Solving Mixed-Integer Programming Problems Using Piecewise Linearization Methods

Bachelor Thesis

by

Bernreuther, Marco
Reg. No. 01/897074

at the

Universität Konstanz

Department of Mathematics and Statistics

(i) Evaluated by Prof. Dr. Stefan Volkwein

Konstanz, 2017

Contents

Abstract

We present an overview on piecewise linearization methods for MINLPs. This will include the concept of disjunctive constraints, which is necessary to define logarithmic reformulations of the so called disaggregated convex combination method and the convex combination method. For the case of a general univariate quadratic function we also calculate the linearization error and proof that equidistant grid points minimize this error. For a bivariate product of two variables we do the same error analysis for the case of J_1-triangulations and again equidistant grid points will be optimal. The presented methods will then be applied to a newly developed model for a hybrid energy supply network problem. We show that linearizations are able to solve this non-convex optimization problem within reasonable time.

Zusammenfassung

Wir geben einen Überblick zu stückweisen Linearisierungsmethoden für MINLPs. Insbesondere wird dabei das Konzept der Disjunctive Constraints vorgestellt, welche notwendig sind um logarithmische Umformulierungen der sogenannten Disaggregated Concex Combination Methode und der Convex Combination Methode durchzuführen. Für den Fall einer univariaten quadratischen Funktion berechnen wir außerdem den Linearisierungsfehler und beweisen, dass eine äquidistante Wahl der Gitterpunkte der Triangulierung diesen minimiert. Für das bivariate Produkt zweier Variablen werden wir dieselbe Fehleranalyse für den Fall, dass eine J_1-Triangulierung gegeben ist, durchführen und erneut zeigen, dass äquidistante Gitterpunkte optimal sind. Danach werden die eingeführten Methoden auf das neu entwickelte Modell eines hybriden Energieversorgungsnetzwerks angewendet. Wir werden zeigen, dass dieses Problem mit Hilfe von stückweisen Linearisierungen in kurzer Zeit gelöst werden kann.

1 Introduction

Many real-life applications of optimization problems naturally combine binary decisions and continuous (nonlinear) constraints. Energy network problems, e.g. for gas or water networks, optimizing automatic transmission in cars, or the structure of an industrial assembly line are but one of many examples. Those problems belong to the general class of mixed-integer nonlinear programs (MINLPs), which is hard to solve. Meaning that no polynomial-time algorithm exists for general MINLPs. Both the binary decisions and the nonlinearities lead to NP-hard problems in general [Geißler, 2011]. Also the wide variety of MINLPs makes it difficult to find general algorithms that converge to a global optimum.

One solution to this problem is to approximate a MINLP by a mixed-integer linear program (MILP). This class of optimization problems is as already mentioned still NP-hard, but plenty of research has been made on fast algorithms and development of state-of-the-art solvers for MIPs.

The basic principle behind those algorithms is (almost always) the Branch-and-Bound (B&B) algorithm, which does not have to search the whole binary decision tree of the MIP to find an optimal solution, but can cut off branches if no better or no solution at all can lie in those branches. Therefore B&B converges with an optimal solution and that often very fast. At each node of the decision tree a linear program (LP) has to be solved. Possible algorithms for that are interior point, which converges in polynomial time, or simplex, which does not converge in polynomial time, but provides good options for warm starts. For more information on B&B for MIPs see for example [Wolsey, 1998].

A general MINLP can be described via

$$\begin{aligned}
\underset{x}{\text{minimize}} \quad & c^T x && (1.1)\\
\text{subject to} \quad & g_i\left(x\right) = b_i, \quad i = 1, \dots, k_1 && (1.2)\\
& h_i\left(x\right) \geq b_i, \quad i = 1, \dots, k_2 && (1.3)\\
& l \leq x \leq u && (1.4)\\
& x \in \mathbb{R}^{d-p} \times \mathbb{Z}^p && (1.5)
\end{aligned}$$

with $k_1, k_2, d, p \in \mathbb{N}$, $g_i, h_i \in C^0\left(\mathbb{R}^d, \mathbb{R}\right)$ and $l, u \in \mathbb{R}^d$ (box constraints). We want to point out that the feasible set defined via equation (1.2) - (1.5) does not have to be convex and might be empty. Also we want to mention that even though the objective function in equation (1.1) is linear, this is no restriction to the generality of the MINLP, because we can always replace

$$\underset{x}{\text{minimize}} \quad f\left(x\right)$$

with

$$\begin{aligned}
\underset{x,y}{\text{minimize}} \quad & y\\
\text{subject to} \quad & y = f\left(x\right).
\end{aligned}$$

The idea of this work will be to develop a framework for the approximation of a MINLP by a MILP and the computation of this MILP in Section 3. Before that the concept of disjunctive constraints which provides the theoretical basics for some of the MILP approximations in Section 3 will be introduced in Section 2. Then an application to energy supply networks will be discussed in Section 4. Section 3 also deals with the error analysis necessary for the application in Section 4.

2 Disjunctive Constraints

As preparation for Section 3 we are going to take a closer look at disjunctive constraints in general, before the special cases of SOS1 and SOS2 constraints are introduced. We mainly follow Section 2 in [Vielma and Nemhauser, 2011], but only present results that are directly related to our case. For a finite index set J let $\Delta^J := \left\{\lambda \in \mathbb{R}_+^{|J|} \,:\, \sum_{j\in J} \lambda_j \leq 1\right\}$ be the $|J|$-dimensional standard simplex. For a vector $\lambda \in \Delta^J$ we only allow certain subsets of components to be non-zero. This can be modeled by introducing a finite set I indexing $S_i \subset J$ and a function defined via $Q\left(S_i\right) := \left\{\lambda \in \Delta^J \,:\, \lambda_j \leq 0 \ \forall j \notin S_i\right\}$. The constraint on λ can be represented as

$$\lambda \in \bigcup_{i\in I} Q\left(S_i\right) \subset \Delta^J. \tag{2.1}$$

Two special cases are that either at most one component of λ can be non-zero, this is called a special ordered set of type 1 (SOS1) constraint, or that at most two components of λ can be non-zero and if so their indices must be adjacent, this is called a SOS2 constraint. With the help of (2.1) those constraints can be formally introduced:

Definition 2.1

(i) A SOS1 constraint on $\lambda \in \mathbb{R}_+^n$ is of the form (2.1) with $I = J = \{1, \dots, n\}$ and $S_i = \{i\}$.

(ii) A SOS2 constraint on $\lambda \in \mathbb{R}_+^{n+1}$ is of the form (2.1) with $J = \{0, \dots, n\}$, $I = J \setminus \{0\}$ and $S_i = \{i-1, i\}$.

An easy way to formulate (2.1) as a mixed-integer formulation is to assign one binary variable to each set $Q\left(S_i\right)$. This result is summarized in the following

Proposition 2.2 (Proposition 1 in [Vielma and Nemhauser, 2011], Section 2)
An equivalent MILP formulation for (2.1) is given by
$\lambda \in \Delta^J$, $\lambda_j \leq \sum_{i\in I(j)} x_i \ \forall j \in J$, $\sum_{i\in I} x_i = 1$, $x_i \in \{0,1\} \ \forall i \in I$,
where $I(j) := \{i \in I : \ j \in S_i\}$. This formulation has $|I|$ binary variables and $|J| + 1$ additional constraints. (Additional to $\lambda \in \Delta^J$.)

Our goal is to reduce the number of binary variables for the case of SOS1 and SOS2 constraints. For SOS1 we observe that since at most one component is non-zero at a time and there are $n \in \mathbb{N}$ components, we essentially have to encode n different states. This is possible with only $\lceil \log_2(n) \rceil$ binary variables.

Proposition 2.3 (Proposition 2 in [Vielma and Nemhauser, 2011], Section 2)
Let $B : \ I \to \{0,1\}^{\lceil \log_2 |I| \rceil}$ be an injective function. Then an equivalent formulation for Proposition 2.2 for SOS1 constraints is given by
$\lambda \in \Delta^J$, $\sum_{j\in J^+(l,B)} \lambda_j \leq x_l$, $\sum_{j\in J^0(l,B)} \lambda_j \leq (1 - x_l)$, $x_l \in \{0,1\} \ \forall l \in L(|I|)$, *with* $L(r) := \{1, \dots, \lceil \log_2(r) \rceil\}$, $J^+(l,B) := \{j \in J : \ l \in \sigma(B(j))\}$, $J^0(l,B) := \{j \in J : \ l \notin \sigma(B(j))\}$ *and $\sigma(B)$ is the support vector of B.*
This formulation has $\lceil \log_2 |I| \rceil$ binary variables and $2\lceil \log_2 |I| \rceil$ additional constraints.

Note that the choice of B is arbitrary for SOS1 constraints. In order to extend this Proposition to SOS2 constraints, one either has to add additional constraints for an arbitrary B in 2.3 or use a specially constructed B to begin with. More details on that together with some examples can be found in Section 2 in [Vielma and Nemhauser, 2011].

Such a specially constructed B is defined as follows:

Definition 2.4 (SOS2 Compatible Function)
Let $B: I \to \{0,1\}^{\lceil \log_2 |I| \rceil}$ be an injective function. Then we call B compatible with a SOS2 constraint on $\lambda \in \mathbb{R}_+^{n+1}$ iff for all $i \in \{1, \dots, n-1\}$ the vectors $B(i)$ and $B(i+1)$ differ in at most one component.

For SOS2 compatible functions we get

Theorem 2.5 (Theorem 1 in [Vielma and Nemhauser, 2011], Section 2)
Let B be a SOS2 compatible function, then Proposition 2.3 also holds for SOS2 constraints.

With the help of those logarithmic reformulations for SOS1 and SOS2 constraints, we will be able to introduce reformulations for some piecewise linearization methods in section 3. Also examples for SOS2 compatible functions and the application of Theorem 2.5 will be given there.

3 Piecewise Linearization Methods for solving MINLPs

The general idea is to use the advantage that MILP has over MINLP, in order to find approximate (MILP) solutions for a MINLP. The first step will be to formally introduce a triangulation - for simplicity on a n-dimensional rectangle - and a piecewise linear approximation of a nonlinear function, defined on that rectangle. Then different approaches to integrate a piecewise linear function into the constraints are discussed. In the course of this the number of additional binary variables and possibilities to reduce them, will be dealt with in detail. Afterwards a short error analysis that focuses on the nonlinearities in the examples from Section 4 follows.

3.1 Piecewise Linear Approximation of general nonlinear functions

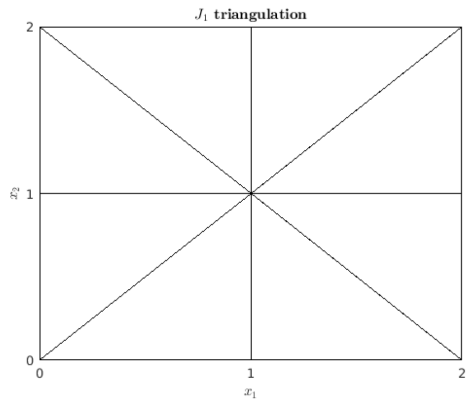
Let a nonlinear function $f(x) = z$ with $x \in D$ ($f \le z$ and $f < z$ work analogously) and $z \in \mathbb{R}$ be given, which is defined on a n-dimensional rectangle D. Then a triangulation is defined as follows:

Definition 3.1
Let $D \subset \mathbb{R}^n$ be a n-dimensional rectangle and $f \in C^0(D, \mathbb{R})$ be a nonlinear function. A triangulation of D is a finite family $\mathscr{T}$ of simplices T with

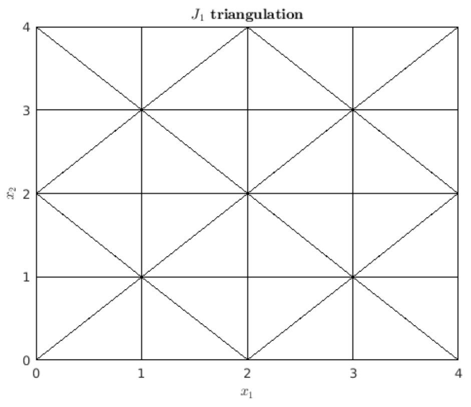
$$D = \bigcup_{T \in \mathscr{T}} T$$

and for T_1, $T_2 \in \mathscr{T}$, $T_1 \cap T_2$ is either empty or a common facet of T_1 and T_2.

