TenSy Grid

Tensors for System Analysis of Converter-dominated Power Grids

D 2.1 Improved Multilinearization Algorithm

by HAW

Public

Co-funded by the European Union

Supported by:

Federal Ministry for Economic Affairs and Energy

on the basis of a decision by the German Bundestag

MINISTERIO DE CIENCIA, INNOVACIÓN Y UNIVERSIDADES

AGENCIA ESTATAL DE INVESTIGACIÓN

CDTI INNOVACIÓN

XJENZA MALTA

This research was funded by CETPartnership, the Clean Energy Transition Partnership under the 2023 joint call for research proposals, co funded by the European Commission (GA 101 069750) and with the funding organizations detailed on https://cetpartnership.eu/funding-agencies-and-call-modules.

About TenSyGrid

The demand for the power grid in Europe is undergoing profound changes due to an increasing number of decentralized feed-in points and the fluctuating supply from renewable energies. This complexity in interactions between power grid components poses a challenge for maintaining system stability. To address this, the European project TenSyGrid is developing a toolbox for direct stability assessment using multilinear models to capture the complex dynamics of power grid components. The objective is to support grid operators in assessing large power grids primarily powered by renewable energy. The toolbox will be compatible with existing commercial software packages to facilitate integration into current workflows.

Project Title	Tensors for System Analysis of Converter-dominated Power Grids
Programme	Horizon Europe - Clean Energy Transition Partnership (CETP)
Project Number	CETP-FP-2023-00138
Project Type	Research-oriented approach (ROA)
Call Module	CM2023-02 Energy system flexibility: renewables production, storage and system integration
Transition Initiative	TRI1 Net-zero emissions energy system
Project Start	01.12.2024
Project Duration	3 years
Coordinator	Fraunhofer IWES
Project Website	`www.tensygrid.eu`

Consortium

(Project Coordinator)

About this document

Deliverable Number	D 2.1
Title	Improved Multilinearization Algorithm
Work Package	2
Leading Partner	Hamburg University of Applied Sciences
Authors	Dr. Teresa Wong Prof. Dr.-Ing. Gerwald Lichtenberg M. Eng. Christoph Kaufmann
Reviewers	M. Eng. Christoph Kaufmann
Version	V1.0
Due Date	30.11.2025
Version Date	28.11.2025
Reviewer Accepted	26.11.2025
WP Leaders Accepted	28.11.2025

Dissemination Level	
PU	Public

Summary

Multilinear models have been shown to capture the dynamics of hybrid nonlinear systems, demonstrating strong potential for applications in power networks. However, computations in full tensors become infeasible for multilinear power system models as the number of dimensions grows exponentially with the number of states and inputs. A new approach to multilinearization is developed to break the curse of dimensionality, offering an optimized algorithm capable of handling higher-dimensional models.

This document outlines the mathematical background of the various multilinearization methods. The implementation of numerical multilinearization with the MTI-Toolbox is illustrated through two power system examples: the inverters in a three-bus system and the grid-following converter in a system connected to a Thévenin-equivalent network. This deliverable report presents an improved approach to multilinearization, which can be applied to components in power networks that contain nonlinearities. Finally, we discuss avenues for extending the multilinearization approaches toward more general use in power grid modeling and analysis.

Contents

List of Figures

List of Tables

1 Introduction

This deliverable report aims to present the mathematical background of multilinearization and to outline the various approaches of multilinearizing nonlinear models, including the newly improved algorithm for numerical multilinearization. The basics of multilinear models are introduced in Deliverable 1.1 [1]. This chapter provides a summary of those concepts and serves as a preliminary material for the subsequent discussion on multilinearization methods and their application to power grid systems.

A nonlinear time-invariant model (NTI) can be described by a system of first-order ODE:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{u}), \\ \mathbf{y} &= \mathbf{g}(\mathbf{x}, \mathbf{u}). \end{aligned} \tag{1.1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state vector, $\mathbf{u} \in \mathbb{R}^m$ is the input vector, $\mathbf{y} \in \mathbb{R}^p$ is the output vector, $f_i : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ are state equations and $g_i : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p$ are output equations. The state and output equations may contain nonlinear terms such as reciprocal and trigonometric functions.

A multilinear state space model can be written in terms of multilinear functions, which are functions that would be linear if all but one state or input are held constant. Thus, only products of the states and inputs are allowed in multilinear functions, but not the quadratic or higher order terms. The combinations of the various states and inputs can be represented by the monomial tensor [2, 3]

$$\mathsf{M}(\mathbf{x}, \mathbf{u}) = \begin{pmatrix} 1 \\ u_m \end{pmatrix} \circ \cdots \circ \begin{pmatrix} 1 \\ u_1 \end{pmatrix} \circ \begin{pmatrix} 1 \\ x_n \end{pmatrix} \circ \cdots \circ \begin{pmatrix} 1 \\ x_1 \end{pmatrix}, \tag{1.2}$$

where $\mathsf{M}(\mathbf{x}, \mathbf{u}) \in \mathbb{R}^{\overbrace{2 \times \cdots \times 2}^{n+m}}$ is a rank-1 tensor, and $\circ$ denotes the outer product. A multilinear time-invariant system (MTI) in tensor form is thus given by a contracted product

$$\begin{aligned} \dot{\mathbf{x}} &= \langle \mathsf{F} | \mathsf{M}(\mathbf{x}, \mathbf{u}) \rangle, \\ \mathbf{y} &= \langle \mathsf{G} | \mathsf{M}(\mathbf{x}, \mathbf{u}) \rangle, \end{aligned} \tag{1.3}$$

where $\mathsf{F} \in \mathbb{R}^{\overbrace{2 \times \cdots \times 2}^{(n+m)} \times n}$ is the state transition tensor and $\mathsf{G} \in \mathbb{R}^{\overbrace{2 \times \cdots \times 2}^{(n+m)} \times p}$ is the output tensor. The j-th element of the state derivative vector is thus

$$\dot{x}_j = \sum_{i_1=1}^{2} \cdots \sum_{i_\beta=1}^{2} \varphi_{\mathbf{i},j} \mu_{\mathbf{i}_i}, \ \forall j = 1, ..., n, \tag{1.4}$$

where $\varphi_{\mathbf{i},j}$ are scalar elements of the state transition tensor F, and $\mu_{\mathbf{i}_i}$ are scalar elements of the monomial tensor $\mathsf{M}(\mathbf{x}, \mathbf{u})$. The subscript vector $\mathbf{i} = [i_1, ..., i_\beta]^{\mathrm{T}}$ are indices of a tensor with dimension $\beta := 2^{(n+m)}$. This means there are at most β multilinear combinations of the elements of $\mathbf{x}$ and $\mathbf{u}$. The state transition tensor F has one more dimension, thus its scalar elements $\varphi_{\mathbf{i},j}$ have an additional index $j = 1, ..., n$. For a comprehensive discussion on tensor representation and tensor decomposition, see Deliverable 1.1 [1].

The models of (1.3) are of the explicit class, and they are often referred to as eMTI models. In eMTI models, the state derivatives and output are expressed as operations between states and inputs. A limitation of eMTI models is that connecting multiple eMTI models may not result in an overall eMTI system [4]. Implicit MTI models (iMTI) addresses this problem by allowing multiplication of state derivatives with states, inputs, and/or outputs, enabling the representation of broken rational functions. The iMTI class is closed, meaning that any combination of iMTI models, whether in series, parallel, or feedback, results in a system that remains within the iMTI class [4, 5]. In tensor form they can be written as

$$\mathbf{0} = \langle \mathsf{H} | \mathsf{M}(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{u}, \mathbf{y}) \rangle, \tag{1.5}$$

where H is the parameter tensor, and $\mathsf{M}(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{u}, \mathbf{y})$ is the augmented monomial tensor. The output equations are not required in the implicit form but can be included in H. A detailed explanation on implicit models and their tensor representation can be found in Deliverable D1.1 [1].

