned} \dot{\mathbf{x}} = & \mathbf{F}_{\phi} \cdot ((\prod_{i=1}^{j-1} (\mathsf{F}_{z_i}(:,1,:) + \mathsf{F}_{z_i}(:,2,:) z_i)) \cdot \\ & (\mathsf{F}_{z_j}(:,1,:) + \mathsf{F}_{z_j}(:,2,:) z_j) \cdot \\ & (\prod_{k=j+1}^{n+m} (\mathsf{F}_{z_k}(:,1,:) + \mathsf{F}_{z_k}(:,2,:) z_k)))^T \end{aligned} \tag{12}$$

The Jacobian vector of $\dot{\mathbf{x}}$ with respect to the individual state $z_j$ is given by

$$\begin{aligned} \frac{\partial \dot{\mathbf{x}}}{\partial z_j} &= \frac{\partial}{\partial z_j} \left( \mathbf{F}_{\phi} \cdot (\prod_{i=1}^{n+m} (\mathsf{F}_{z_i}(:,1,:) + \mathsf{F}_{z_i}(:,2,:) z_i))^T \right) \\ &= \mathbf{F}_{\phi} \cdot ((\prod_{i=1}^{j-1} (\mathsf{F}_{z_i}(:,1,:) + \mathsf{F}_{z_i}(:,2,:) z_i)) \cdot \\ & \quad \frac{\partial}{\partial z_j} \left( \mathsf{F}_{z_j}(:,1,:) + \mathsf{F}_{z_j}(:,2,:) z_j \right) \cdot \\ & \quad (\prod_{k=j+1}^{n+m} (\mathsf{F}_{z_k}(:,1,:) + \mathsf{F}_{z_k}(:,2,:) z_k)))^T \end{aligned} \tag{13}$$

The partial derivative of $\dot{\mathbf{x}}$ with respect to $z_j$ gives the final Jacobian vector

$$\begin{aligned}\dot{\mathbf{x}} =& \mathbf{F}_\phi \cdot ((\prod_{i=1}^{j-1}(\mathsf{F}_{z_i}(:,1,:)+\mathsf{F}_{z_i}(:,2,:)z_i))\cdot \\ & \mathsf{F}_{z_j}(:,2,:)\cdot \\ & (\prod_{k=j+1}^{n+m}(\mathsf{F}_{z_k}(:,1,:)+\mathsf{F}_{z_k}(:,2,:)z_k)))^T\end{aligned} \tag{14}$$

Consequently, Eq. (14) corresponds to the specialized TT tensors introduced in Definition 3. Analogues operations to Eq. (14) can be carried out to compute all the partial derivatives of $\dot{\mathbf{x}}$ with respect to the remaining states and inputs leading to the Jacobian matrix $J \in \mathbb{R}^{n\times(n+m)}$. The Jacobian matrix can then be separated into the system matrix and the input matrix for the state space equations evaluated at the operating point $(\bar{\mathbf{x}}, \bar{\mathbf{u}})$.

■

## IV. Example and Simulation Results

Let us analyse proposition 2 with the same example 1. The individual TT cores of the parameter tensor $\mathsf{F}$ and the monomial tensor $\mathsf{M}$ are given in Eq. (15). For simplicity, let us assume that we have sparsity and equal dimensions for the 3D TT cores in $\mathsf{F}$. The expanded state space equations of the model along with its partial derivatives are given in (16).

$$\begin{aligned}\mathbf{F}_{x_1}(1,:,:) &= \begin{bmatrix}0.4 & 0\\ 0 & 0.6\end{bmatrix}\\ \mathbf{F}_{x_2}(:,1,:) &= \begin{bmatrix}0.8 & 0\\ 0.3 & 0.5\end{bmatrix},\ \mathbf{F}_{x_2}(:,2,:) = \begin{bmatrix}0.1 & 0\\ 0 & 0.7\end{bmatrix}\\ \mathbf{F}_{u}(:,1,:) &= \begin{bmatrix}0 & 0\\ 0.9 & 0\end{bmatrix},\ \mathbf{F}_{u}(:,2,:) = \begin{bmatrix}0.25 & 0\\ 0 & 0.65\end{bmatrix}\\ \mathbf{F}_{\phi}(:,:,1) &= \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\\ \mathsf{M}(\mathbf{x},\mathbf{u}) &= \left[\!\!\left[\begin{pmatrix}1\\ x_1\end{pmatrix},\begin{pmatrix}1\\ x_2\end{pmatrix},\begin{pmatrix}1\\ u\end{pmatrix}\right]\!\!\right]\end{aligned} \tag{15}$$