A special triangulation on the two-dimensional rectangle $[0, \omega]^2$ with $\omega \in 2\mathbb{N}$ ($2\mathbb{N}$ are the even natural numbers) is the so called "Union Jack" or J_1 triangulation, which is shown in Figure 1a for $\omega = 2$ and in Figure 1b for $\omega = 4$. A formal definition and properties can be found e.g. in [Todd, 1974]. The J_1 triangulation is going to be necessary for some linearization methods in Section 3.2.



(A) J_1 triangulation for $\omega = 2$.



(B) J_1 triangulation for $\omega = 4$.

With the help of Definition 3.1 we can now introduce a piecewise linear approximation of f:

Definition 3.2
Given a triangulation $\mathscr{T}$ of D, a piecewise linear approximation of f is a function $\phi \in C^0(D, \mathbb{R})$ s.t. for all $T \in \mathscr{T}$: $\phi_T = \phi$ and for all $v \in \mathscr{V}(T) : \phi_T(v) = f(v)$, whereas ϕ_T is a linear function and $\mathscr{V}(T)$ is the vertice set of T.

Note that the original function and its piecewise linear approximation have the same values on the vertices $\mathscr{V}(\mathscr{T})$ of the triangulation. We now approximate $f(x) = z$ with $\phi(x) = z$. A one-dimensional example with $f(x) = x|x|$ is given in Figure 2.

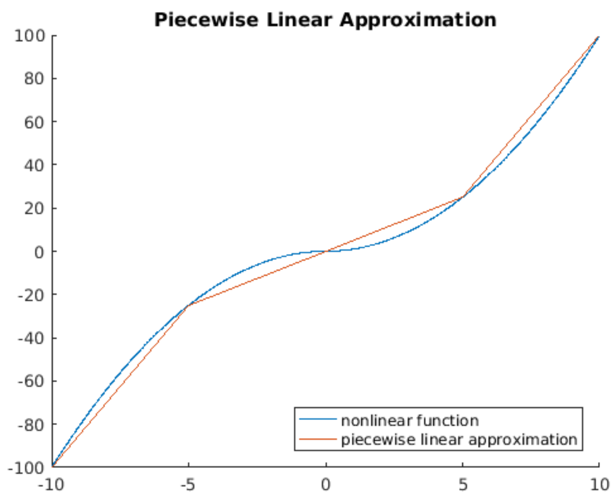


Figure 2: Piecewise linear approximation of $f(x) = |x|x$ on $D = [-10, 10]$.

The piecewise linear approximation ϕ of f on $[-10, 10]$ is given by

$$\phi(x) = \begin{cases} 15x + 50 & , x \in [-10, -5] \\ 5x & , x \in [-5, 0] \\ 5x & , x \in [0, 5] \\ 15x - 50 & , x \in [5, 10] \end{cases} . \tag{3.1}$$

It can be seen that on $\mathbb{R}^- = (-\infty, 0]$ the approximation ϕ underestimates f and on $\mathbb{R}^+ = [0, \infty)$ the approximation ϕ overestimates f. The interesting thing to notice is that f is concave on $\mathbb{R}^-$ and convex on $\mathbb{R}^+$. This will be important for an error analysis on f as will be done in Section 3.3.

It should also be mentioned that our MILP approximations are (in general) no relaxation of the original MINLP, but as shown in [Geißler, 2011] the MILP approximations can be used to generate MILP relaxations without adding additional auxiliary variables to the problem. Though this might seem desirable it also comes with the disadvantage, that particular properties of the nonlinear function can be lost in this process [Geißler, 2011].

3.2 Piecewise Linearization Methods

In the following sections we assume that the piecewise linear approximation ϕ of f is already at hand and different formulations of $\phi(x)$ in the literature will be presented. We mainly follow the work of [Vielma et al., 2010]. Note that for the so called LogCC method we will restrict ourself to the two-dimensional case.

3.2.1 Disaggregated Convex Combination Method (DCC)

One easy approach to represent ϕ is to introduce continuous auxiliary variables $\lambda_{T,v} \geq 0$ for each point v of every simplex T and binary auxiliary variables y_T for every simplex. To ensure that only one simplex T_1 is active in the calculation of x and z, the constraint $\sum_{T \in \mathscr{T}} y_T = 1$ is introduced. In T_1, x and z are build as convex combinations with the help of $\lambda_{T_1,v}$. The constraints $\lambda_{T,v} \geq 0$ and $\sum_{v \in V(T)} \lambda_{T,v} = y_T$ for all $T \in \mathscr{T}$ ensure $x \in T_1$ and $\lambda_{T,v} = 0$ for $T \neq T_1$.
This representation is called Disaggregated Convex Combination (DCC) method. That is because more than one $\lambda_{T,v}$ is associated with each v and therefore the continuous auxiliary variables are “disaggregated”.
All together this gives:

$$\sum_{T \in \mathscr{T}} \sum_{v \in V(T)} \lambda_{T,v} v = x, \qquad \sum_{T \in \mathscr{T}} \sum_{v \in V(T)} \lambda_{T,v} f(v) = z, \tag{3.2}$$

$$\lambda_{T,v} \geq 0 \qquad \forall T \in \mathscr{T}, v \in V(T), \tag{3.3}$$

$$\sum_{v \in V(T)} \lambda_{T,v} = y_T \qquad \forall T \in \mathscr{T}, \tag{3.4}$$

$$\sum_{T \in \mathscr{T}} y_T = 1, \qquad y_T \in \{0,1\} \, \forall T \in \mathscr{T}. \tag{3.5}$$

For a better understanding, we will consider a 1-dimensional

Example (DCC)
Consider the piecewise linear function given in equation (3.1).
We have $T_1 = [-10, -5]$, $T_2 = [-5, 0]$, $T_3 = [0, 5]$, $T_4 = [5, 10]$ *with* $V(T_1) = \{-10, -5\}$, $V(T_2) = \{-5, 0\}$, $V(T_3) = \{0, 5\}$, $V(T_4) = \{5, 10\}$ *and* $\mathscr{V}(\mathscr{T}) = \{-10, -5, 0, 5, 10\}$. *We note that due to point symmetry of* ϕ *we could use one simplex* $T_2 \cup T_3$ *instead of* T_2, T_3, *but since this is only possible because* T_2 *and* T_3 *are intervals of the same length with common vertice* 0, *we will not make use of that.*

DCC yields the following constraints:

$$\begin{aligned}
-10\lambda_{T_1,-10} - 5\lambda_{T_1,-5} - 5\lambda_{T_2,-5} + 0\lambda_{T_2,0} + 0\lambda_{T_3,0} + 5\lambda_{T_3,5} + 5\lambda_{T_4,5} + 10\lambda_{T_4,10} &= x,\\
-100\lambda_{T_1,-10} - 25\lambda_{T_1,-5} - 25\lambda_{T_2,-5} + 0\lambda_{T_2,0} + 0\lambda_{T_3,0} + 25\lambda_{T_3,5} + 25\lambda_{T_4,5} + 100\lambda_{T_4,10} &= z,\\
\lambda_{T_1,-10}, \lambda_{T_1,-5}, \lambda_{T_2,-5}, \lambda_{T_2,0}, \lambda_{T_3,0}, \lambda_{T_3,5}, \lambda_{T_4,5}, \lambda_{T_4,10} &\geq 0,\\
\lambda_{T_1,-10} + \lambda_{T_1,-5} &= y_{T_1},\\
\lambda_{T_2,-5} + \lambda_{T_2,0} &= y_{T_2},\\
\lambda_{T_3,0} + \lambda_{T_3,5} &= y_{T_3},\\
\lambda_{T_4,5} + \lambda_{T_4,10} &= y_{T_4},\\
y_{T_1} + y_{T_2} + y_{T_3} + y_{T_4} &= 1,\\
y_{T_1}, y_{T_2}, y_{T_3}, y_{T_4} &\in \{0,1\}
\end{aligned}$$

For $y_{T_i} = 1$ we have $y_{T_j} = 0$ for $j \in \{1,2,3,4\} \setminus \{i\}$, thus only one simplex is active for the convex combination. This argumentation is independent of the dimension and triangulation and therefore always holds for DCC.

3.2.2 Convex Combination Method (CC)

A natural modification of DCC is the Convex Combination Method (CC), which aggregates the continuous variables of DCC by introducing only one auxiliary continuous variable $\lambda_v \geq 0$ for each $v \in \mathscr{V}(\mathscr{T})$. The auxiliary binary variables stay the same. Let $\mathscr{T}(v) := \{T \in \mathscr{T} : v \in T\}$ denote the set of all simplices with vertex v. Again only one simplex is active, but now $\lambda_v \leq \sum_{T\in\mathscr{T}(v)} y_T$ for all $v \in \mathscr{V}(\mathscr{T})$ ensures that only those λ's whose indexes are vertices of the active simplex, can be chosen. Furthermore $\sum_{v\in\mathscr{V}(\mathscr{T})} \lambda_v = 1$ and $\lambda_v \geq 0$ ensure the convex combination for x and z.
All together this gives:

$$\sum_{v\in\mathscr{V}(\mathscr{T})} \lambda_v v = x, \qquad \sum_{v\in\mathscr{V}(\mathscr{T})} \lambda_v f(v) = z, \tag{3.6}$$

$$\lambda_v \geq 0 \qquad \forall v \in \mathscr{V}(\mathscr{T}), \qquad \sum_{v\in\mathscr{V}(\mathscr{T})} \lambda_v = 1, \tag{3.7}$$

$$\lambda_v \leq \sum_{T\in\mathscr{T}(v)} y_T \qquad \forall v \in \mathscr{V}(\mathscr{T}), \tag{3.8}$$

$$\sum_{T\in\mathscr{T}} y_T = 1, \qquad y_T \in \{0,1\}\, \forall T \in \mathscr{T}. \tag{3.9}$$

Example (CC)
For the previous example CC yields the following constraints:

$$\begin{aligned}
-10\lambda_{-10} - 5\lambda_{-5} + 0\lambda_0 + 5\lambda_5 + 10\lambda_{10} &= x,\\
-100\lambda_{-10} - 25\lambda_{-5} + 0\lambda_0 + 25\lambda_5 + 100\lambda_{10} &= z,\\
\lambda_{-10}, \lambda_{-5}, \lambda_0, \lambda_5, \lambda_{10} &\geq 0,\\
\lambda_{-10} + \lambda_{-5} + \lambda_0 + \lambda_5 + \lambda_{10} &= 1,\\
\lambda_{-10} &\leq y_{T_1},\\
\lambda_{-5} &\leq y_{T_1} + y_{T_2},\\
\lambda_0 &\leq y_{T_2} + y_{T_3},\\
\lambda_5 &\leq y_{T_3} + y_{T_4},\\
\lambda_{10} &\leq y_{T_4},\\
y_{T_1} + y_{T_2} + y_{T_3} + y_{T_4} &= 1,\\
y_{T_1}, y_{T_2}, y_{T_3}, y_{T_4} &\in \{0,1\}
\end{aligned}$$

For $y_{T_1} = 1$, $y_{T_2}, y_{T_3}, y_{T_4} = 0$, we force	For $y_{T_2} = 1$, $y_{T_1}, y_{T_3}, y_{T_4} = 0$, we force	For $y_{T_3} = 1$, $y_{T_1}, y_{T_2}, y_{T_4} = 0$, we force	For $y_{T_4} = 1$, $y_{T_1}, y_{T_2}, y_{T_3} = 0$, we force
$\lambda_{-10}, \lambda_{-5} \leq 1$, $\lambda_0, \lambda_5, \lambda_{10} \leq 0.$	$\lambda_{-5}, \lambda_0 \leq 1$, $\lambda_{-10}, \lambda_5, \lambda_{10} \leq 0.$	$\lambda_0, \lambda_5 \leq 1$, $\lambda_{-10}, \lambda_{-5}, \lambda_{10} \leq 0.$	$\lambda_5, \lambda_{10} \leq 1$, $\lambda_{-10}, \lambda_{-5}, \lambda_0 \leq 0.$

This means, that the constraints on λ are SOS2. This is only true in dimension $n = 1$, because for $n \geq 2$ no unique ordering is imposed on the simplices.