It is practical to convert between model classes to adapt to various applications, since they may be represented in either of these classes while their computation may be more feasible in a particular class, depending on the nature of mathematical functions, their solvability, and the complexity of numerical solution methods. The approaches of multilinearizing NTI systems to eMTI and iMTI models, and the conversion between eMTI and iMTI models are summarized in Fig. 1.1. The different approaches of multilinearization will be detailed in Section 2 and 3, while Section 4 describes the implementation of multilinearization using the MTI Toolbox, and Section 4.3 provides examples of implementation for components in power grid systems.

To illustrate the methods of multilinearization in Sections 2 and 3, we use an example of a polynomial model of second-order:

$$\begin{aligned} \dot{x}_1 &= x_1 + 2x_1^2 + x_1 x_2, \\ \dot{x}_2 &= x_2 - 4x_1^2 - 2x_1 x_2. \end{aligned} \tag{1.6}$$

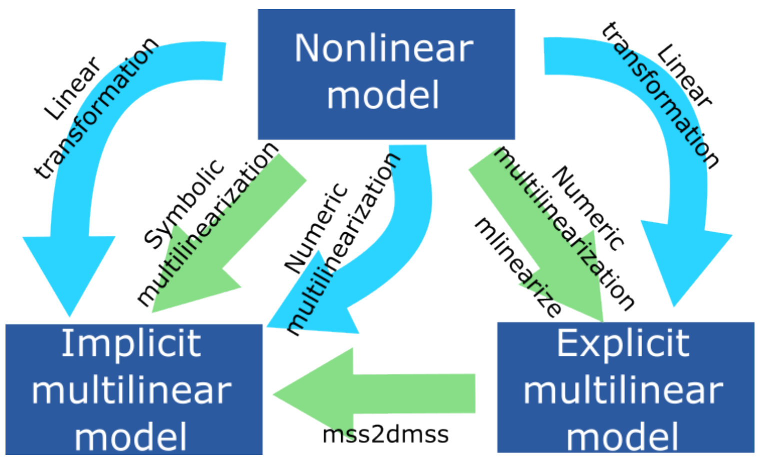

Nonlinear model
Linear transformation
Linear transformation
Symbolic multilinearization
Numeric multilinearization
Numeric multilinearization
mlinearize
Implicit multilinear model
Explicit multilinear model
mss2dmss

Figure 1.1: Approaches to convert nonlinear time-invariant into explicit and implicit multilinear time-invariant models. The conversion from explicit to implicit multilinear models is demonstrated in Section 2, and is implemented by the function `mss2dmss` in the MTI Toolbox [6].

2 Symbolic Multilinearization

Symbolic multilinearization involves the finding of a set of symbolic equations which are mathematical equivalent. It has the advantage that the multilinearized model represents the dynamic behaviour exactly as the nonlinear model and not approximative. Here we discuss two symbolic methods: implicit multilinearization using differential algebraic equations (DAEs), and linear transformation of polynomial to multilinear models, where omputer algebra such as the Symbolic Math Toolbox in MATLAB® could be used.

2.1 Implicit multilinearization

An algebraic equation can be added to the implicit ordinary differential equations to form a set of DAEs. An auxiliary variable y_1 is introduced to multilinearize the nonlinear terms, such that the multilinear model can directly represent the physical systems with fewer approximations [5, 7, 8]. In the general form, the implicit model in (1.5) extended with a vector of auxiliary variables is written as

$$\mathbf{0} = \langle \mathsf{H} | \mathsf{M}(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{u}, \mathbf{y}) \rangle, \tag{2.1}$$

where H is the parameter tensor, and $\mathbf{y} \in \mathbb{R}^p$ is a vector of p auxiliary variables.

In this approach, the example model in (1.6) is transformed into an implicit state space model as:

$$\begin{aligned} 0 &= \dot{x}_1 - x_1 - 2x_1y_1 - x_1x_2, \\ 0 &= \dot{x}_2 - x_2 + 4x_1y_1 + 2x_1x_2, \\ 0 &= y_1 - x_1. \end{aligned} \tag{2.2}$$

An extensive list of nonlinearities identified for power system components is reported in Deliverables 3.1 [9]. They include terms such as saturation limits, squared, trigonometric and exponential terms, hysteresis, and discrete switching behaviors in components such as synchronous generators, converters, power system stabilizers, and controllers. The symbolic representation of these nonlinearities are reported in Deliverables 3.2 [10].

2.2 Linear transformation

Certain classes of polynomial models can be transformed into multilinear models using linear state transformation [11]. This method uses a non-singular transformation matrix to change the state vector $\mathbf{x}$ in a nonlinear model to a new basis $\tilde{\mathbf{x}}$ in a multilinear model. That is, for a system of ordinary differential equations $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$, the polynomial functions $\mathbf{f}(\mathbf{x})$ can be transformed into multilinear functions by an invertible transformation matrix $\mathbf{T}$:

$$T : \mathbb{R}^n \to \mathbb{R}^n, \tilde{\mathbf{x}} := \mathbf{T}\mathbf{x}, \tag{2.3}$$

In the example model (1.6), a linear coordinate transformation can be applied to the states such that the nonlinear model can be mapped to a multilinear one:

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{pmatrix}, \tag{2.4}$$

$$\begin{aligned} \dot{\tilde{x}}_1 &= \tilde{x}_1 + \tilde{x}_1\tilde{x}_2, \\ \dot{\tilde{x}}_2 &= \tilde{x}_2. \end{aligned} \tag{2.5}$$

The work [11] uses the following notation for a generic second-order polynomial model:

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \mathbf{g}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \pi_{20}^{(1)} x_1^2 + \pi_{11}^{(1)} x_1 x_2 + \pi_{02}^{(1)} x_2^2 + \pi_{10}^{(1)} x_1 + \pi_{01}^{(1)} x_2 + \pi_{00}^{(1)} \\ \pi_{20}^{(2)} x_1^2 + \pi_{11}^{(2)} x_1 x_2 + \pi_{02}^{(2)} x_2^2 + \pi_{10}^{(2)} x_1 + \pi_{01}^{(2)} x_2 + \pi_{00}^{(2)} \end{pmatrix}. \tag{2.6}$$

where $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^n$ is the polynomial function, $(x_1, x_2)^T \in \mathbb{R}$ are states, with $\pi_\alpha^{(i)} \in \mathbb{R}$, $i \in (1, 2)$ and $\alpha \in \mathbb{N}_0^2$ is a multi-index. The goal was to find a linear transformation

$$\mathbf{f} := \mathbf{T} \circ \mathbf{g} \circ \mathbf{T}^{-1} \tag{2.7}$$

where $\mathbf{f}$ is a multilinear function. The work [11] found sufficient and necessary conditions for the existence of solution to (2.7) for polynomial functions that are not affine. One case is illustrated in the following. Consider the matrix $\mathbf{T}$:

$$\mathbf{T} = \begin{pmatrix} 1 & y^* \\ x^* & 1 \end{pmatrix} \tag{2.8}$$

where $x*, y* \in \mathbb{C}$. Then $\mathbf{T}$ is a valid transformation for (2.7) if and only if one of a few conditions listed in [11] is satisfied. For example:

It either holds that

$$\pi_{02}^{(1)}, \pi_{20}^{(1)} \neq 0, \quad \pi_{02}^{(2)}, \pi_{11}^{(2)}, \pi_{20}^{(2)} = 0, \tag{2.9}$$

or

$$\pi_{02}^{(1)}, \pi_{11}^{(1)}, \pi_{20}^{(1)} = 0, \quad \pi_{02}^{(2)}, \pi_{20}^{(2)} \neq 0. \tag{2.10}$$

Furthermore, it holds that

$$x^* = \frac{\pi_{11}^{(1)} \pm \sqrt{\left(\pi_{11}^{(1)}\right)^2 - 4\pi_{20}^{(1)}\pi_{02}^{(1)}}}{2\pi_{02}^{(1)}} \neq 0, \quad \text{and} \quad y^* = \frac{\pi_{11}^{(1)} \pm \sqrt{\left(\pi_{11}^{(1)}\right)^2 - 4\pi_{20}^{(1)}\pi_{02}^{(1)}}}{2\pi_{20}^{(1)}} \neq 0, \tag{2.11}$$

in case of (2.9), or in case of (2.10)

$$x^* = \frac{\pi_{11}^{(2)} \pm \sqrt{\left(\pi_{11}^{(2)}\right)^2 - 4\pi_{20}^{(2)}\pi_{02}^{(2)}}}{2\pi_{02}^{(2)}} \neq 0, \quad \text{and} \quad y^* = \frac{\pi_{11}^{(2)} \pm \sqrt{\left(\pi_{11}^{(2)}\right)^2 - 4\pi_{20}^{(2)}\pi_{02}^{(2)}}}{2\pi_{20}^{(2)}} \neq 0, \tag{2.12}$$

where, in the equations (2.11) and (2.12), there is either a plus in both cases or a minus in both equalities.