The mathematical equation (10) in Proposition 2 calculates the Jacobian matrix column by column, taking advantage of the TT based eMTI model structure as explained in Proposition 1 and Definition 3. The algorithm of Proposition 2 is described as follows.

$$\begin{aligned}\dot{\mathbf{x}} &= \begin{bmatrix}0.27x_1+0.08u+0.378x_1x_2+0.045x_1u+0.1x_2u\\ 0.195x_1u+0.273x_1x_2u\end{bmatrix}\\ \frac{\partial\dot{\mathbf{x}}}{\partial x_1} &= \begin{bmatrix}0.378x_2+0.045u+0.27\\ 0.273x_2u+0.195u\end{bmatrix}\\ \frac{\partial\dot{\mathbf{x}}}{\partial x_2} &= \begin{bmatrix}0.378x_1+0.1u\\ 0.273x_1u\end{bmatrix}\\ \frac{\partial\dot{\mathbf{x}}}{\partial u} &= \begin{bmatrix}0.045x_1+0.1x_2+0.08\\ 0.195x_1+0.273x_1x_2\end{bmatrix}\\ \mathbf{J} &= \begin{bmatrix}\frac{\partial\dot{\mathbf{x}}}{\partial x_1} & \frac{\partial\dot{\mathbf{x}}}{\partial x_2} & \frac{\partial\dot{\mathbf{x}}}{\partial u}\end{bmatrix}\end{aligned} \tag{16}$$

1) Initially, the TT contraction between the parameter tensor $\mathsf{F}$ and the monomial tensor $\mathsf{M}(\mathbf{x},\mathbf{u})$ as explained in Proposition 1 is computed which is given in (17). Since the eMTI model encompasses the multilinear algebra, the representation of $\dot{\mathbf{x}}$ in (16) shows us that each element of the vector contains the system variables written as a sum of products.

$$\begin{aligned}\mathbf{H}_{x_1}(1,1,:) &= \begin{bmatrix}0.4 & 0.6x_1\end{bmatrix}\\ \mathbf{H}_{x_2}(:,1,:) &= \begin{bmatrix}0.8+0.1x_2 & 0\\ 0.3 & 0.5+0.7x_2\end{bmatrix}\\ \mathbf{H}_{u}(:,1,:) &= \begin{bmatrix}0.25u & 0\\ 0.9 & 0.65u\end{bmatrix}\\ \mathbf{H}_{\phi}(:,:,1) &= \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\end{aligned} \tag{17}$$

2) The partial derivative of this function vector $\dot{\mathbf{x}}$ with respect to each system variable in (16) involves computing the special TT tensor explained in Definition 3 for each variable. The special TT tensor cores for each system variable of the example 1 is given in equation (18) where only the modified TT cores are expressed corresponding to each variable and the rest of the cores will be equal to the ones in (17).

$$\begin{aligned}\mathbf{H}^1_{x_1}(1,1,:) &= \begin{bmatrix}0 & 0.6\end{bmatrix}\\ \mathbf{H}^2_{x_2}(:,1,:) &= \begin{bmatrix}0.1 & 0\\ 0 & 0.7\end{bmatrix}\\ \mathbf{H}^3_{u}(:,1,:) &= \begin{bmatrix}0.25 & 0\\ 0 & 0.65\end{bmatrix}\end{aligned} \tag{18}$$

3) Finally, these special TT tensors undergo a tensor contraction operation between its individual TT cores along the TT ranks as explained in Definitions 1 and 2 to provide the symbolic solution of the partial derivatives of $\dot{\mathbf{x}}$ as given in (16) or, in other words, the Jacobian matrix defined in Proposition 2.

The developed linearization method for the Jacobian matrix is coded in the Python programming language and is explained as a pseudocode in Algorithm 1.