3.2.3 Logarithmic Models

With the help of Section 2 we are now able to logarithmically reduce the number of auxiliary binary variables for DCC and CC. In DCC binaries are used to force that only one simplex is active at a time. This is nothing but a SOS1 constraint on a binary vector. We choose $I = J = \{1, \dots, |\mathscr{T}|\}$ in Definition 2.1. Therefore we can apply Proposition 2.3 and get the following logarithmic reformulation (DLog) for DCC:

$$\sum_{T \in \mathscr{T}} \sum_{v \in V(T)} \lambda_{T,v} v = x, \qquad \sum_{T \in \mathscr{T}} \sum_{v \in V(T)} \lambda_{T,v} f(v) = z, \tag{3.10}$$

$$\lambda_{T,v} \geq 0 \qquad \forall T \in \mathscr{T}, v \in V(T), \tag{3.11}$$

$$\sum_{T \in \mathscr{T}} \sum_{v \in V(T)} \lambda_{T,v} = 1, \tag{3.12}$$

$$\sum_{T \in J^+(l,B)} \sum_{v \in V(T)} \lambda_{T,v} \leq x_l, \tag{3.13}$$

$$\sum_{T \in J^0(l,B)} \sum_{v \in V(T)} \lambda_{T,v} \leq (1 - x_l), \; x_l \in \{0,1\} \, \forall l \in L(|\mathscr{T}|). \tag{3.14}$$

Where L, B, J^0, J^+ are as defined in Proposition 2.3

Again we will consider the previous

Example (DLog)
For the SOS2 compatible function B defined via $B(1) = (0,0)^T$, $B(2) = (0,1)^T$, $B(3) = (1,1)^T$ and $B(4) = (1,0)^T$ DLog yields the following constraints:

$$\begin{aligned}
-10\lambda_{T_1,-10} - 5\lambda_{T_1,-5} - 5\lambda_{T_2,-5} + 0\lambda_{T_2,0} + 0\lambda_{T_3,0} + 5\lambda_{T_3,5} + 5\lambda_{T_4,5} + 10\lambda_{T_4,10} = x,\\
-100\lambda_{T_1,-10} - 25\lambda_{T_1,-5} - 25\lambda_{T_2,-5} + 0\lambda_{T_2,0} + 0\lambda_{T_3,0} + 25\lambda_{T_3,5} + 25\lambda_{T_4,5} + 100\lambda_{T_4,10} = z,\\
\lambda_{T_1,-10}, \lambda_{T_1,-5}, \lambda_{T_2,-5}, \lambda_{T_2,0}, \lambda_{T_3,0}, \lambda_{T_3,5}, \lambda_{T_4,5}, \lambda_{T_4,10} \geq 0,\\
\lambda_{T_1,-10} + \lambda_{T_1,-5} + \lambda_{T_2,-5} + \lambda_{T_2,0} + \lambda_{T_3,0} + \lambda_{T_3,5} + \lambda_{T_4,5} + \lambda_{T_4,10} = 1,\\
\lambda_{T_3,0} + \lambda_{T_3,5} + \lambda_{T_4,5} + \lambda_{T_4,10} \leq x_1, \quad \lambda_{T_1,-10} + \lambda_{T_1,-5} + \lambda_{T_2,-5} + \lambda_{T_2,0} \leq (1 - x_1),\\
\lambda_{T_2,-5} + \lambda_{T_2,0} + \lambda_{T_3,0} + \lambda_{T_3,5} \leq x_2, \quad \lambda_{T_1,-10} + \lambda_{T_1,-5} + \lambda_{T_4,5} + \lambda_{T_4,10} \leq (1 - x_2),\\
x_1, x_2 \in \{0,1\}.
\end{aligned}$$

Therefore in every case exactly one simplex is active for the convex combination as desired.

For $x_1 = 0$, $x_2 = 0$, we force

$$\lambda_{T_1,-10}, \lambda_{T_1,-5} \le 1,$$
$$\lambda_{T_2,-5}, \lambda_{T_2,0}, \lambda_{T_3,0}, \lambda_{T_3,5}, \lambda_{T_4,5}, \lambda_{T_4,10} \le 0.$$

For $x_1 = 1$, $x_2 = 0$, we force

$$\lambda_{T_4,5}, \lambda_{T_4,10} \le 1,$$
$$\lambda_{T_1,-10}, \lambda_{T_1,-5}, \lambda_{T_2,-5}, \lambda_{T_2,0}, \lambda_{T_3,0}, \lambda_{T_3,5} \le 0.$$

For $x_1 = 0$, $x_2 = 1$, we force

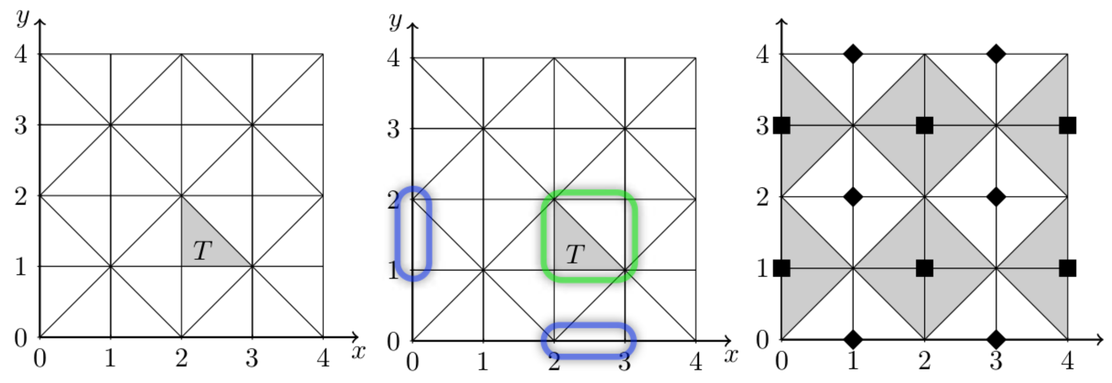
$$\lambda_{T_2,-5}, \lambda_{T_2,0} \le 1,$$
$$\lambda_{T_1,-10}, \lambda_{T_1,-5}, \lambda_{T_3,0}, \lambda_{T_3,5}, \lambda_{T_4,5}, \lambda_{T_4,10} \le 0.$$

For $x_1 = 1$, $x_2 = 1$, we force

$$\lambda_{T_3,0}, \lambda_{T_3,5} \le 1,$$
$$\lambda_{T_1,-10}, \lambda_{T_1,-5}, \lambda_{T_2,-5}, \lambda_{T_2,0}, \lambda_{T_4,5}, \lambda_{T_4,10} \le 0.$$

It is considerably more difficult to find a logarithmic reformulation for CC, since there is no explicit connection between vertices and simplices as in DCC. For the one-dimensional case we know from Example 2 that the λ's are SOS2 constrained. So we have to find a way to extend this result to higher dimensions. For a general triangulation, this might not be possible, but for J_1 triangulations it is. However we will restrict ourself to the two-dimensional case and refer to [Vielma et al., 2010] for generalization.

Let us assume that the J_1 triangulation is defined on $J := [0, \omega]^2$. We want to select one simplex T as shown in Figure 3a. This simplex belongs to a square (green), who's projections (blue) on the coordinate axes satisfy a SOS2 condition. This is shown in Figure 3b. We can also see, that this is only true, because the triangulation is J_1. Via Theorem 2.5 we can reformulate those constraints with a logarithmic number of binaries. Now all we have to do is choose one of two simplices in the square. This can be archived with a pattern as shown in Figure 3c. Each square has one light and one dark simplex and those simplices with a diamond vertice are light and those with a square vertice are dark. The set of all diamond vertices is $R = \{(r, s) \in \mathscr{V}(\mathscr{T}) : r \text{ is odd, } s \text{ is even}\}$ and the set of all square vertices is $L = \{(r, s) \in \mathscr{V}(\mathscr{T}) : r \text{ is even, } s \text{ is odd}\}$.



(A) One simplex T of a J_1 triangulation. [Vielma, 2008]
(B) T belongs to the green square who's projections are marked in blue. [Vielma, 2008]
(C) Diamond and square vertices induce a pattern on the J_1 triangulation. [Vielma, 2008]

If we put all those constraints together with the remaining constraints from CC, we get the following logarithmic reformulation (LogCC):

$$\sum_{(r,s)\in J} \lambda_{(r,s)}\left(r,s\right) = x, \qquad \sum_{(r,s)\in J} \lambda_{(r,s)} f\left(r,s\right) = z, \tag{3.15}$$

$$\lambda_{(r,s)} \geq 0 \qquad \forall\left(r,s\right)\in J, \qquad \sum_{(r,s)\in J} \lambda_{(r,s)} = 1, \tag{3.16}$$

$$\sum_{s=0}^{\omega}\sum_{r\in J^{+}(l,B)} \lambda_{(r,s)} \leq x_{(1,l)}, \sum_{s=0}^{\omega}\sum_{r\in J^{0}(l,B)} \lambda_{(r,s)} \leq 1 - x_{(1,l)}, \tag{3.17}$$

$$x_{(1,l)} \in \{0,1\} \ \forall L\left(\omega\right), \tag{3.18}$$

$$\sum_{r=0}^{\omega}\sum_{s\in J^{+}(l,B)} \lambda_{(r,s)} \leq x_{(2,l)}, \sum_{r=0}^{\omega}\sum_{s\in J^{0}(l,B)} \lambda_{(r,s)} \leq 1 - x_{(2,l)}, \tag{3.19}$$

$$x_{(2,l)} \in \{0,1\} \ \forall L\left(\omega\right), \tag{3.20}$$

$$\sum_{(r,s)\in L} \lambda_{(r,s)} \leq x_0, \ \sum_{(r,s)\in R} \lambda_{(r,s)} \leq 1 - x_0, \ x_0 \in \{0,1\}. \tag{3.21}$$

Where L, B, J^0, J^+ are as defined in Proposition 2.3 and B is a SOS2 compatible function. From the two-dimensional case the one-dimensional case easily follows by simply leaving out equation (3.20) - (3.21) and simplifying some notations.

Again we will consider the previous

Example (LogCC)
To show that SOS2 compatible functions are not unique, we use G defined via $G(1) = (0,0)^T$, $G(2) = (1,0)^T$, $G(3) = (1,1)^T$, $G(4) = (0,1)^T$. The function G is SOS2 compatible and $G \neq B$ where B denotes the SOS2 compatible function from the DLog example. For G LogCC gives the following constraints:

$$\begin{aligned}
-10\lambda_{-10} - 5\lambda_{-5} + 0\lambda_0 + 5\lambda_5 + 10\lambda_{10} &= x,\\
-100\lambda_{-10} - 25\lambda_{-5} + 0\lambda_0 + 25\lambda_5 + 100\lambda_{10} &= z,\\
\lambda_{-10}, \lambda_{-5}, \lambda_0, \lambda_5, \lambda_{10} &\geq 0,\\
\lambda_{-10} + \lambda_{-5} + \lambda_0 + \lambda_5 + \lambda_{10} &= 1,\\
\lambda_0 \leq x_1, \quad \lambda_{-10} + \lambda_{10} \leq (1-x_1), \quad \lambda_5 + \lambda_{10} \leq x_2, \quad \lambda_{-10} + \lambda_{-5} &\leq (1-x_2),\\
x_1, x_2 &\in \{0,1\}.
\end{aligned}$$

For $x_1 = 0$, $x_2 = 0$, we force

$$\begin{aligned}\lambda_{-10}, \lambda_{-5} &\leq 1,\\ \lambda_0, \lambda_5, \lambda_{10} &\leq 0.\end{aligned}$$

For $x_1 = 1$, $x_2 = 0$, we force

$$\begin{aligned}\lambda_{-5}, \lambda_0 &\leq 1,\\ \lambda_{-10}, \lambda_5, \lambda_{10} &\leq 0.\end{aligned}$$

For $x_1 = 0$, $x_2 = 1$, we force

$$\begin{aligned}\lambda_5, \lambda_{10} &\leq 1,\\ \lambda_{-10}, \lambda_{-5}, \lambda_0 &\leq 0.\end{aligned}$$

For $x_1 = 1$, $x_2 = 1$, we force

$$\begin{aligned}\lambda_0, \lambda_5 &\leq 1,\\ \lambda_{-10}, \lambda_{-5}, \lambda_{10} &\leq 0.\end{aligned}$$

We can see that the constraints on λ are truly SOS2 and therefore only one simplex is active for the convex combination.