Algorithms for conversion of explicit NTI models to eMTI are presented in [11]. Linear transformation for implicit MTI systems would have the potential to include more types of variables and parameters, and will be an avenue for future studies.

3 Numerical Multilinearization

3.1 Multilinear approximation

Numerical multilinearization is formulated as a minimization problem, where the multilinear model can be obtained by employing the orthogonal projection theory [3]. In this approach, nonlinear functions are approximated as a linear combination of base functions, and the difference between this approximation and the original functions are minimized. The computation involves integration of functions which are performed numerically.

We seek to approximate a nonlinear model such as (1.1) with a multilinear model $\mathbf{h}(\mathbf{x}, \mathbf{u}) \approx \mathbf{f}(\mathbf{x}, \mathbf{u})$. The components of $\mathbf{h}(\mathbf{x})$ are

$$\begin{aligned} h_j(\mathbf{x}, \mathbf{u}) &= \sum_{i=1}^{\beta} \varphi_{\mathbf{i}_i,j} \mu_{\mathbf{i}_i}(\mathbf{x}, \mathbf{u}) \\ &= \varphi_{\mathbf{i}_1,j} + \varphi_{\mathbf{i}_2,j} x_1 + \varphi_{\mathbf{i}_3,j} x_2 + \varphi_{\mathbf{i}_4,j} x_1 x_2 + \cdots \\ &\quad + \varphi_{\mathbf{i}_\beta,j} x_1 \cdots x_n u_1 \cdots u_m, \ \forall j = 1, ..., n \end{aligned} \tag{3.1}$$

where $\mu_{\mathbf{i}}(\mathbf{x}, \mathbf{u})$ are the scalar elements of $\mathsf{M}(\mathbf{x}, \mathbf{u})$, which are the base functions that spans some multilinear space $\mathcal{M}$, representing all possible multilinear combinations of the elements of $\mathbf{x}$ and $\mathbf{u}$. In multilinearization, the monomial tensor $\mathsf{M}(\mathbf{x}, \mathbf{u})$ is typically decomposed into a factorized form, such as the canonical polyadic form. Here we present the expanded form of the monomial tensor and explore how approximations can be made by neglecting certain higher-order terms with the sparse grids method.

The approximation problem uses the orthogonality condition to mininize the distance between the approximation and the original function [12]. Several definitions are needed for the formulation of this problem. First the inner product of two real functions $f(\mathbf{x})$ and $g(\mathbf{x})$ over a domain D is defined as

$$\langle f(\mathbf{x}), g(\mathbf{x}) \rangle_w := \int_D f(\mathbf{x}) g(\mathbf{x}) w(\mathbf{x}) d\mathbf{x}, \tag{3.2}$$

where $w(\mathbf{x})$ is a weighing function that defines the importance of good approximation properties for certain regions in D. The norm of the inner product is defined as

$$\|f(\mathbf{x})\|_w := \sqrt{\langle f(\mathbf{x}), f(\mathbf{x}) \rangle_w}. \tag{3.3}$$

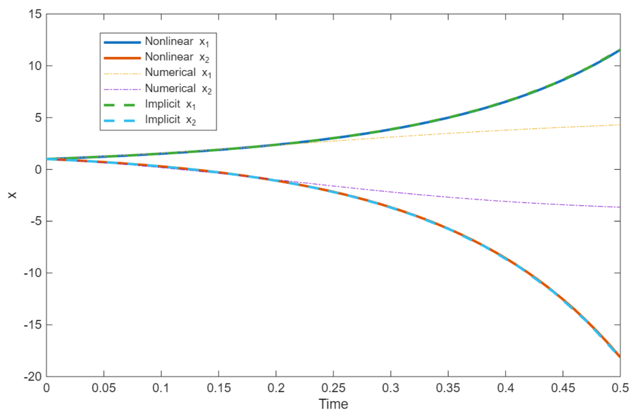


Figure 3.1: Comparison of the nonlinear model (1.6) and multilinear models from numerical multilinearization and implicit symbolic multilinearization.

The approximation problem is to find a function h_j that minimizes the difference

$$\min_{h_j \in \mathcal{M}} \|h_j(\mathbf{x}, \mathbf{u}) - f_j(\mathbf{x}, \mathbf{u})\|_w, \ \forall j = 1, ..., n. \tag{3.4}$$

The parameters $\varphi_{\mathbf{i}_i,j}$ in h_j are elements of the transition tensor $\mathbf{F} \in \mathbb{R}^{2^n}$ calculated as

$$\varphi_{\mathbf{i}_i,j} = \langle f_j(\mathbf{x}, \mathbf{u}), \mu_{\mathbf{i}_i}(\mathbf{x}, \mathbf{u}) \rangle_w, \forall i = 1, ..., \beta, \ \forall j = 1, ..., n. \tag{3.5}$$

which are orthogonal projections of nonlinear functions f_j on the bases $\mu_{\mathbf{i}_i}$.

The main task of numerical multilinearization is to perform numerical integration to compute the inner products in (3.4) and (3.5).

The multilinear model of the example system (1.6) after numerical multilinearization is

$$\begin{aligned} \dot{x}_1 &= 5x_1 + x_1x_2 - 4/3, \\ \dot{x}_2 &= x_2 - 8x_1 - 2x_1x_2 - 8/3. \end{aligned} \tag{3.6}$$

The coefficients and constants depend on the range of operating values and the algorithm for numerical integration, which will be detailed in the following sections. As shown in Fig. 3.1, the implicit multilinear model closely matches the nonlinear model as it is an exact representation of it; the numerical multilinearization matches the functions before the nonlinear model takes an exponential growth.

Reducing multilinear sparsity: Multilinear order

It is useful to introduce the concept of multilinear order [3], which is defined as the number of variables multiplied in a monomial. By imposing a maximum multilinear order, we limit the degree of terms in the monomial tensor M$(\mathbf{x}, \mathbf{u})$ and eliminate sparsity that is inherent in the multilinear structure, particularly in the higher-order terms involving products of many states and inputs. The total number of coefficients in the multilinear model is $2^{(n+m)}$ per state, which can lead to significant storage and computational effort. However, systems with inherent multilinear structure often exhibit sparsity, particularly in higher-order terms involving products of many states and inputs. To exploit this, we introduce here the concept of multilinear order [3], defined as the number of variables multiplied in a monomial. By imposing a maximum multilinear order k, we limit the degree of terms in the monomial tensor and reduce the number of coefficients per state to,

$$\sum_{l=0}^{k} \binom{n+m}{l}, \tag{3.7}$$

which shows that each kth-degree multilinear term corresponds to choosing k variables from $(n+m)$.

3.2 Problems for real application

Whereas symbolic multilinearization can represent the nonlinear model exactly, it may not be always possible to find an analytical representation of the model. For example, the electrolyzer model in [13] has nonlinearities that can only be solved with multilinear approximation. Numerical multilinearization offers a way to model such nonlinear system, for example using Simulink® models.

Multilinearization requires the integration of functions, and a large number of grid points in each dimension is needed for sufficient accuracy in numerical integration. This results in the curse of dimensionality, where computations with full grids are impossible in higher dimensions due to excessive memory and time requirements. The following section discusses an approach to resolve this issue: the sparse grids method for the numerical integration in the process of multilinearization.