**Algorithm 1** Efficient Linearization of TT based eMTI model

**Require:** $\mathsf{F} \leftarrow \left[\!\!\left[\mathbf{F}_{x_1}, \mathsf{F}_{x_2}, \cdots, \mathsf{F}_{x_n}, \mathsf{F}_{u_{n+1}}, \mathsf{F}_{u_{n+m}}, \mathbf{F}_\phi\right]\!\!\right]$,

$\mathsf{M}(\mathbf{x},\mathbf{u}) \leftarrow \left[\!\!\left[\begin{pmatrix}1\\ x_1\end{pmatrix},\cdots,\begin{pmatrix}1\\ x_n\end{pmatrix},\begin{pmatrix}1\\ u_1\end{pmatrix},\cdots,\begin{pmatrix}1\\ u_m\end{pmatrix}\right]\!\!\right]$

1: $\mathsf{H} \leftarrow \left[\!\!\left[\mathbf{H}_{x_1}, \mathsf{H}_{x_2}, \cdots, \mathsf{H}_{x_n}, \mathsf{H}_{u_{n+1}}, \mathsf{H}_{u_{n+m}}, \mathbf{H}_\phi\right]\!\!\right]$
2: **for** $i \leftarrow 1, n+m$ **do**
3: $\quad \mathsf{H}^i \leftarrow \left[\!\!\left[\mathbf{H}^i_{x_1}, \mathsf{H}^i_{x_2}, \cdots, \mathsf{H}^i_{x_n}, \mathsf{H}^i_{u_{n+1}}, \mathsf{H}^i_{u_{n+m}}, \mathbf{H}^i_\phi\right]\!\!\right]$
4: $\quad \mathbf{J} \leftarrow \mathbf{J}$ *column concatenated with* $\left[\mathsf{H}^i\right]$
5: **end for**

Figure 10 presents the root mean square error (RMSE) values of the state predictions between the TT-based eMTI model in (15) and its linearized counterparts, evaluated around fixed operating points. A fixed step size, $h = 10^{-8}$ is used in the forward finite differences algorithm, which is the default method and step size used in IPOPT for finite-difference approximation. The ordinary differential equa-

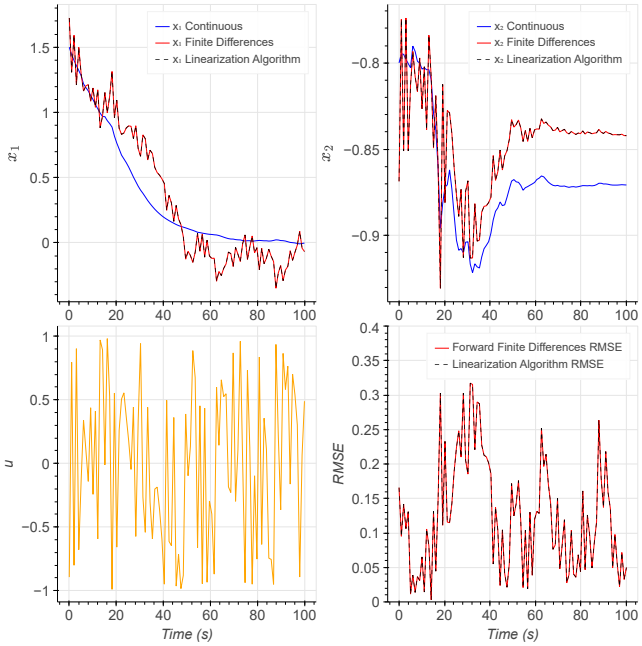


Fig. 10. Model Simulation RMSE: Finite differences vs Proposed algorithm

tion (ODE) solver used for obtaining the numerical solution of (16) is the explicit Runge-Kutta method of order 5(4), commonly known as the RK45 method. Figure 10 also shows that the numerical accuracy of both the linearization strategies are almost identical for a fixed set of parameters. The computational efficiency of the proposed algorithm compared to the conventional finite differences method is verified by the TT based eMTI model, developed and simulated for building heating systems, based on real-time measurement data from smart thermostats. More information on this model can be found in [11]. The computational efficiency of the solver after the implementation of the new algorithm improved significantly in comparison with the finite differences method. Table I shows the comparison between the simulation results of the different computational time averages of the Jacobian for varying number of system variables corresponding to a fixed set of model parameters of the TT based eMTI model. The simulation experiments were performed using the numerical Jacobian where the TT contraction between $\mathsf{F}$ and $\mathsf{M}$ is pre-computed and the current operating point $(\bar{\mathbf{x}}, \bar{\mathbf{u}})$ is provided at each iteration to the specialized TT tensor in definition 3. All simulations were conducted on a 64-bit operating system featuring an Intel (R) Core (TM) Ultra 7 155U (1.70 GHz) processor and a 32 GB RAM.