3.2.4 Incremental Method (INC)

Another approach to represent ϕ is the Incremental Method (INC). We will only introduce it for the sake of completeness, because INC is used in Section 4. For a more detailed overview see [Vielma et al., 2010].
The formulation of INC requires the triangulation $\mathscr{T}$ to satisfy the following ordering properties:

Definition 3.3 (Ordering Properties for INC)

(i) The simplices in $\mathscr{T}$ can be ordered as $T_1, \ldots, T_{\mathscr{T}}$ s.t. $T_i \cap T_{i-1} \neq \emptyset$ for $i \in \{2, \ldots, |\mathscr{T}|\}$.

(ii) For the order in (i), the vertices of each simplex T_i can be ordered as $v_i^0, \ldots, v_i^{|V(T_i)|-1}$ s.t. $v_{i-1}^{|V(T_i)|-1} = v_i^0$ for $i \in \{2, \ldots, |T|\}$.

For a J1 triangulation such an ordering always exists [Wilson, 1998].
We now introduce auxiliary continuous variables δ_i^j for each vertice v_i^j and auxiliary binary variables y_i for each simplex T_i in Definition 3.3. Then for a given ordered $\mathscr{T}$, the piecewise linearization can be formulated as follows:

$$v_1^0 + \sum_{i=1}^{|\mathscr{T}|} \sum_{j=1}^{|V(T_i)|-1} \delta_i^j \left(v_i^j - v_i^0 \right) = x, \tag{3.22}$$

$$\phi\left(v_1^0\right) + \sum_{i=1}^{|\mathscr{T}|} \sum_{j=1}^{|V(T_i)|-1} \delta_i^j \left(f\left(v_i^j\right) - f\left(v_i^0\right) \right) = z, \tag{3.23}$$

$$\sum_{j=1}^{|V(T_i)|-1} \delta_i^j \leq 1,\ \delta_i^j \geq 0 \qquad \forall i \in \{1, \ldots, |\mathscr{T}|\}, j \in \{1, \ldots, |V(T_i)| - 1\}, \tag{3.24}$$

$$y_i \leq \delta_i^{|V(T_i)|-1}, \qquad \sum_{j=1}^{|V(T_{i+1})|-1} \delta_{i+1}^j \leq y_i, \qquad y_i \in \{0,1\}\ \forall i \in \{1, \ldots, |\mathscr{T}| - 1\}. \tag{3.25}$$

A more detailed derivation can be found in [Lee and Leyffer, 2012].

3.2.5 Model size

For the models introduced in Section 3 the number of additional constraints and auxiliary continuous and binary variables is listed in Table 1.

Model	Constraints	Continuous variables	Binary variables
DCC	$n + \lvert\mathscr{T}\rvert + 2$	$\sum_{T\in\mathscr{T}} \lvert V(T)\rvert$	$\lvert\mathscr{T}\rvert$
CC	$n + 3 + \lvert\mathscr{V}(\mathscr{T})\rvert$	$\lvert\mathscr{V}(\mathscr{T})\rvert$	$\lvert\mathscr{T}\rvert$
DLog	$n + 2\lceil\log_2(\lvert\mathscr{T}\rvert)\rceil + 2$	$\sum_{T\in\mathscr{T}} \lvert V(T)\rvert$	$2\lceil\log_2(\lvert\mathscr{T}\rvert)\rceil$
INC	$1 + 2\lvert\mathscr{T}\rvert$	$\sum_{T\in\mathscr{T}} (\lvert V(T)\rvert - 1)$	$\lvert\mathscr{T}\rvert - 1$

Table 1: Size of the different piecewise linearization models. LogCC is left out, since only the one-dimensional and two-dimension case were discussed.

Besides model size also algebraic properties of the different approaches can be examined. In this work we want to focus more on numerics and will not go into detail on those algebraic properties, a good overview on that can e.g. be found in [Vielma et al., 2010].

3.3 Linearization Error

This Section is dedicated to an a priori error estimation for special nonlinear functions that will appear in the numerical examples in Section 4. We want to show that for those functions an equidistant meshgrid leads to a minimal approximation error - in the two dimensional case that is true under the assumption that the triangulation is J_1. We follow the dissertation of Andrea Zelmer [Zelmer, 2010]. The following definitions are necessary in order to discuss what sort of error we should take into account.

Definition 3.4

(i) The absolute maximum linearization error on a simplex T is defined by

$$\Delta_T^{abs} := \max_{x \in T} |f(x) - \phi_T(x)|. \tag{3.26}$$

The absolute maximum linearization error on a triangulation $\mathscr{T}$ is defined by

$$\Delta_{\mathscr{T}}^{abs} := \max_{T \in \mathscr{T}} \Delta_T^{abs}. \tag{3.27}$$

(ii) The relative maximum linearization error on a simplex T is defined by

$$\Delta_T^{rel} := \max_{x \in T, f(x) \neq 0} \frac{|f(x) - \phi_T(x)|}{|f(x)|}. \tag{3.28}$$

The relative maximum linearization error on a triangulation $\mathscr{T}$ is defined by

$$\Delta_{\mathscr{T}}^{rel} := \max_{T \in \mathscr{T}} \Delta_T^{rel}. \tag{3.29}$$

(iii) The maximum scaled linearization error on a triangulation $\mathscr{T}$ is defined by

$$\Delta_{\mathscr{T}}^{sca} := \frac{\Delta_{\mathscr{T}}^{abs}}{\max_{x \in D} |f(x)|}. \tag{3.30}$$

The general idea is to calculate the number of intervals $n \in \mathbb{N}$ per dimension necessary for each linearization performed on a nonlinear model in order to result in an error smaller than a given tolerance $\epsilon > 0$. Therefore the absolute maximum linearization error $\Delta_{\mathscr{T}}^{abs}$ is of no use, since it is not normalized. The relative maximum linearization error is normalized but tends to be really high whenever $f(x)$ is small. That is why instead we will use the scaled linearization error $\Delta_{\mathscr{T}}^{sca}$.

3.3.1 Univariate Quadratic Functions

First we will consider a univariate quadratic function $f(x) = \alpha x^2$ on $[x_{min}, x_{max}]$ with $\alpha \in \mathbb{R}\backslash\{0\}$.

Question: How many intervals are necessary for a given $\epsilon > 0$ s.t. $\Delta^{sca}_{\mathscr{T}} \leq \epsilon$?

Let $\mathscr{T}$ be a triangulation of $[x_{min}, x_{max}]$ with $|\mathscr{T}| = n \in \mathbb{N}$ and ϕ be the piecewise linear approximation of f on $[x_{min}, x_{max}]$. Then $\mathscr{V}(\mathscr{T}) = \{x_0, \ldots, x_n\}$ holds and can be ordered as: $x_0 = x_{min}$, $x_n = x_{max}$ and $x_{i-1} < x_i$ for all $i \in \{1, \ldots, n\}$.
Chose an arbitrary interval $[x_{i-1}, x_i]$. Then $f(x_{i-1}) = \phi(x_{i-1})$ and $f(x_i) = \phi(x_i)$ implies that $\max\limits_{x\in[x_{i-1},x_i]} |\phi(x) - f(x)| = \max\limits_{x\in(x_{i-1},x_i)} |\phi(x) - f(x)|$. Also $\alpha \in \mathbb{R} \setminus \{0\}$ implies that f is either strictly convex ($\alpha > 0$), in which case ϕ overestimates f or strictly concave ($\alpha < 0$), in which case ϕ underestimates f. Without loss of generality we assume $\alpha > 0$. This implies $|\phi(x) - f(x)| = \phi(x) - f(x)$ for all $x \in [x_{min}, x_{max}]$. Each point $x \in [x_{i-1}, x_i]$ can be parameterized as $x = x(t) = x_{i-1} + t(x_i - x_{i-1})$ for some $t \in [0, 1]$. Hence $f(x) = f(x(t))$, $\phi(x) = \phi(x(t))$ on $[x_{i-1}, x_i]$. For any critical point the derivative of the distance must be 0:

$$\begin{aligned} 0 &\overset{!}{=} \frac{d}{dt}\left[\phi(x(t)) - f(x(t))\right] \\ &= \frac{d}{dt}\alpha[x_{i-1}^2 + t(x_i^2 - x_{i-1}^2) - (x_{i-1} + t(x_i - x_{i-1}))^2] \\ &= \alpha(x_i - x_{i-1})^2(1 - 2t) \end{aligned} \tag{3.31}$$

Then $t = \frac{1}{2}$ follows for the critical point and $\frac{d^2}{dt^2}[\phi(x(t)) - f(x(t))] = -2\alpha(x_i - x_{i-1})^2 < 0$ for all $t \in [0, 1]$ implies that $x\left(\frac{1}{2}\right)$ actually is a maximum. Further calculation gives

$$\begin{aligned} \Delta^{abs}_{[x_{i-1},x_i]} &= \phi\left(x\left(\frac{1}{2}\right)\right) - f\left(x\left(\frac{1}{2}\right)\right) \\ &= \alpha\left[x_{i-1}^2 + \frac{1}{2}(x_i^2 - x_{i-1}^2) - x_{i-1} + \frac{1}{2}(x_i - x_{i-1}))^2\right] \\ &= \frac{\alpha}{4}(x_i - x_{i-1})^2. \end{aligned} \tag{3.32}$$

We can see that in equation (3.32) the absolute maximum linearization error only depends on the distance $x_i - x_{i-1}$. Therefore intervals must be chosen equidistant in order to minimize the absolute maximum linearization error of the triangulation. For n equidistant intervals this implies

$$\Delta^{abs}_{\mathscr{T}} = \frac{\alpha}{4n^2}(x_{max} - x_{min})^2 \tag{3.33}$$

and by definition of the absolute maximum scaled linearization error $\Delta^{sca}_{\mathscr{T}} \leq \epsilon$ then gives

$$n = \left\lceil \sqrt{\frac{\alpha}{\epsilon \max\limits_{x\in[x_{min},x_{max}]} |f(x)|}} \frac{x_{max} - x_{min}}{2} \right\rceil. \tag{3.34}$$

In summary equidistant points are always optimal in order to minimize $\Delta^{sca}_{\mathscr{T}}$ and the number of intervals can be calculated via equation (3.34). In our numerical example we want $0 \in \mathscr{V}(\mathscr{T})$ to ensure that for $x \in [x_{min}, x_{max}]$ we have $f(x) = 0$ if and only if $\phi(x) = 0$ holds.

Bivariate Functions

Next we want to estimate the linearization error of the bivariate nonlinear function $f(x,y) = \alpha xy$ on $D := [x_{min}, x_{max}] \times [y_{min}, y_{max}]$ with $x_{min} < x_{max}$, $y_{min} < y_{max}$ and $\alpha \in \mathbb{R} \setminus \{0\}$.