3.3 Possible solution: numerical sparse grid

This section presents the sparse grids method for the numerical multilinearization [14]. The integration of a nonlinear function $f(\mathbf{x}, \mathbf{u})$ is evaluated over a set of chosen sampling points for $\mathbf{x}$

and $\mathbf{u}$, raising the question of how to choose the distribution of sampling points in the variables to achieve sufficient accuracy in optimal computational time. The sparse grids method can be used to evaluate functions of each point of the grid and approximate an integral by quadrature formulas and interpolation.

Quadrature

A quadrature is the approximation of a definite integral as a weighted sum of function evaluations:

$$\int_a^b f(x)dx \approx \sum_{i=1}^{N} f(x_i)w_i, \tag{3.8}$$

where $f(x)$ is a continuous function within an interval $[a,b]$, and w_i is some precalculated weight. The weight w_i and the grid locations of variable x_i, also referred to as collocation knots, are determined by the choice of quadrature rule based on the probability distribution of x_i. Examples of probability distributions are uniform, normal, exponential, gamma, beta, and triangular. For each type of distribution, there can be different families of collocation knots specified by quadrature rules defined by polynomials (Fig. 3.2).

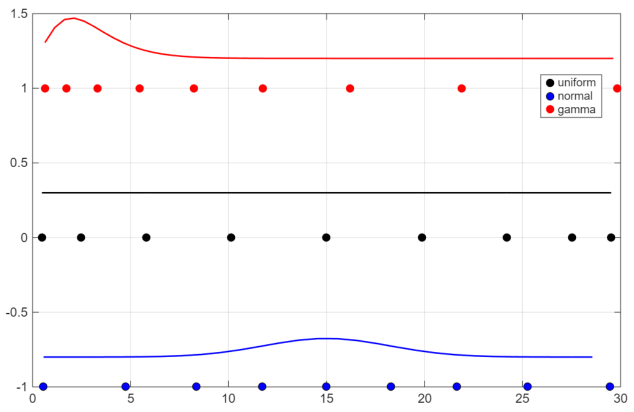


Figure 3.2: Examples of Gauss-type knots and their corresponding probability distribution function shown in solid lines [15].

The number of collocation knots is defined by both the quadruature rule and the level-to-knots function $m(\ell)$, where ℓ is the discretization level or level of quadrature (Fig. 3.3). The function $m(\ell)$ determines the order of the quadrature rule. Examples of level-to-knots function include the linear function $m(\ell) = \ell$, and the doubling function

$$m(\ell) = \begin{cases} 1, & \text{if } \ell = 1, \\ 2^{\ell-1}+1 & \text{if } \ell > 1. \end{cases} \tag{3.9}$$

One difference between the families of knots is whether they are nested or not. Nestedness means that as the order of the quadrature rule increases, the function evaluations at grid points from lower orders are also preserved. As shown in Fig. 3.3, Clenshaw-Curtis-type knots, which are an example of a nested sequence, retain the lower-order knots, allowing the function values at those grid points to be reused and thereby improving efficiency. On the contrary, families such as the Gauss-Legendre rule do not give a nested sequence, meaning that higher-order points do not repeat the lower ones, and therefore calculations must be performed for all the higher-order points.

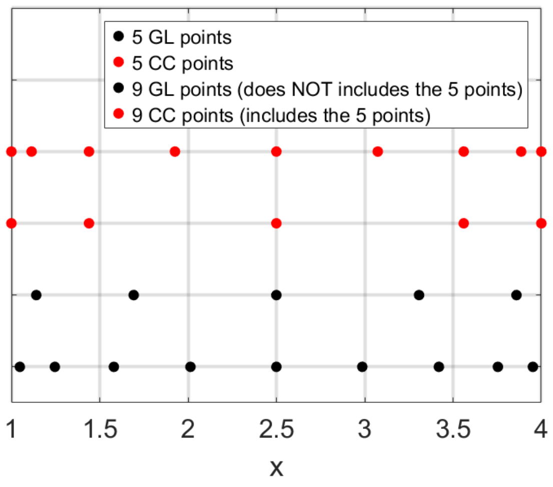


Figure 3.3: Clenshaw-Curtis points (nested, in red) and Gauss-Legendre points (non-nested, in black) with doubling level-to-knot function $m(\ell = 3)$ (5 points) and $m(\ell = 4)$ (9 points) for uniform distribution.

The 1D quadrature can thus be symbolized with the integration operator:

$$Q_\ell := \sum_{i=1}^{m(\ell)} f(x_i) w_i. \tag{3.10}$$

A multi-dimensional quadrature is a tensor product of 1D quadratures:

$$Q_\ell^d = Q_{\ell_1} \otimes Q_{\ell_2} \otimes \cdots \otimes Q_{\ell_d}, \tag{3.11}$$

where d is the dimension (Fig. 3.4). It is a combination of each point in a set of variables, and the weights are the products of the 1D weights.

Exactness of quadrature rules

The accuracy of the approximation can be expressed as the polynomial exactness of the 1D rule, or the degree of polynomial for which the integral can be computed exactly. For example, the trapezoidal (equispaced) rule $\int_b^a (c_0 + c_1 x) dx$ is only first order exact, which means that even with a large number of points, the quadrature will only recover a straight line. For higher orders the

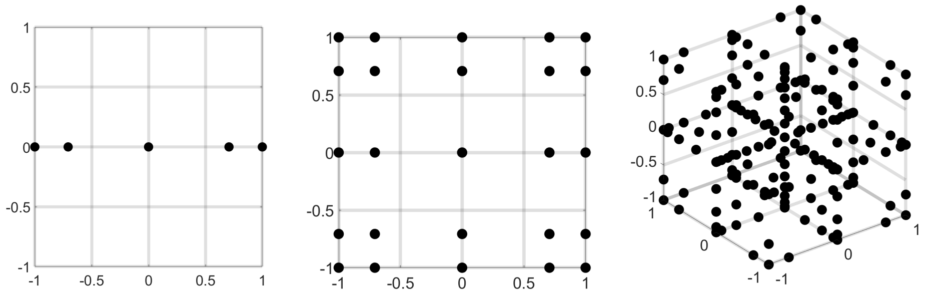


Figure 3.4: 1D, 2D, and 3D Clenshaw-Curtis quadatures.

trapezoidal rule encounters the Runge phenomenon [16], in which the errors grow exponentially. An $(N+1)$-point Clenshaw-Curtis rule $\int_b^a (c_0 + c_1 x + c_2 x^2 + \cdots + c_N x^N)dx$ is N^{th}-order exact, whereas the Gauss-Legendre rule has a high degree of exactness $(2N+1)$. However, exactness may not be a definitive measure of accuracy, as the performance of quadrature rules can vary depending on the type of functions being integrated, such as whether they are polynomials or not [17].

The accuracy of multi-dimensional quadratures depends on the exactness of each 1D quadrature (Fig. 3.5).

Multi-index truncation and Smolyak quadrature

The number of points in a full-tensor multidimensional quadrature using an n-point 1D rule in d dimensions is $N = n^d$. This grows rapidly with d, making evaluations of $f(x)$ expensive, with a total cost of $\prod_{n=1}^{N} m(\ell_n)$ calculations. In practice, high monomial accuracy is often sufficient, meaning accurate integration of high powers of individual variables x_i is needed, while accuracy for mixed terms like $x_1 x_2 \cdots x_N$ is not [18, 19].