TABLE I

CPU TIMES FOR LINEARIZATION (IN S)

| n | standard finite differences | TT algorithm |
|---|---|---|
| 20 | 3.0 | 0.3 |
| 40 | 11.3 | 2.0 |
| 80 | 43.7 | 11.0 |

## V. CONCLUSIONS

This paper proposes a new algorithm to calculate the partial derivatives of an explict multilinear time-invariant (eMTI) state space model with respect to the system variables, i.e. the Jacobian – which is necessary for linearization. In contrast to other methods [13], which use canonical polyadic (CP) decompositions of the parameter tensors, this paper is the first to tackle the problem if the tensor-train (TT) decomposition is used. Tensor train decomposition techniques offer tools for approximations in different application domains, especially for large scale energy modeling.

Linearized eMTI models in TT representation enables the use of well-established tools from linear system theory for various analysis and design techniques. As the computation of the proposed linearization is superior to standard finite differences methods, the algorithm proposed in this paper could enable shorter sampling times which are relevant especially for fast or complex systems. Future work will focus on real-time implementation of the algorithm in industrial applications such as HVAC systems for buildings.

## REFERENCES


[1] G. Lichtenberg, "Hybrid tensor systems," Ph.D. dissertation, Technische Universität Hamburg, 2011.

[2] G. Lichtenberg, G. Pangalos, C. C. Yáñez, A. Luxa, N. Jöres, L. Schnelle, and C. Kaufmann, "Implicit multilinear modeling : an introduction with application to energy systems," *at - Automatisierungstechnik*, vol. 70, no. 1, pp. 13–30, 2022.

[3] K. Kruppa, G. Pangalos, and G. Lichtenberg, "Multilinear approximation of nonlinear state space models," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 9474–9479, 2014, 19th IFAC World Congress.

[4] W. Hackbusch, "Tensor spaces and numerical tensor calculus," *Springer Series in Computational Mathematics*, 2012.

[5] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009. [Online]. Available: https://doi.org/10.1137/07070111X

[6] G. Pangalos, A. Eichler, and G. Lichtenberg, "Tensor systems - multilinear modeling and applications," in *Proceedings of the 3rd International Conference on Simulation and Modeling Methodologies, Technologies and Applications*. SciTePress, 2013, pp. 275–285. [Online]. Available: https://doi.org/10.5220/0004475602750285

[7] P. Gelß, S. Klus, J. Eisert, and C. Schütte, "Multidimensional approximation of nonlinear dynamical systems," *Journal of Computational and Nonlinear Dynamics*, vol. 14, no. 6, 2019. [Online]. Available: http://dx.doi.org/10.1115/1.4043148

[8] P. Gelß, A. Ramos, T. Bake, H. D. Nguyen, M. Lücke, D. S. Klus, M. Scherer, and D. F. Nüske, "scikit_tt," 2025, accessed: 26.11.2025. [Online]. Available: https://github.com/PGelss/scikit_tt

[9] K. Kruppa, "Multilinear design of decentralized controller networks for building automation systems," Ph.D. dissertation, HafenCity Universität Hamburg, 2018.

[10] I. V. Oseledets, "Tensor-train decomposition," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011. [Online]. Available: https://doi.org/10.1137/090752286

[11] J. Cherian, E. Uhlenberg, H. G. S. Maregowda, L. S. Vallejos, T. Warnecke, and G. Lichtenberg, "Tensor train based explicit multilinear modeling and control of heating systems," 2026, accepted to the $12^{th}$ CoDIT Conference 2026.

[12] A. Wächter, "Short tutorial: Getting started with ipopt in 90 minutes," in *Combinatorial Scientific Computing, 01.02. - 06.02.2009*, ser. Dagstuhl Seminar Proceedings, U. Naumann, O. Schenk, H. D. Simon, and S. Toledo, Eds., vol. 09061. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany, 2009.

[13] C. Kaufmann, G. Pangalos, G. Lichtenberg, and O. Gomis-Bellmunt, "Small-signal stability analysis of power systems by implicit multilinear models," *IEEE Access*, vol. 14, pp. 37 951–37 967, 2026.