Question: How many intervals per dimension are necessary for a given $\epsilon > 0$ s.t. $\Delta_{\mathscr{T}}^{sca} \leq \epsilon$ holds under the assumption that the triangulation is J_1?

Let $\mathscr{T}$ be a J_1 triangulation of D with $n \in 2\mathbb{N}$ intervals per dimension and ϕ be the piecewise linear approximation of f on D. Then $\mathscr{V}(\mathscr{T}) = \{x_0, \dots, x_n\} \times \{y_0, \dots, y_n\}$ where the latter sets are ordered. $R = [x_{i-1}, x_i] \times [y_{j-1}, y_j]$ with $i, j \in \{1, \dots, n\}$ is an arbitrary rectangle in D and is either up- or down-divided in the J_1 triangulation. This is illustrated in Figure 4.

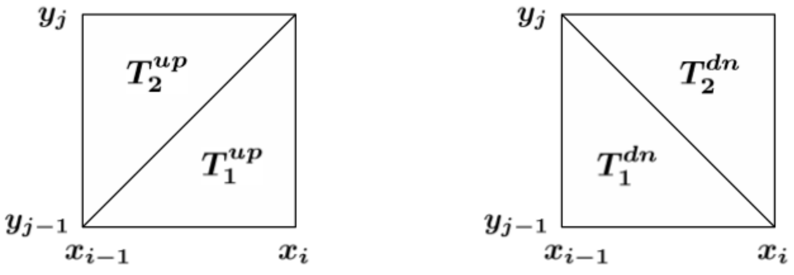


FIGURE 4: R is the union of two triangles and either of the left or right type [Zelmer, 2010].

This means that either $R = T_1^{up} \cup T_2^{up}$ or $R = T_1^{dn} \cup T_2^{dn}$. For T_k^l with $k \in \{1, 2\}$ and $l \in \{up, dn\}$ we get the following linear approximations:

$$\begin{aligned}
\phi_1^{up}(x,y) &= \alpha(y_{j-1}x + x_i y - x_i y_{j-1}); \quad (x,y) \in T_1^{up} \\
\phi_2^{up}(x,y) &= \alpha(y_j x + x_{i-1} y - x_{i-1} y_j); \quad (x,y) \in T_2^{up} \\
\phi_1^{dn}(x,y) &= \alpha(y_{j-1}x + x_{i-1} y - x_{i-1} y_{j-1}); \quad (x,y) \in T_1^{dn} \\
\phi_2^{dn}(x,y) &= \alpha(y_j x + x_i y - x_i y_j); \quad (x,y) \in T_2^{dn}
\end{aligned} \tag{3.35}$$

First we want to determine the critical points of $\phi_k^l(x,y) - f(x,y)$. Solving

$$0 \overset{!}{=} \nabla_{(x,y)} \left[\phi_k^l(x,y) - f(x,y) \right] \tag{3.36}$$

gives

$$\begin{aligned}
(x_1^{up}, y_1^{up}) &= (x_i, y_{j-1}) \\
(x_2^{up}, y_2^{up}) &= (x_{i-1}, y_j) \\
(x_1^{dn}, y_1^{dn}) &= (x_{i-1}, y_{j-1}) \\
(x_2^{dn}, y_2^{dn}) &= (x_i, y_j)
\end{aligned} \tag{3.37}$$

All those points must be minimal points with function value 0, since they are all vertices of the corresponding triangles. ($\phi(v) = f(v)$ for all $v \in \mathscr{V}(\mathscr{T})$ by definition.) Therefore the maximum must be attained on the facets of the triangles. Each facet of the two triangles on R can be represented as connection line of two points $(x_1, y_1), (x_2, y_2) \in \mathscr{V}(R)$ with $(x_1, y_1) \neq (x_2, y_2)$. An arbitrary point (x,y) on a line connecting (x_1, y_1) and (x_2, y_2) is given by the parametrization $(x,y) = (x,y)(t) = (x_1, y_1) + t((x_2, y_2) - (x_1, y_1))$ with $t \in [0, 1]$.

Then analogously to the univariate case

$$\begin{aligned} f((x,y)(t)) &= \alpha(x_1 + t(x_1 - x_1))(y_1 + t(y_2 - y_1)) \\ \phi((x,y)(t)) &= f(x_1,y_1) + t(f(x_2,y_2) - f(x_1,y_1)) \end{aligned} \tag{3.38}$$

follows for the corresponding parametrization of f and ϕ on the facets. Since $\phi((x,y)(0)) - f((x,y)(0)) = \phi((x,y)(1)) - f((x,y)(1)) = 0$ the maximum must be attained inside $[0,1]$. Again we determine the critical points of $\phi(x,y) - f(x,y)$:

$$\begin{aligned} 0 &\overset{!}{=} \frac{d}{dt}\left[\phi((x,y)(t)) - f((x,y)(t))\right]] \\ &= \alpha[x_2y_2 - x_2y_1 - x_1y_2 + x_1y_1 + 2t(-x_2y_2 + x_2y_1 + x_1y_2 - x_1y_1)] \\ &= \alpha(x_2 - x_1)(y_2 - y_1)(1 - 2t). \end{aligned} \tag{3.39}$$

By assumption $\alpha \neq 0$ implies that if $x_1 = x_2$ or $y_1 = y_2$ equation (3.39) holds for all $t \in [0,1]$. We already know that the absolute maximum linearization error disappears on the vertices of R and since f, ϕ are continuous, it must disappear on ∂R. For $x_1 \neq x_2$ and $y_1 \neq y_2$ the only extreme point is $t = \frac{1}{2}$. Since the error disappears for $t = 0$ and $t = 1$ if $\phi((x,y)(\frac{1}{2})) - f((x,y)(\frac{1}{2})) \neq 0$ then $(x,y)(\frac{1}{2})$ must be a maximum. Reformulation gives $(x,y)(\frac{1}{2}) = \frac{1}{2}(x_1 + x_2, y_1 + y_2)$, which is the intersection of up- and down-division. Therefore we use up-division and insert the solution into $\phi - f$:

$$\Delta^{abs}_{T^l_k} = \frac{\alpha(x_i - x_{i-1})(y_j - y_{j-1})}{4}. \tag{3.40}$$

In equation (3.40) we can see that again the maximum linearization error only depends on the distances $x_i - x_{i-1}$ and $y_j - y_{j-1}$. Thus equidistant intervals are optimal. This gives

$$\Delta^{abs}_{\mathscr{T}} = \frac{\alpha(x_{max} - x_{min})(y_{max} - y_{min})}{4n^2} \tag{3.41}$$

and since $\frac{\Delta^{abs}_{\mathscr{T}}}{\max\limits_{(x,y)\in D} |f(x,y)|} = \Delta^{sca}_{\mathscr{T}} \leq \epsilon$ this gives

$$n = \left\lceil \left(\frac{\alpha(x_{max} - x_{min})(y_{max} - y_{min})}{4\epsilon \max\limits_{(x,y)\in D} |f(x,y)|} \right)^{\frac{1}{2}} \right\rceil. \tag{3.42}$$

For the optimization we want $(0,0) \in \mathscr{V}(\mathscr{T})$ and $(0,y)$ and $(x,0)$ to be the union of facets of $T \in \mathscr{T}$. Note that for a J1 triangulation the first already implicates the latter.

4 Application to a Hybrid Energy Supply Network Design Problem

As an example to test the different linearization methods from Section 3.2 we consider a hybrid energy supply network problem with two types of nonlinear constraints. Suppose a utility company wants to operate a hybrid energy supply network based on the energy carriers electricity and gas. Modern micro-energy technologies such as combined heat and power units (CHPs) are used to transform one energy form into another. The other technologies are heat pumps (HPs) and condensing boilers (CBs). The company wants to minimize the overall operating and installing costs. To model the problem, a graph (V, E) with households at the nodes and energy flows on the edges is introduced. The optimization model has the following variables and parameters:

Sets	V	Nodes
	V_0	Source node
	V_1	Sink nodes
	E	Arcs
	E_0	Entering arc
	E_1	Arcs without the entering arc
Network variables	u_i	Electric voltage at node i [V]
	v_i	Electric current through node i [A]
	$\bar{v}_{ij}$	Electric current at arc (i,j) [A]
	s_i^e	Electric instantaneous power at node i [kWh]
	$\bar{u}_{ij}$	Electrical voltage difference on arc (i,j) [V]
	Ψ_i	Auxiliary variable as a placeholder for $s_i^e x_{ele,i}$
	p_i	Gas pressure at node i [mbar]
	q_i	Gas flow through node i [m^3/h]
	$\bar{q}_{ij}$	Gas flow at arc (i,j) [m^3/h]
	s_i^g	Gas instantaneous power at node i [kWh]
	$\bar{p}$	Gas pressure difference on arc (i,j) [mbar]
	Φ_{ij}	Auxiliary variable as a placeholder for $\operatorname{sign}(\bar{q}_{ij})\,\bar{q}_{ij}^2$
	C_{energy}	Overall energy cost of gas and electricity [€]
	C_{carbon}	Overall carbon emission cost by gas and electricity [€]
	C_{net}	Overall cost of network construction [€]
Hub system variables	$s_{chp,i}^g$	Gas power flowing into CHP at node i [kWh]
	$s_{cb,i}^g$	Gas power flowing into CB at node i [kWh]
	$s_{hp,i}^e$	Electrical power flowing into HP at node i [kWh]
	$s_{chp,i}^h$	Heat generated by CHP at node i [kWh]
	$s_{cb,i}^h$	Heat generated by CB at node i [kWh]
	$s_{hp,i}^h$	Heat generated by HP at node i [kWh]
	$s_{chp,i}^e$	Electricity generated by CHP at node i [kWh]
	C_{system}^m	Overall maintenance cost of technologies [€]
	C_{system}^{inv}	Overall investment cost of technologies [€]
Binary variables	y_{ij}^e	Binary decision for electrical cable construction on arc (i,j)
	y_{ij}^g	Binary decision for gas pipeline construction on arc (i,j)
	$x_{chp,i}$	Binary decision for installing CHP at node i
	$x_{cb,i}$	Binary decision for installing CB at node i
	$x_{hp,i}$	Binary decision for installing HP at node i
	$x_{ele,i}$	Binary decision for electricity supply/consumption at node i

Network parameters	d_i^h	Heat demand at node i [kWh]
	d_i^e	Electricity demand at node i [kWh]
	R_{ij}^e	Electric resistance on arc (i,j) [Ω]
	R_{ij}^g	Gas resistance on arc (i,j) $\left[\frac{\text{mbar}}{(\text{m}^3/\text{h})^2}\right]$
	β^g	Cost of gas [€/kWh]
	β_b^e	Cost of buying electricity from power grid [€/kWh]
	β_s^e	Cost of selling electricity to power grid [€/kWh]
	ι^g	Cost of carbon emission by gas [€/kWh]
	ι^e	Cost of carbon emission by electricity [€/kWh]
	γ_{ij}^g	Construction cost of gas pipeline on arc (i,j) [€]
	γ_{ij}^e	Construction cost of electrical cable on arc (i,j) [€]
Hub system parameters	Λ_{chp}	Heating power capacity of CHPs [kW]
	Λ_{cb}	Heating power capacity of CBs [kW]
	Λ_{hp}	Heating power capacity of HPs [kW]
	ξ_{chp}^h	Coefficient of performance of CHP for heat
	ξ_{cb}^h	Coefficient of performance of CB for heat
	ξ_{hp}^h	Coefficient of performance of HP for heat
	ξ_{chp}^e	Coefficient of performance of CHP for electricity
	β_{chp}^{inv}	Investment cost of CHP [€]
	β_{cb}^{inv}	Investment cost of CB [€]
	β_{hp}^{inv}	Investment cost of HP [€]
	β_{chp}^{m}	Annual maintenance cost of CHP [€]
	β_{cb}^{m}	Annual maintenance cost of CB [€]
	β_{hp}^{m}	Annal maintenance cost of HP [€]
	ρ_{max}	Maximal penetration degree of CHPs defined by electricity-generation

The cost function and constraints are as follows:

$$\begin{aligned} \underset{z\in\mathscr{Z}}{\text{minimize}} \quad & f(z) = C_{energy} + C_{carbon} + C_{system}^{m} + \alpha(T)\left(C_{system}^{inv} + C_{cable}\right) \\ \text{subject to} \quad & \text{Equations (4.1a) - (4.7e)} \end{aligned}$$

Whereas $\alpha(T)$ is the financial discounting factor for a planning time horizon of $T = 20$ years and C are the different cost terms.