To address this problem, a telescopic sum operator is introduced to the quadrature operator,

$$Q_l = \sum_{\ell=1}^{l} Q_\ell - Q_{\ell-1} = \sum_{\ell=1}^{l} \Delta_\ell, \tag{3.12}$$

where $\Delta_\ell = Q_\ell - Q_{\ell-1}$, and $Q_0 \equiv \varnothing$. Following (3.11), the multi-dimensional quadrature is:

$$\begin{aligned} Q_{\vec{l}}^d &= \sum_{\ell_1=1}^{l_1} \Delta_{\ell_1} \otimes \sum_{\ell_2=1}^{l_2} \Delta_{\ell_2} \otimes \cdots \otimes \sum_{\ell_d=1}^{l_d} \Delta_{\ell_d} \\ &= \sum_{\ell_1=1}^{l_1} \sum_{\ell_2=1}^{l_2} \cdots \sum_{\ell_d=1}^{l_d} (\Delta_{\ell_1} \otimes \Delta_{\ell_2} \otimes \cdots \otimes \Delta_{\ell_d}). \end{aligned} \tag{3.13}$$

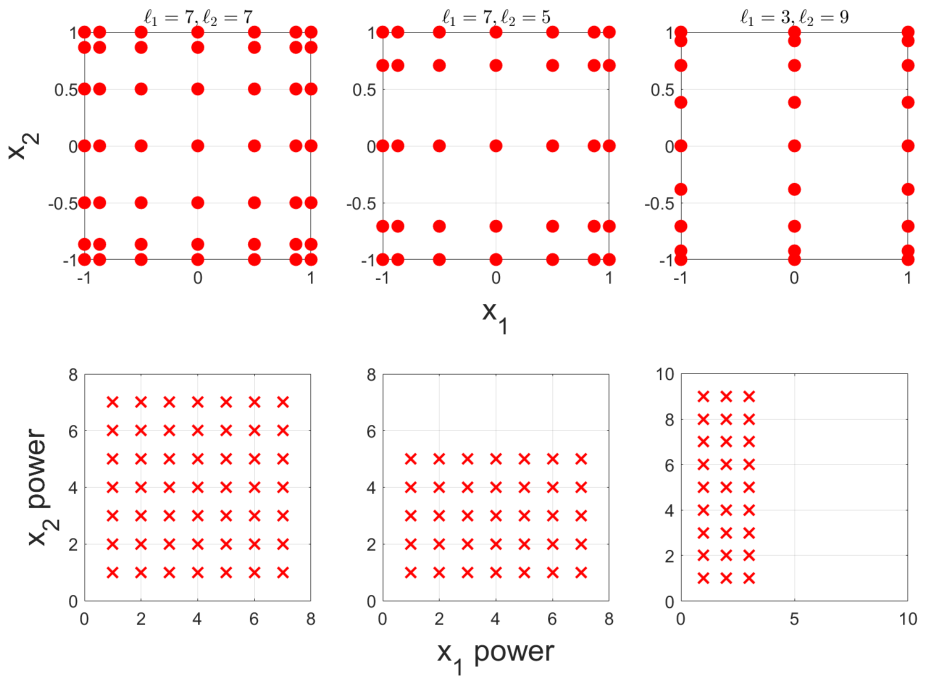


Figure 3.5: Illustration of the polynomial exactness of 2D quadratures with Clenshaw-Curtis rule and linear level-to-knots function. Figures in the top row show the sparse grid defined by the different levels ℓ in each variable, and the corresponding figures in the bottom row show the degree of polynomials that can be integrated exactly.

To exclude the computation of polynomials with higher degree of exactness, the multi-indices can be truncated by the 1-norm of the multi-indices:

$$\|\vec{\ell}\|_1 = \sum_{n=1}^{N} \ell_n \leq \mathcal{L}, \tag{3.14}$$

where $\vec{\ell} = [\ell_1, \ell_2, \cdots, \ell_N]^T \in \mathbb{N}$ are the multi-indices, and the total level $\mathcal{L} \in \mathbb{N}$ dictates the admissible indices. We note that $\mathcal{L}$ can be multi-dimensional and imposes various constraints on each ℓ_i, but exploring these aspects is beyond the scope of this study. A detailed discussion can be found in [18, 19].

The Smolyak grid is defined by the type of truncation in (3.14) and the doubling level-to-knots function in (3.9). Fig. 3.6 shows an example of the Smolyak quadrature with $\mathcal{L} = 4$, i.e. $\ell_1 + \ell_2 \leq 4$. Equation (3.13) for this Smolyak quadrature is thus

$$Q^2_{\|\vec{\ell}\|_1 \leq 4} = \sum_{\ell_1=1}^{3} \sum_{\ell_2=1}^{3} \Delta_{\ell_1} \otimes \Delta_{\ell_2}.$$

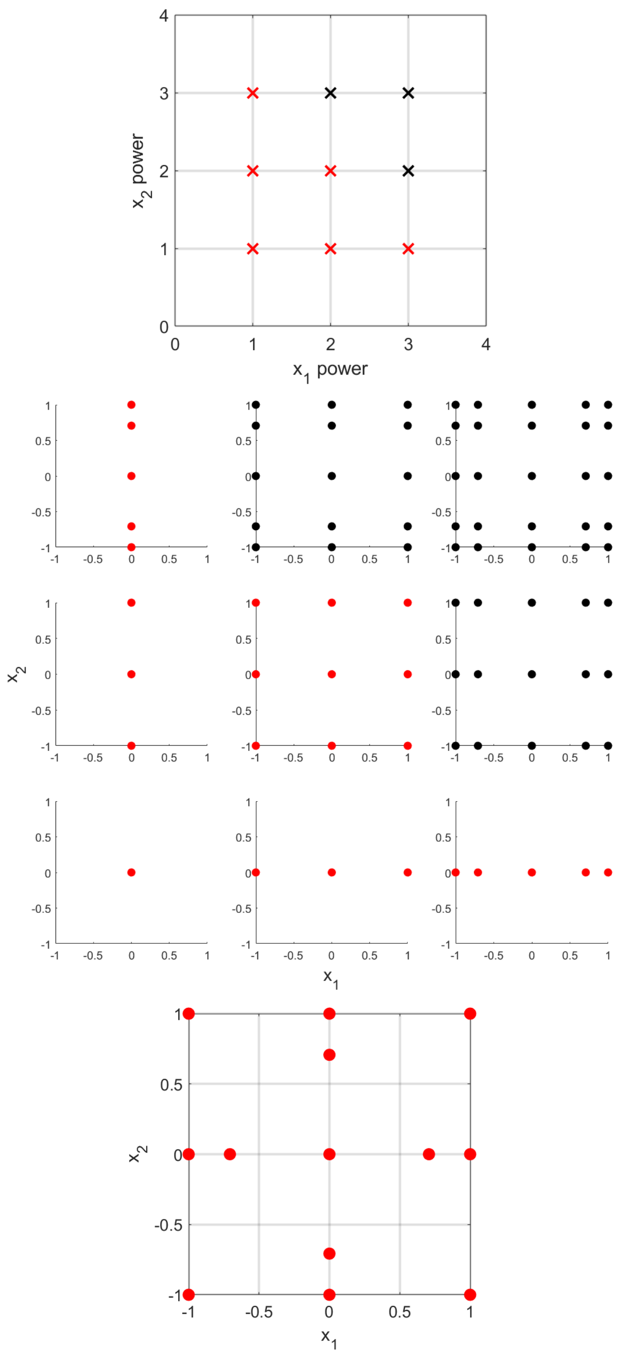


Figure 3.6: A Smolyak quadrature with $l_1 = l_2 = 3$. After truncation of indices (top figure), the higher-order grids (middle panel, black grids) are discarded, and the resultant combination of the lower order grids gives the sparse grid (bottom figure).

4 Implementation: MTI toolbox

The MTI-Toolbox [6] is a mathematical toolbox developed in MATLAB® for multilinear modeling, simulation, and analysis. It has been demonstrated that the dynamics of hybrid systems consisting of continuous- and discrete-valued parts can be represented in an adquate way by MTI models and reduction methods can be applied [2]. Its application include HVAC systems, fluid networks, and power networks.

4.1 Formats for iMTI and eMTI

iMTI

The implementation of the implicit multilinear model class in the MTI-Toolbox is reported in Deliverables D1.1 [1].

eMTI

The implementation of the explicit multilinear model class is described in the documentation for MTI-Toolbox 2.1 [6].

4.2 Sparse grids method

Numerical multilinearization in the explicit class is implemented in the function `mlinearize` in the MTI-Toolbox, employing the MATLAB® toolkit by [19] for the sparse grid method. The process of numerical multilinearization in `mlinearize` consists of the following steps:

1. Creating the sparse grid
2. Evaluating the functions for $\dot{\mathbf{x}}$ and $\mu_{\mathbf{i}_i}$

3. Integrating the functions as quadrats in the sparse grid

The sparse grid in our implementation is the Smolyak grid (Fig. 3.6), with Clenshaw-Curtis knots (Fig. 3.3), the doubling level-to-knots function (3.9), and the multi-index rule (3.14). The functions $\mathbf{f}(\mathbf{x}, \mathbf{u})$ for $\dot{\mathbf{x}}$ and the monomial base functions $\mu_{\mathbf{i}_i}$ are evaluated by Simulink®. The integration of (3.5) is then performed on the sparse grid in domain D.

The `mlinearize` function takes in a Simulink® model, and the upper and lower bounds of operating values for states, inputs, and outputs of this model. By defining a range of operating values, this method eliminates the need for a single operating point, as is required for eigenvalue-based stability analysis in linear models. The user also defines the level of approximation of the sparse grid, and the limit on the number of variables multiplied in the monomial (multilinear order). The function creates a Matlab class object `mss`, which is then included in the Simulink® model using a level-2 S-function block for simulation.

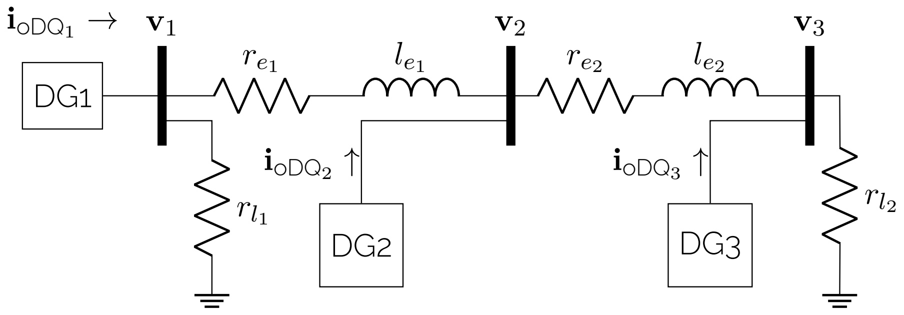
The number of states and inputs that the sparse grids tool can handle is limited by the memory and time requirement to set up the sparse grid and perform calculations. For up to 15 states and inputs, multilinearization takes less than a minute to complete in a standard laptop, but computational complexity increases nonlinearly with number of states. It is therefore useful to partition the system and identify recurring components to reduce the number of calculations, especially in larger systems where repeated structures may be common.

4.3 Power systems application examples

Three bus network model

The application of numerical multilinearization to a three-bus network is presented in [14]. The three-bus power system from [20, 21], as shown in Fig. 4.1, comprises three inverter-based generator buses, transmission lines, and purely resistive loads. Inverter and network parameters are adopted from [20]. The system operates at a nominal frequency of $50\,\mathrm{Hz}$ and a bus voltage of $380\,\mathrm{V}$.

Modeling is performed in the rotating DQ-frame; inverter DG1, Fig. 4.1, defines the global reference, while DG2 and DG3 are represented in local dq-frames. Transformations between global and local frames are done using a rotation matrix. Each inverter model includes 15 states, such as current, voltage, and controller variables. These models are split into linear and nonlinear components, the latter is being approximated here by mulitilinearization.



$\mathbf{i}_{\text{oDQ}_1} \rightarrow$
$\mathbf{v}_1$
$\mathbf{v}_2$
$\mathbf{v}_3$
r_{e_1}
l_{e_1}
r_{e_2}
l_{e_2}
DG1
$\mathbf{i}_{\text{oDQ}_2} \uparrow$
$\mathbf{i}_{\text{oDQ}_3} \uparrow$
r_{l_1}
r_{l_2}
DG2
DG3

Figure 4.1: Three bus network, [20].

Model Equations

The full model is represented as a set of differential-algebraic equations (DAEs), capturing both inverter dynamics and network constraints. The states of a single inverter are denoted by $\mathbf{x}_i$, one for each. The states of all inverters in the network are collected under, $\mathbf{x} = [\mathbf{x}_1^{\mathrm{T}}, \mathbf{x}_2^{\mathrm{T}}, \mathbf{x}_3^{\mathrm{T}}]^{\mathrm{T}}$. The network dynamics are represented by the two line currents $\mathbf{i} = [\mathbf{i}_1^{\mathrm{T}}, \mathbf{i}_2^{\mathrm{T}}]^{\mathrm{T}}$, and the three nodal voltages $\mathbf{v} = [\mathbf{v}_1^{\mathrm{T}}, \mathbf{v}_2^{\mathrm{T}}, \mathbf{v}_3^{\mathrm{T}}]^{\mathrm{T}}$, both in DQ-frame. Collecting all equations leads to the nonlinear model,

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{v}) \\ \mathbf{L}\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{i} &= -(\mathbf{R} + \mathbf{L}\boldsymbol{\mathcal{W}}_{\mathbf{i}}(\omega_1))\mathbf{i} + \mathbf{B}^{\mathrm{T}}\mathbf{v}, \\ \mathbf{v} &= \mathbf{R}_N(\mathbf{i}_{\text{oDQ}} - \mathbf{G}_{\text{L}}\mathbf{v} - \mathbf{B}\mathbf{i}), \end{aligned} \tag{4.1}$$

where, $\mathbf{L}$ and $\mathbf{R}$ hold the transmission line parameters, $\mathbf{G}_{\text{L}}$ holds the conductivity of the loads, $\mathbf{B}$ is a node-edge oriented incidence matrix representing the network graph, $\mathbf{R}_N$ models virtual resistors to ground at each node, ensuring that the node voltages are defined and properly conditioned for numerical solution, [20, 21]. The skew-symmetric matrix $\boldsymbol{\mathcal{W}}_{\mathbf{i}}(\omega_1)$ is an additional term that accounts for equations based on the rotating DQ-frame. The global frequency of the network ω_1, is dictated by the inverter in bus one of the network. The $\mathbf{i}_{\text{oDQ}}$ represent the collection of injected-inverter or output currents into the network nodes, expressed in global DQ-frame. Further details on the nonlinear model formulation can be found in [21, 8].

Multilinearization setup

Each inverter model is partitioned into a linear and nonlinear, the multilinear approximation is applied exclusively to the nonlinear part. The linear dynamics, are preserved in both cases. Multilinearization is done over the range defined by the minimum and maximum bounds of the state trajectories obtained from the nonlinear simulation, which starts from a non-equilibrium initial condition.

The nonlinear part is consists of 8 states and 7 inputs, resulting in $2^{(8+7)} = 32768$ coefficients per state to be identified. By limiting the maximum multilinear order to 2, this is reduced to just 121 coefficients per state according to (3.7). For integration truncation, a grid level of $\mathcal{L} = 4$ is used.

Simulation setup

To compare the nonlinear and multilinear models, both are simulated under identical conditions. The simulation begins from a non-equilibrium state, and at time $t = 0.05\,\mathrm{s}$, a sudden change in the conductance matrix is applied at buses one and three: bus one experiences a $20\,\%$ load decrease, while bus three sees a $100\,\%$ increase. Both models are simulated for $0.1\,\mathrm{s}$ using the MATLAB®/Simulink® `ode15i` solver with variable step size.

Results, Analysis

The simulation results for the three voltages at the buses is presented in Fig 4.2. It can be observed that the trajectories between the nonlinear and multilinear models are a good match, at all buses. Error metrics are presented in Table 4.1.

Table 4.1: Voltage trajectory errors per component.

Component	RMSE (V)	Abs. Max. Error (V)
v_{1d}	0.027	0.30
v_{1q}	0.020	0.18
v_{2d}	0.024	0.16
v_{2q}	0.020	0.19
v_{3d}	0.040	0.63
v_{3q}	0.018	0.16

The maximum errors occur immediately after the load step, where the multilinear and nonlinear models exhibit differences in peak response. Bus three exhibits the highest deviation in the d-axis component, with a peak of $0.63\,\mathrm{V}$, as the multilinear model slightly overestimates the response. The RMS errors across all buses range from $0.018\,\mathrm{V}$ to $0.04\,\mathrm{V}$, corresponding to less than $0.01\,\%$ of the nominal $380\,\mathrm{V}$. Overall, the errors indicate that the multilinear model closely approximates the nonlinear dynamics, particularly outside fast transients. While errors peak at the load-affected buses, their magnitude is sufficiently low to encourage testing the multilinear model for control and other application-relevant scenarios.