The physical constraints for the electricity network especially include the conservation equation and the relations between voltage and current:

$$\sum_{j|(j,i)\in E} \bar{v}_{ji} - \sum_{j|(i,j)\in E} \bar{v}_{ij} - v_i = 0, \quad \forall i \in V_1, \tag{4.1a}$$

$$\mathbf{s_i^e} = \mathbf{a^e u_i v_i}, \quad \forall \mathbf{i} \in \mathbf{V_1}, \tag{4.1b}$$

$$\bar{u}_{ij} = u_i - u_j, \quad \forall (i,j) \in E, \tag{4.1c}$$

$$\bar{\mathbf{v}}_{\mathbf{ij}} = \left(\mathbf{R_{ij}^e}\right)^{-\mathbf{1}} \mathbf{y_{ij}^e} \bar{\mathbf{u}}_{\mathbf{ij}}, \quad \forall (\mathbf{i},\mathbf{j}) \in \mathbf{E}, \tag{4.1d}$$

$$u_i - u_{max} = 0, \quad \forall i \in V_0, \tag{4.1e}$$

$$u_{min} \le u_i \le u_{max}, \quad \forall i \in V_1, \tag{4.1f}$$

$$-\bar{v}_{max} \le \bar{v}_{ij} \le \bar{v}_{max}, \quad (i,j) \in E, \tag{4.1g}$$

$$y_{ij}^e \in \{0,1\}, \quad \forall (i,j) \in E. \tag{4.1h}$$

The physical constraints for the gas network especially contain the conservation equation and the quadratic proportionality between gas pressure $\bar{p}_{ij}$ and volume flow $\bar{q}_{ij}$ on edge (i,j), or shorter $\bar{p}_{ij} \propto \bar{q}_{ij}^2$:

$$
\begin{aligned}
&\sum_{j|(j,i)\in E} \bar{q}_{ji} - \sum_{j|(i,j)\in E} \bar{q}_{ij} - q_i = 0, \quad \forall i \in V_1, && (4.2a)\\
&s_i^g = a^g q_i, \quad \forall i \in V_1, && (4.2b)\\
&\bar{p}_{ij} = p_i - p_j, \quad (i,j) \in E, && (4.2c)\\
&\boldsymbol{\Phi}_{\mathbf{ij}} = \left(\mathbf{R}^{\mathbf{g}}_{\mathbf{ij}}\right)^{\mathbf{-1}} \mathbf{y}^{\mathbf{g}}_{\mathbf{ij}} \bar{\mathbf{p}}_{\mathbf{ij}}, \quad \forall (\mathbf{i},\mathbf{j}) \in \mathbf{E}, && (4.2d)\\
&\boldsymbol{\Phi}_{\mathbf{ij}} = \operatorname{sign}\left(\bar{\mathbf{q}}_{\mathbf{ij}}\right) \bar{\mathbf{q}}^{\mathbf{2}}_{\mathbf{ij}}, \quad \forall (\mathbf{i},\mathbf{j}) \in \mathbf{E}, && (4.2e)\\
&p_i - p_{max} = 0, \quad \forall i \in V_0, && (4.2f)\\
&p_{min} \le p_i \le p_{max}, \quad \forall i \in V_1, && (4.2g)\\
&-\bar{q}_{max} \le \bar{q}_{ij} \le \bar{q}_{max}, \quad \forall (i,j) \in E, && (4.2h)\\
&q_i \ge 0, \quad \forall i \in V_1, && (4.2i)\\
&y_{ij}^g \in \{0,1\}, \quad \forall (i,j) \in E. && (4.2j)
\end{aligned}
$$

The remaining constraints ensure that e.g. demands of the households must be satisfied, the mixture of energy carriers due to CHPs is included, only installed technologies can be used and the cost terms are defined properly:

$$
\sum_{i\in V_1} s^e_{chp,i} \le \rho \sum_{i\in V_1} \left(d_i^e + s^e_{hp,i}\right). \tag{4.3}
$$

And for all $i \in V_1$

$$
\begin{aligned}
&d_i^e = s_i^e + s^e_{chp,i} - s^e_{hp,i}, && (4.4a)\\
&d_i^h = s^h_{chp,i} + s^h_{cb,i} + s^h_{hp,i}, && (4.4b)\\
&s_i^g = s^g_{chp,i} + s^g_{cb,i}, && (4.4c)\\
&s^e_{chp,i} - \xi^e_{chp} s^g_{chp,i} = 0, && (4.4d)\\
&s^h_{chp,i} - \xi^h_{chp} s^g_{chp,i} = 0, && (4.4e)\\
&s^h_{cb,i} - \xi^h_{cb} s^g_{cb,i} = 0, && (4.4f)\\
&s^h_{hp,i} - \xi^h_{hp} s^e_{hp,i} = 0, && (4.4g)\\
&\mathbf{s}^{\mathbf{g}}_{\mathbf{chp,i}}\left(\mathbf{1} - \mathbf{x}_{\mathbf{chp,i}}\right) = \mathbf{0}, && (4.4h)\\
&\mathbf{s}^{\mathbf{g}}_{\mathbf{cb,i}}\left(\mathbf{1} - \mathbf{x}_{\mathbf{cb,i}}\right) = \mathbf{0}, && (4.4i)\\
&\mathbf{s}^{\mathbf{e}}_{\mathbf{hp,i}}\left(\mathbf{1} - \mathbf{x}_{\mathbf{hp,i}}\right) = \mathbf{0}, && (4.4j)\\
&s^h_{chp,i} - \Lambda^h_{chp} \le 0, && (4.4k)\\
&s^h_{cb,i} - \Lambda^h_{cb} \le 0, && (4.4l)\\
&s^h_{hp,i} - \Lambda^h_{hp} \le 0, && (4.4m)\\
&s^e_{\{chp,hp\},i} \ge 0,\ s^g_{\{chp,cb\},i} \ge 0,\ s^h_{\{chp,cb,hp\},i} \ge 0, && (4.4n)\\
&x_{\{chp,cb,hp\},i} \in \{0,1\}. && (4.4o)
\end{aligned}
$$

$$
\begin{aligned}
&\mathbf{s}^{\mathbf{e}}_{\mathbf{i}}\left(\mathbf{1} - \mathbf{2x}_{\mathbf{ele,i}}\right) \le \mathbf{0}, && (4.5)\\
&x_{ele,i} \in \{0,1\}. && (4.6)
\end{aligned}
$$

$$\mathbf{C_{energy}} = \sum_{\mathbf{i \in V_1}} \boldsymbol{\beta^{g} s_i^{g} + \beta_b^{e} s_i^{e} x_{ele,i} + \beta_s^{e} s_i^{e} \left(1 - x_{ele,i}\right)}, \tag{4.7a}$$

$$\mathbf{C_{carbon}} = \sum_{\mathbf{i \in V_1}} \boldsymbol{\iota^{g} s_i^{g} + \iota^{e} s_i^{e} x_{ele,i}}, \tag{4.7b}$$

$$C_{system}^{inv} = \sum_{i \in V_1} \beta_{chp}^{inv} x_{chp,i} + \beta_{cb}^{inv} x_{cb,i} + \beta_{hp}^{inv} x_{hp,i}, \tag{4.7c}$$

$$C_{system}^{m} = \sum_{i \in V_1} \beta_{chp}^{m} x_{chp,i} + \beta_{cb}^{m} x_{cb,i} + \beta_{hp}^{m} x_{hp,i}, \tag{4.7d}$$

$$C_{net} = \sum_{(i,j) \in E} \gamma_{ij}^{e} y_{ij}^{e} + \gamma_{ij}^{g} y_{ij}^{g}. \tag{4.7e}$$

A very detailed introduction and derivation of the model can be found in [Lu et al., 2017] and is left out here, because we only want to use the hybrid energy supply network problem as a numerical example and will not do any interpretation of our solutions with respect to real life applications.
The equations written in bold are nonlinear, but all except equation (4.1b) and equation (4.2e) can be linearized via reformulation. For more on that see [Lu et al., 2017]. The equations (4.1b), (4.2e) cannot be linearized via reformulation and thus piecewise linearization methods are applied.

4.1 Numerical Error Analysis

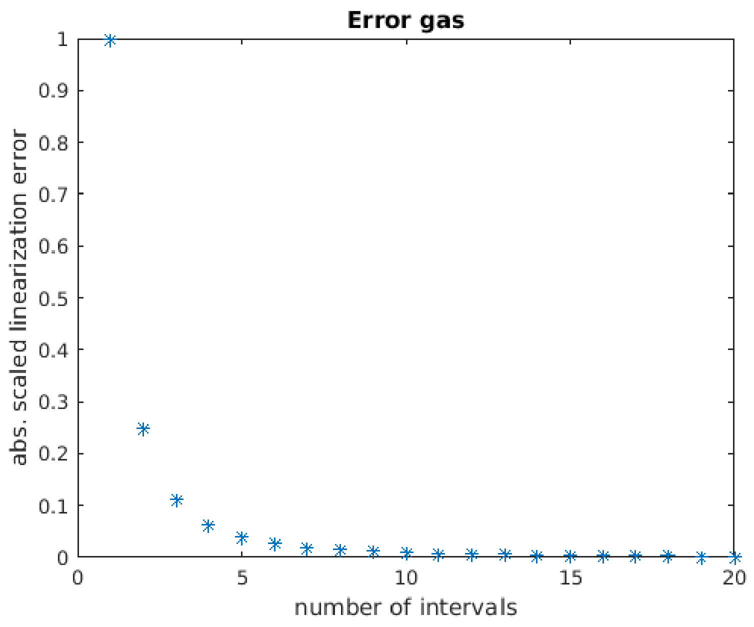
Now we want to apply the general error analysis from Section 3.3 onto the nonlinearities

$$\begin{aligned} \Phi(\bar{q}) &= |\bar{q}|\bar{q} \text{ on } [\bar{q}_{min}, \bar{q}_{max}] \text{ with } \bar{q}_{min} < \bar{q}_{max} \\ s^e(u,v) &= a^e uv \text{ with } u \in [u_{min}, u_{max}],\ v \in [v_{min}, v_{max}] \text{ and } u_{min} < u_{max},\ v_{min} < v_{max} \end{aligned} \tag{4.8}$$

of the hybrid energy supply network problem. The bounds on $\bar{q}$, u, v are chosen independently of the graph (V, E). Therefore no indices $i \in V$ or $(i,j) \in E$ are necessary in the function definitions. Figure 5 shows the dependency of the maximum scaled linearization error on the number of intervals for the parameters listed in Table 2.

$\bar{q}_{min}$	$\bar{q}_{max}$	u_{min}	u_{max}	v_{min}	v_{max}	a^e
-200	200	350	450	-50	50	0.00173

Table 2: Bounds and parameters for Φ and s^e.