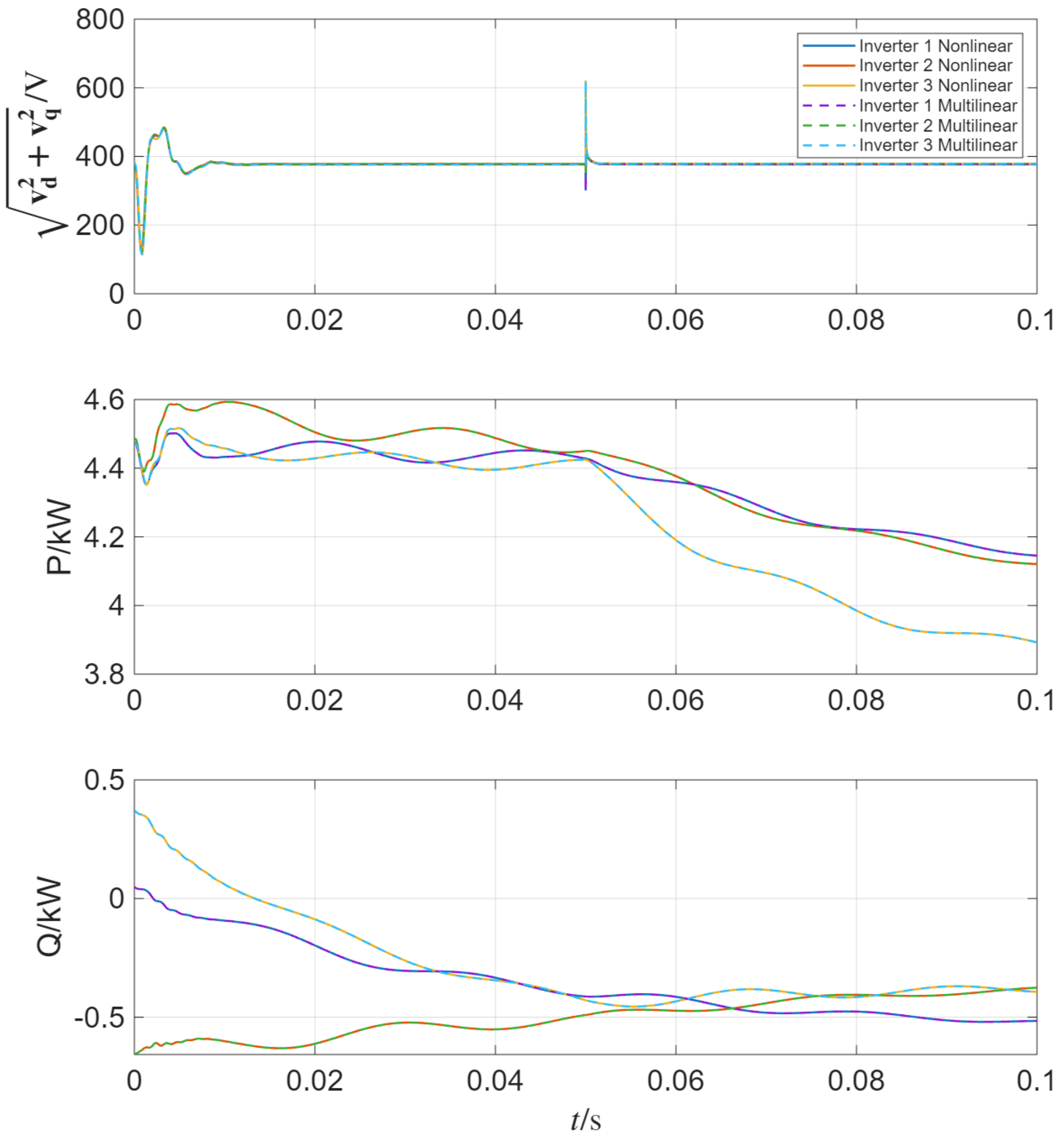


Figure 4.2: Node voltages, active power P and reactive power Q.

Grid following converter

Another illustrative example is the multilinearization of a grid-following converter. The implementation in Simulink® is provided in `demo_mlinearize.mlx` of the MTI-Toolbox Demo. The grid following converter is part of an essential system described in the demo `DemoSSA.mlx` in the Toolbox [22]. It consists of a grid-following converter, a grid-forming converter, and a load, all of them connected to a Thévenin-equivalent grid. For a more detailed description of model and simulation setup, please refer to [22].

The structure of the grid-following converter control is shown Fig. 4.3, where the nonlinearities including multiliniear functions in the grid-following converter are partitioned into a subsystem for multilinearization.

The nonlinear subsystem has 4 states and 4 inputs. The maximum multilinear order is limited to 3, which reduces the coefficients per state from 256 to 93. The sparse grid is set up with a grid level of $\mathcal{L} = 4$.

The multilinear subsystem is connected in parallel to the original nonlinear subsystem, so they are simulated under identical conditions for comparison. The simulation is set up to run for 1 s, during which the control modes of the neighbouring grid-forming converter are switched at $t = 0.2$, and occurs at $t = 0.8$. The resulting node voltages are shown in Fig. 4.4. While the voltages immediately after the signal deviate from the nonlinear simulation, the quasi-steady state values are close to those of the implicit simulation.

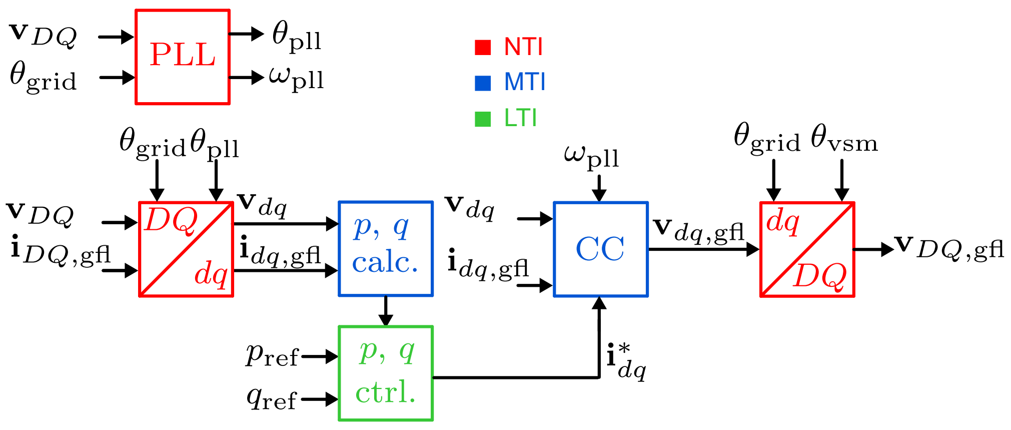


Figure 4.3: Grid-following converter's control structure [22] with separation into NTI, MTI, and linear time-invariant (LTI) models

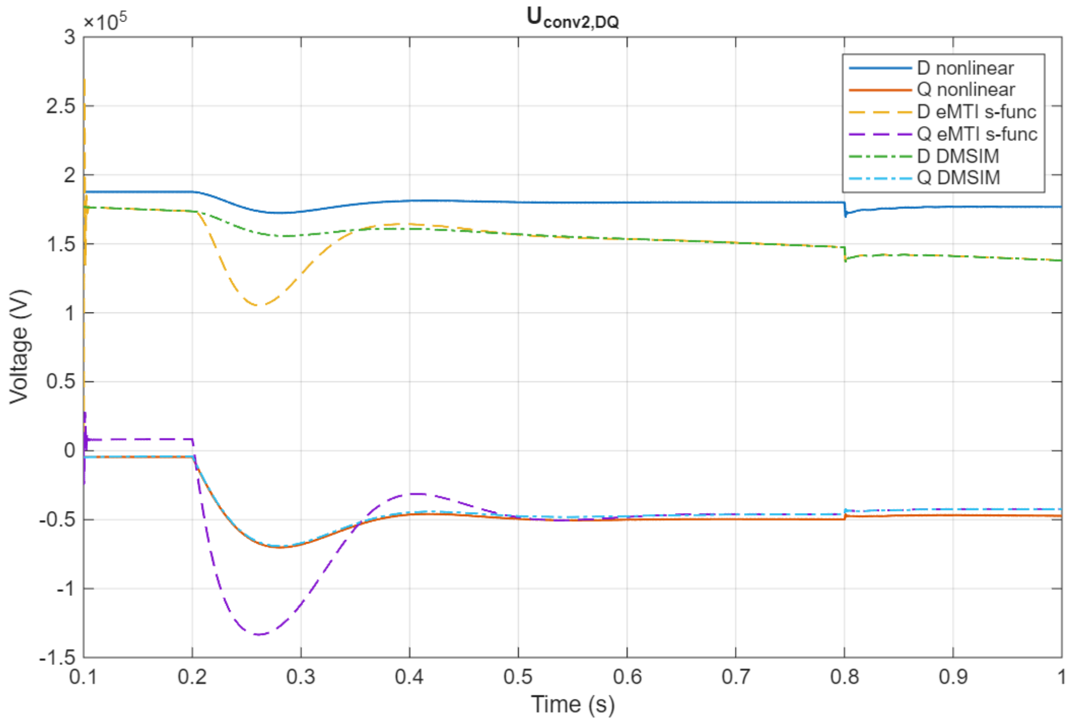


Figure 4.4: Node voltages of nonlinear simulation (solid line), multilinear simulation with numerical multilinearization (dashed line), and with implicit multilinearization (dash-dot line).

5 Conclusion

This report introduced core concepts of MTI modeling, and presented an improved multilinearization algorithm for approximating nonlinear dynamical systems as explicit MTI models. The application to power system components demonstrates that despite the model approximation, the multilinear models closely matched the nonlinear simulation, particularly in steady-state and post-transient behavior, i.e., the operating points that form the basis for eigenvalue-based stability analysis.

5.1 Outlook

The multilinearization approaches currently available in the MTI-Toolbox is shown in the green arrows in Fig. 1.1. Extending on the multilinearization methods for the MTI-Toolbox to include linear transformation for both the explicit and implicit model classes would provide another way of accurately representing nonlinear models with multilinear models.

The numerical multilinearization method discussed in this report is suited for the explicit model class. Developing numerical multilinearization for the implicit class would be beneficial as more parameters as well as discrete states can be included in the multilinearization. With the implementation on Simulink®, nonlinearities not represented analytically or occurring in grey-box or black-box models can be multilinearized for further stability analysis. The results from this deliverable will aid in the development of system identification capabilities in the future tasks of this project.

Bibliography

[1] Christoph Kaufmann, Carlos Cateriano Yáñez, and Sarah Casura. *D 1.1 Hybrid Implicit Multilinear Model Class and Minimal Format*. TenSyGrid deliverable 1.1. Fraunhofer IWES, Nov. 2025.

[2] 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 - SIMULTECH*. 2013, pp. 275–285. ISBN: 978-989-8565-69-3. DOI: 10.5220/0004475602750285.

[3] Kai Kruppa, Georg Pangalos, and Gerwald Lichtenberg. "Multilinear Approximation of Nonlinear State Space Models". In: *IFAC Proceedings Volumes* 47.3 (2014). 19th IFAC World Congress, pp. 9474–9479. ISSN: 1474-6670. DOI: https://doi.org/10.3182/20140824-6-ZA-1003.00455.

[4] Gerwald Lichtenberg et al. "Implicit multilinear modeling". In: *at - Automatisierungstechnik* 70.1 (2022), pp. 13–30. DOI: doi:10.1515/auto-2021-0133.

[5] Torben Warnecke and Gerwald Lichtenberg. "Implicit Multilinear Modeling of Air Conditioning Systems". In: *Proceedings of the 13th International Conference on Simulation and Modeling Methodologies, Technologies and Applications - SIMULTECH*. 2023, pp. 440–447. ISBN: 978-989-758-668-2. DOI: 10.5220/0012138200003546.

[6] G. Lichtenberg et al. *MTI-Toolbox*. Online. 2024. URL: mti.systems.

[7] Christoph Kaufmann et al. "Semi-explicit multilinear modeling of a PQ open-loop controlled PV inverter in $\alpha\beta$-frame". In: *Energy Reports* 9 (2023). 2022 The 3rd International Conference on Power, Energy and Electrical Engineering, pp. 354–360. ISSN: 2352-4847. DOI: https://doi.org/10.1016/j.egyr.2022.12.135.

[8] Leandro Samaniego et al. "An Approach to Multi-Energy Network Modeling by Multilinear Models". In: *2024 European Control Conference (ECC)*. 2024, pp. 1522–1527. DOI: 10.23919/ECC64448.2024.10590975.

[9] Pablo de Juan Vela and Josep Fanals i Batllori. *D 3.1 Report on Identified Non-linearities and Saturations*. TenSyGrid deliverable 3.1. eRoots, May 2025. URL: https://tensygrid.eu/wp-content/uploads/2025/06/D3.1-Non-linearities-and-saturations_final_ERoots.pdf.

[10] Pablo de Juan Vela et al. *D 3.2 Generation Multilinear Models and Validation*. TenSyGrid deliverable 3.2. eRoots, Nov. 2025.

[11] Simon Wallgram. "Multilinearization of polynomial models by linear state transformation". MA thesis. Technische Universität Hamburg, 2025.

[12] D. Kincaid and E. Cheney. *Numerical Analysis: Mathematics of Scientific Computing*. Brooks/Cole Publications, 1996.

[13] Aline Luxa, Richard Hanke-Rauschenbach, and Gerwald Lichtenberg. “Model predictive supervisory control for multi-stack electrolyzers using multilinear modeling”. In: *International Journal of Hydrogen Energy* 185 (2025), p. 151847. ISSN: 0360-3199. DOI: `https://doi.org/10.1016/j.ijhydene.2025.151847`. URL: `https://www.sciencedirect.com/science/article/pii/S0360319925048505`.

[14] Teresa Wong et al. “Numerical Multilinearization of nonlinear models by sparse grids method”. In: *2025 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)*. IEEE, 2025.

[15] Chiara Piazzola and Lorenzo Tamellini. *The Sparse Grids Matlab Kit user manual - v. 23-5 Robert*. https://sites.google.com/view/sparse-grids-kit.

[16] C. Runge. “Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten.” In: *Z. Math. Phys.* 46 (1901), pp. 224–243.

[17] Lloyd N. Trefethen. “Exactness of Quadrature Formulas”. In: *SIAM Review* 64.1 (2022), pp. 132–150. DOI: `10.1137/20M1389522`.

[18] Justin Gregory Winokur. “Adaptive Sparse Grid Approaches to Polynomial Chaos Expansions for Uncertainty Quantication.” PhD thesis. Duke University, 2015.

[19] Chiara Piazzola and Lorenzo Tamellini. “Algorithm 1040: The Sparse Grids Matlab Kit - a Matlab implementation of sparse grids for high-dimensional function approximation and uncertainty quantification”. In: *ACM Trans. Math. Softw.* 50.1 (Mar. 2024). ISSN: 0098-3500. DOI: `10.1145/3630023`.

[20] Nagaraju Pogaku, Milan Prodanovic, and Timothy C. Green. “Modeling, Analysis and Testing of Autonomous Operation of an Inverter-Based Microgrid”. In: *IEEE Transactions on Power Electronics* 22.2 (2007), pp. 613–625. DOI: `10.1109/TPEL.2006.890003`.

[21] Mahmoud Kabalan, Pritpal Singh, and Dagmar Niebur. “A Design and Optimization Tool for Inverter-Based Microgrids Using Large-Signal Nonlinear Analysis”. In: *IEEE Transactions on Smart Grid* 10.4 (2019), pp. 4566–4576. DOI: `10.1109/TSG.2018.2864208`.

[22] Christoph Kaufmann et al. *Small-Signal Stability Analysis of Power Systems by Implicit Multilinear Models*. 2025. arXiv: `2510.16534 [eess.SY]`. URL: `https://arxiv.org/abs/2510.16534`.

Acknowledgement

The authors would like to thank Leandro Samaniego for contributions and discussions on numerical multilinearization.