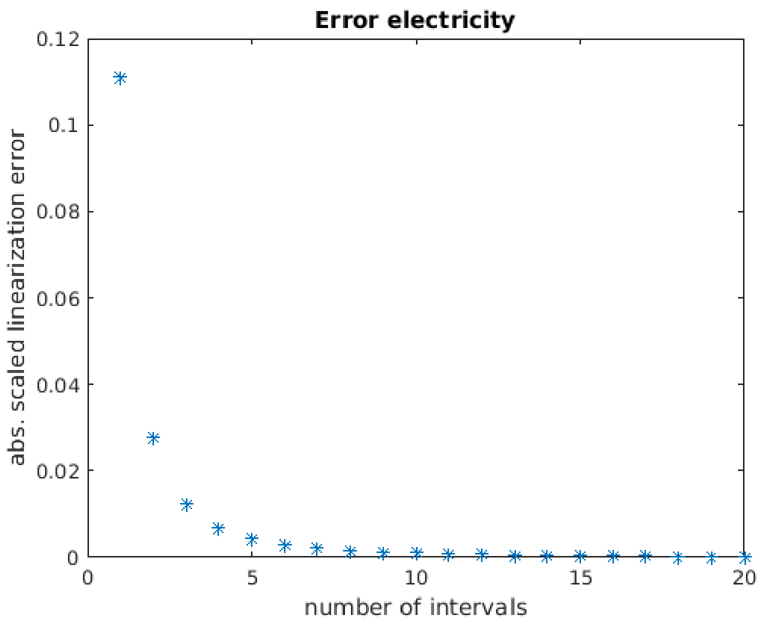


Figure 5: Absolute maximum scaled linearization error for gas network Φ and electricity network s^e.

One can see that the influence of the number of intervals n on $\Delta^{sca}_{\mathscr{T}}$ is stronger for smaller n and becomes very weak for $n \geq 10$. This behavior can already be seen in equations (3.33) and (3.41), because absolute and scaled error only differ in a proportionality factor. For the gas network the number of intervals is equivalent to the number of simplices, while for the electricity network the number of simplices is given by $2n^2$. Since the number of simplices also has a big influence on computational time, n should not be chosen too big, especially for the electricity network. Also the error on the gas network is bigger than the one on the electricity network. That is why we will choose $n_{gas} = 8$, $n_{el} = 4$ and $n_{gas} = 16$, $n_{el} = 8$ for computation.

Another very important issue is the generation of bounds in Table 2. In our application not all bounds are given or arise naturally. For v_{min} and v_{max} no bound exists. This means that we have to estimate bounds as tight as possible, but not too tight, because that could lead to infeasibility. This process is crucial for the performance and solution quality and must be done individually for every optimization problem.

4.2 Implementation

The data generation for the hybrid energy networks and triangulations for the linearizations were done with MATLAB 2016a. For the implementation of the hybrid energy supply network problem we used ZIMPL as modeling language. ZIMPL stands for Zuse Institut Mathematical Programming Language and was developed at the Zuse Institut in Berlin by Thorsten Koch in 2004 [Koch, 2004]. The big advantage of ZIMPL is that the implementation of the equations describing the optimization model in Section 4 is straightforward - almost a one-to-one translation. Also ZIMPL is beginner friendly and only few commands are necessary. In general ZIMPL always works well on graphs and is open source.

As MILP solver we used the MATLAB toolbox of CPLEX 12.6.3.0, which is developed by IBM and free for students. The basic principle is a Branch and Cut algorithm, which uses amongst others simplex and interior point methods as LP solver. Then a main file in MATLAB is used to call ZIMPL and CPLEX. For the original (nonlinear) model we use SCIP as MINLP solver. All computations were performed on an Intel Core i5-3317U 1.7 GHz, 4GB RAM Ubuntu 17.04 Notebook. The gaps in Tables 5, 8 and 11 are the actual primal-dual gaps given by CPLEX. The time limit was set to 3600 s for all computations. We also set $\rho_{max} = 10000$ which means that there is no restriction on the number of CHPs used in the network.

4.3 Computational Results

4.3.1 12-Node Network

The first test instance is a 12-node network given by the graph

$$\begin{aligned} V &= \{0, 1, \dots, 11\} \\ E &= \{(0,1),(1,2),(3,4),(1,5),(4,5),(1,6),(6,7), \\ &\quad (7,8),(8,9),(2,10),(9,10),(2,11)\}. \end{aligned} \tag{4.9}$$

This graph is visualized in Figure 6.

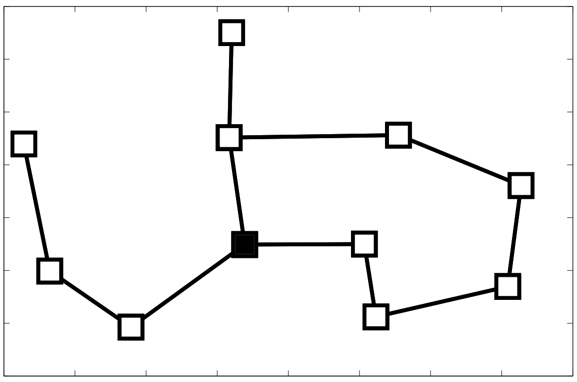


Figure 6: Visualization of the graph defined in equation (4.9). Note that V_1 is marked black and V_0 is missing.

The arc lengths and electricity and heat loads for (V, E) are given in Table 3

Length												
(i,j)	(0,1)	(1,2)	(3,4)	(1,5)	(4,5)	(1,6)	(6,7)	(7,8)	(8,9)	(2,10)	(9,10)	(2,11)
l_{ij}	999	408	487	447	312	336	277	388	384	476	394	397
Heat												
i	0	1	2	3	4	5	6	7	8	9	10	11
d_i^h	0	7.9949	9.8391	7.3635	8.3423	6.8975	9.0058	7.0225	8.0253	9.7154	7.9584	9.5082
Electricity												
i	0	1	2	3	4	5	6	7	8	9	10	11
d_i^h	0	0.8906	0.9590	0.5471	0.1386	0.1492	0.2574	0.8405	0.2542	0.2317	0.3611	0.6203

Table 3: Parameters for the 12-node network.

The computational results are displayed in Table 5. As already mentioned the J_1 triangulations were generated with $n_{gas} = 8$, $n_{el} = 4$ or $n_{gas} = 16$, $n_{el} = 8$ equidistant points per dimension. As we have proven in Section 3.3 it is in our case optimal to choose grid points equidistant. The model sizes for the two settings are listed in Table 4.

	Original	DCC		CC		DLog		LogCC		INC	
		4/8	8/16	4/8	8/16	4/8	8/16	4/8	8/16	4/8	8/16
# Binary Variables	68	516	1668	516	1668	159	193	159	193	493	1645
# Continuous Variables	286	1510	4870	645	1357	1510	4870	645	1357	1073	3281
# Constraints	457	938	2090	896	1608	672	740	672	740	1352	3656

Table 4: Number of variables and constraints of the 12-node network for different linearization methods.

We can see that the piecewise linearizations need more than twice the number of binary variables even for the logarithmic formulations and about seven times the number of binary variables for the other formulations. Also we can see now that in our case LogCC and DLog need the same number of binary variables, even though we did not discuss the model size for LogCC in Section 3.2.5.

	Original	DCC		CC		DLog		LogCC		INC	
		4/8	8/16	4/8	8/16	4/8	8/16	4/8	8/16	4/8	8/16
Gas ($\bar{q}_{01}$)	*	3.82530	3.82530	3.82530	3.82530	3.82530	3.82530	3.82530	3.82530	3.82530	3.82530
Elec. ($\bar{v}_{01}$)	*	17.25540	16.76242	17.25538	16.76242	17.25543	16.76242	17.25543	16.76242	17.25543	16.76240
Obj. value	*	21.02671	21.02671	21.02671	21.02671	21.02671	21.02671	21.02671	21.02671	21.02671	21.02671
Gap (%)	*	0	0	0	0	0	16.27	0	0	0	0
CPU (s)	*	4.5722	26.4300	4.1618	3.8129	32.3530	*	3.3910	10.2112	2.5898	10.7861

Table 5: Solutions of the 12-node network for different linearization methods. * indicates that the time limit was reached or that no solution could be found within the given time.

The most important observation from the results in Table 5 is, that whilst for the original model not even a primal solution can be found within 3600 s, all linearizations - except DLog in one case - can be solved to optimality in under 35 s. Also the objective function value of all linearizations is the same. This shows that the chosen method does not influence the objective value and that a finer triangulation does not lead to different (better) solutions.
The variables $\bar{q}_{01}$ and $\bar{v}_{01}$ stand for the total gas flow and electric current injected into the network. For the two triangulations those values differ, but are equal for all linearization methods. Already we can see that DLog performs worst and DCC might also be inferior to the other models. This will become clearer for bigger networks.

In order to interpret the results with respect to the application, we would also have to e.g. take into account how many CHPs are installed in the solution. We do not want to focus on that here and refer to [Lu et al., 2017] for interpretations with respect to application.

4.3.2 20-Node Network

The second network is a 20-node network given by

$$\begin{aligned} &V = \{0, 1, \ldots, 19\} \\ &E = \{(0,1), (2,3), (3,4), (1,5), (4,5), (1,6), (6,7), (7,8), (8,9), (6,10), \\ &\quad (9,10), (2,11), (10,12), (11,12), (12,13), (13,14), (11,15), (14,15), (3,16), \\ &\quad (15,17), (16,17), (17,18), (18,19)\}. \end{aligned} \tag{4.10}$$

This graph is visualized in Figure 7.

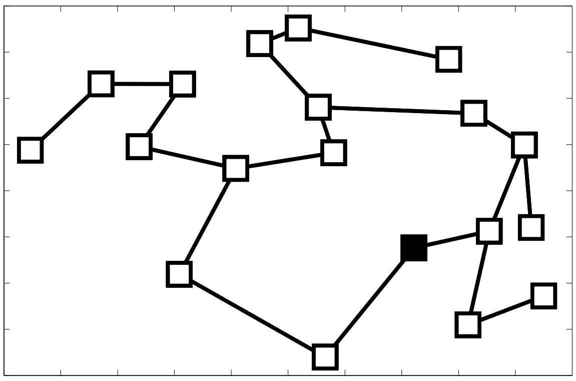


Figure 7: Visualization of the graph defined in equation (4.10). Note that V_1 is marked black and V_0 is missing.

The arc lengths and electricity and heat loads for (V, E) are given in Table 6.

						Length						
(i,j)	(0,1)	(2,3)	(3,4)	(1,5)	(4,5)	(1,6)	(6,7)	(7,8)	(8,9)	(6,10)	(9,10)	(2,11)
l_{ij}	999	351	499	567	626	275	412	294	300	393	358	204
(i,j)	(10,12)	(11,12)	(12,13)	(13,14)	(11,15)	(14,15)	(3,16)	(15,17)	(16,17)	(17,18)	(18,19)	
l_{ij}	224	548	250	546	344	152	353	323	311	287	380	
						Heat						
i	0	1	2	3	4	5	6	7	8	9	10	11
d_i^h	0	9.8993	6.3039	8.3481	7.6555	7.2365	3.0553	9.0351	9.9809	6.7463	9.1246	6.7832
i	12	13	14	15	16	17	18	19				
d_i^h	9.9694	9.2090	7.6969	8.9155	7.9934	9.2360	7.4260	6.2930				
						Electricity						
i	0	1	2	3	4	5	6	7	8	9	10	11
d_i^h	0	0.3405	0.5853	0.2239	0.7513	0.2552	0.5060	0.6991	0.8909	0.9593	0.5473	0.1387
i	12	13	14	15	16	17	18	19				
d_i^h	0.1494	0.2576	0.8407	0.2544	0.8143	0.2436	0.9293	0.3500				

Table 6: Parameters for the 20-node network.

The model sizes are listed in Table 7 and the results are displayed in Table 8. Again $n_{gas} = 8$, $n_{el} = 4$ or $n_{gas} = 16$, $n_{el} = 8$ were used to generate the grid.

	Original	DCC		CC		DLog		LogCC		INC	
		4/8	8/16	4/8	8/16	4/8	8/16	4/8	8/16	4/8	8/16
# Binary Variables	116	884	2868	884	2868	271	329	271	329	845	2829
# Continuous Variables	486	2590	8382	1101	2325	2590	8382	1101	2325	1841	5649
# Constraints	777	1602	3586	1528	2752	1144	1260	1144	1260	2312	6280

Table 7: Number of variables and constraints of the 20-node network for different linearization methods.

As in the last example, we can see that even for the coarser triangulation the logarithmic formulations need more than twice the number of binaries and the other formulations more than seven times the number of binaries as the original model.

	Original	DCC		CC		DLog		LogCC		INC	
		4/8	8/16	4/8	8/16	4/8	8/16	4/8	8/16	4/8	8/16
Gas ($\bar{q}_{01}$)	*	7.31348	7.31348	7.31351	7.31348	7.31348	7.22117	7.31348	6.25132	7.31348	7.31348
Elec. ($\bar{v}_{01}$)	*	28.92300	28.09663	28.92284	28.09663	28.92300	27.32332	28.92300	30.44025	28.92300	28.09663
Obj. value	*	34.43473	34.43473	34.43473	34.43473	34.43473	34.45395	34.43473	33.76965	34.43473	34.43473
Gap (%)	*	0	0	0	0	10.09	15.25	0	12.50	0	0
CPU (s)	*	17.5208	20.0100	8.1475	18.0186	*	*	5.7385	*	8.9057	94.9870

Table 8: Solutions of the 20-node network for different linearization methods. * indicates that the time limit was reached or that no solution could be found within the given time.

Again no primal solution can be found based on the original problem formulation. All linearization methods find primal solutions within the given time and for the coarser triangulation this is done in under 20 s except for DLog. The assumption that DLog is inferior can now be validated, since CPLEX does not converge within the given time. Also it is interesting that LogCC performs better than DCC, CC and INC for the coarser triangulation, but worst for the finer triangulation, where CC performs best. Another important observation is, that all models that converge within the given time do so with the same objective value. This again confirms the solution quality.

4.3.3 32-Node Network

The third network is a 32-node network given by

$$\begin{aligned} &V = \{0,1,\dots,31\} \\ &E = \{(0,1),(1,2),(2,3),(1,5),(4,5),(1,6),(6,7),(7,8),(6,10),(9,10), \\ &\quad (2,11),(10,12),(11,12),(13,14),(11,15),(14,15),(3,16),(15,17),(17,18),(18,19), \\ &\quad (16,20),(19,20),(4,21),(20,22),(21,22),(22,23),(21,25),(24,25),(5,26),(25,27), \\ &\quad (26,27),(27,28),(28,29),(13,30),(9,31),(30,31)\}. \end{aligned} \tag{4.11}$$

This graph is visualized in Figure 8.

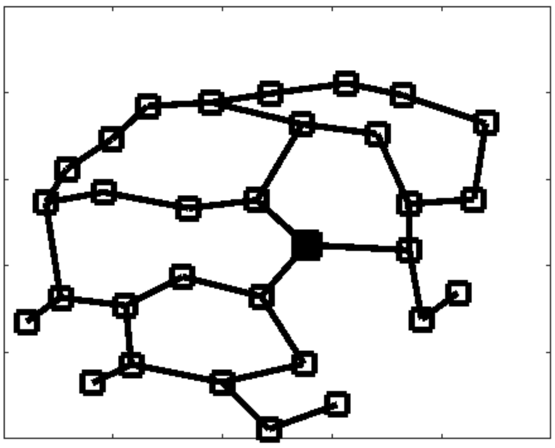


Figure 8: Visualization of the graph defined in equation (4.11). Note that V_1 is marked black and V_0 is missing.

The arc lengths and electricity and heat loads for (V, E) are given in Table 9.

						Length						
(i,j)	(0,1)	(1,2)	(2,3)	(1,5)	(4,5)	(1,6)	(6,7)	(7,8)	(6,10)	(9,10)	(2,11)	(10,12)
l_{ij}	999	344	314	362	377	471	404	225	267	299	485	428
(i,j)	(11,12)	(13,14)	(11,15)	(14,15)	(3,16)	(15,17)	(17,18)	(18,19)	(16,20)	(19,20)	(4,21)	(20,22)
l_{ij}	347	351	433	275	401	294	251	269	269	216	307	554
(i,j)	(21,22)	(22,23)	(21,25)	(24,25)	(5,26)	(25,27)	(26,27)	(27,28)	(28,29)	(13,30)	(9,31)	(30,31)
l_{ij}	301	209	341	208	436	426	389	346	342	268	441	411
						Heat						
i	0	1	2	3	4	5	6	7	8	9	10	11
d_i^h	0	8.0238	8.7963	9.5636	9.8372	8.1889	6.5545	6.5972	7.0300	9.3629	7.0171	9.2571
i	12	13	14	15	16	17	18	19	20	21	22	23
d_i^h	6.9741	9.7171	7.3999	6.7864	7.0043	8.4642	7.8932	7.4066	9.3233	8.3411	8.1989	9.6688
i	24	25	26	27	28	29	30	31				
d_i^h	7.1434	9.0288	9.0149	7.5218	8.2713	6.3034	6.2158	8.1232				
						Electricity						
i	0	1	2	3	4	5	6	7	8	9	10	11
d_i^h	0	0.7792	0.9340	0.1300	0.5689	0.4694	0.0120	0.3372	0.1623	0.7943	0.3113	0.5286
i	12	13	14	15	16	17	18	19	20	21	22	23
d_i^h	0.1657	0.6020	0.2630	0.6541	0.6892	0.7482	0.4506	0.0839	0.2291	0.9133	0.1525	0.8258
i	24	25	26	27	28	29	30	31				
d_i^h	0.5384	0.9961	0.0783	0.4427	0.1067	0.9619	0.0047	0.7749				

Table 9: Parameters for the 32-node network.

The model sizes are listed in Table 10 and the computational results are displayed in Table 11. Again $n_{gas} = 8$, $n_{el} = 4$ or $n_{gas} = 16$, $n_{el} = 8$ were used to generate the grid.

	Original	DCC		CC		DLog		LogCC		INC	
		4/8	8/16	4/8	8/16	4/8	8/16	4/8	8/16	4/8	8/16
# Binary Variables	196	1476	4740	1476	4740	459	557	459	557	1409	4673
# Continuous Variables	818	4298	13802	1845	3869	4298	13802	1845	3869	3049	9289
# Constraints	1317	2690	5954	2576	4600	1936	2132	1936	2132	3872	10400

Table 10: Number of variables and constraints of the 32-node network for different linearization methods.

As previously the ratios between the number of binaries in the original model and the linearizations stay the same. This is simply because the ratio equals the ratio between binaries in the original model and in the linearization per node.

	Original	DCC		CC		DLog		LogCC		INC	
		4/8	8/16	4/8	8/16	4/8	8/16	4/8	8/16	4/8	8/16
Gas ($\bar{q}_{01}$)	*	11.46776	11.46776	11.46778	11.49939	15.71106	*	11.46776	11.46046	11.46778	11.46776
Elec. ($\bar{v}_{01}$)	*	45.12865	43.83926	45.12857	43.46362	13.56927	*	45.12865	43.62201	45.12853	43.83926
Obj. value	*	53.90526	53.90526	53.90526	54.00902	55.95459	*	53.90526	53.95023	53.90526	53.90526
Gap (%)	*	13.39	17.90	0	5.00	20.88	*	0	19.27	0	7.99
CPU (s)	*	*	*	554.6928	*	*	*	1209.0425	*	759.8455	*

Table 11: Solutions of the 32-node network for different linearization methods. * indicates that the time limit was reached or that no solution could be found within the given time.

Again no primal solution for the original problem can be found within 3600 s. As previously seen DLog performs worse than the other models for larger networks, even more so for the 32-node network. Therefore it will be left out in the following analysis of the solutions. Also we will only focus on the coarser triangulation, because as already seen for the two other networks a higher resolution does not lead to a different objective value. Even though CPLEX does not converge for the finer triangulation we can see that the primal solutions are the same (or very close) to those of the coarser triangulation.
For the coarser triangulation DCC, CC, LogCC and INC all lead to the same objective value, whereas DCC is not able to close the gap within the given time. CC has the lowest computational time with only about 555 s, INC needs about 1.3 times that long and LogCC more than twice as long.

5 Conclusion

If we focus on the numerical results first it is remarkable how well piecewise linearization methods work for the hybrid energy supply network problem. With the exception of DLog they are always capable of generating fast and equivalent solutions for networks up to a size of 32-nodes. Even then the limit seems not to be reached for CC, LogCC and INC. Especially if we take into account that a J_1-triangulation generated with $n_{gas} = 8$ and $n_{el} = 4$ seems to be sufficient.

The bad performance of DLog is surprising and cannot be explained here. If solutions exist they correspond to those of the other formulations, thus the implementation seems to be correct. Also one would expect that LogCC might perform better than CC, especially with increasing model size. This is only the case for the first two given test instances, but it should be mentioned that we also did tests with $\rho_{max} = 1$ (higher combinatorics) and LogCC performed better than CC for the 32-node network in that case. Especially the model size alone is not enough to estimate the performance of the chosen linearization.

A general problem for piecewise linear approximations is, that the performance and solution quality strongly depends on the dimension of the nonlinear function. To be more precise: A sufficiently fine triangulation in large dimension means that a lot of simplices are necessary, in particular the number of binary variables increases.
One approach to tackle this problem is the decomposition of higher dimensional nonlinear functions f into univariate and bivariate functions. This is always possible and essentially means that only those univariate and bivariate functions need to be linearized. For a given error tolerance this can be archived with a number of simplices polynomial in the dimension of the domain of f [Leyffer et al., 2008].

Also our approach generates approximations which are no relaxations of the underlying MINLP, this can lead to infeasibility or the exclusion of optimal solutions. With the approach presented in [Geißler, 2011] our implementation can be modified to build piecewise linear relaxations easily.
One proverb that summarizes piecewise linearizations pretty good is “If you have a hammer, everything is a nail”. In theory everything can be linearized, but whether or not this approach is reasonable depends on the application.

References

[Bernreuther, 2017] Bernreuther, M. (2017). Linearisierungstechniken in der Gemischt Ganzzahligen Nichtlinearen Optimierung - Ein Gasnetzwerk als Anwendung. Seminar paper.

[Geißler, 2011] Geißler, B. (2011). *Towards Globally Optimal Solutions for MINLPs by Discretization Techniques with Applications in Gas Network Optimization.* PhD thesis, Uni Erlangen-Nürnberg.

[Koch, 2004] Koch, T. (2004). *Rapid Mathematical Prototyping.* PhD thesis, Technische Universität Berlin.

[Lee and Leyffer, 2012] Lee, J. and Leyffer, S., editors (2012). *Mixed Integer Nonlinear Programming.* Springer Science+Business Media.

[Leyffer et al., 2008] Leyffer, S., Sartenaer, A., and Wanufelle, E. (2008). Branch-and-Refine for Mixed-Integer Nonconvex Global Optimization.

[Lu et al., 2017] Lu, J., Bernreuther, M., and Volkwein, S. (2017). Mathematical modeling and global optimization of hybrid energy supply networks. Technical Report 358. Universität Konstanz.

[Todd, 1974] Todd, M. J. (1974). Union Jack Triangulations. Technical report, Department of Operations Research, College of Engineering, Cornell University.

[Vielma, 2008] Vielma, J. P. (2008). Mixed Integer Programming Models for Non-Separable Piecewise Linear Cost Functions.

[Vielma et al., 2010] Vielma, J. P., Ahmed, S., and Nemhauser, G. L. (2010). Mixed-Integer Models for Nonseparable Piecewise-Linear Optimization: Unifying Framework and Extensions. *Operations Research*, 58(2):303–315.

[Vielma and Nemhauser, 2011] Vielma, J. P. and Nemhauser, G. L. (2011). Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. *Math. Program.*, 128(1-2):49–72.

[Wilson, 1998] Wilson, D. L. (1998). *Polyhedral methods for piecewise-linear functions.* PhD thesis, University of Kentucky.

[Wolsey, 1998] Wolsey, L. (1998). *Integer Programming.* John Wiley & Sons, New York.

[Zelmer, 2010] Zelmer, D.-M. A. (2010). *Designing Coupled Energy Carrier Networks by Mixed-Integer Programming Methods.* PhD thesis, Uni Erlangen-Nürnberg.