Journal of Global Optimization (2021) 80:731–756
https://doi.org/10.1007/s10898-020-00977-x

Linearization of McCormick relaxations and hybridization with the auxiliary variable method

Jaromił Najman[1] · **Dominik Bongartz**[1] · **Alexander Mitsos**[1]

Received: 20 December 2019 / Accepted: 6 December 2020 / Published online: 9 February 2021
© The Author(s) 2021

Abstract
The computation of lower bounds via the solution of convex lower bounding problems depicts current state-of-the-art in deterministic global optimization. Typically, the nonlinear convex relaxations are further underestimated through linearizations of the convex underestimators at one or several points resulting in a lower bounding linear optimization problem. The selection of linearization points substantially affects the tightness of the lower bounding linear problem. Established methods for the computation of such linearization points, e.g., the sandwich algorithm, are already available for the auxiliary variable method used in state-of-the-art deterministic global optimization solvers. In contrast, no such methods have been proposed for the (multivariate) McCormick relaxations. The difficulty of determining a good set of linearization points for the McCormick technique lies in the fact that no auxiliary variables are introduced and thus, the linearization points have to be determined in the space of original optimization variables. We propose algorithms for the computation of linearization points for convex relaxations constructed via the (multivariate) McCormick theorems. We discuss alternative approaches based on an adaptation of Kelley's algorithm; computation of all vertices of an n-simplex; a combination of the two; and random selection. All algorithms provide substantial speed ups when compared to the single point strategy used in our previous works. Moreover, we provide first results on the hybridization of the auxiliary variable method with the McCormick technique benefiting from the presented linearization strategies resulting in additional computational advantages.

Keywords Global optimization · McCormick · Linearization · MAiNGO

Mathematics Subject Classification 49M20 · 49M37 · 65K05 · 90C26

✉ Alexander Mitsos
amitsos@alum.mit.edu

[1] Process Systems Engineering (AVT.SVT), RWTH Aachen University, Forckenbeckstraße 51, 52074 Aachen, Germany

1 Introduction

State-of-the-art algorithms for deterministic global optimization of nonconvex problems are mainly based on adaptations of the spatial branch-and-bound algorithm [17,25,35], where lower and upper bounds are computed in each node of the branch-and-bound tree. In most cases, established nonlinear local solvers supported by several heuristics are used to compute valid upper bounds. The computation of lower bounds is performed through the application of a general method for the construction of valid convex and concave relaxations of the original functions participating in the optimization problem. Due to numerical reliability and computational costs, established deterministic global solvers such as, e.g., ANTIGONE [27], BARON [22] or SCIP [38], construct a linear outer approximation of the nonlinear convex relaxation and use well-developed linear programming methods for the determination of valid lower bounds instead of applying local convex techniques directly. The selection of points for linearization affects the tightness of the linear relaxation and consequently the lower bound obtained through the solution of the resulting linear program.

The auxiliary variable method (AVM) [33,35,36] is a general method for the construction of convex and concave relaxations of factorable functions, i.e., functions representable as a finite recursion of addition, multiplication and composition of intrinsic functions. The AVM introduces an auxiliary variable together with a corresponding auxiliary equality constraint for every intermediate nonlinear factor of a given function. Then, the convex and concave envelopes of each factor are constructed providing a convex relaxation and a concave relaxation of the original function. To linearize the convex and concave envelopes of each factor provided by the AVM, the so-called sandwich algorithm has been developed (Section 4.2 in [35]). The algorithm starts with a small set of linearization points and computes additional points based on a particular rule, e.g., the maximum error rule or the chord rule, for every auxiliary factor until a pre-defined maximum of linearization points has been computed. This method works very well for the AVM, since each introduced auxiliary factor is defined by a low dimensional function for which the convex and concave envelopes are known analytically allowing for an efficient computation of maximum distances and/or angles used in the sandwich procedure. Moreover, all computations are conducted in the low dimensional domain space of the intermediate factor further lowering the computational effort. However, the resulting linear program suffers from an increased dimensionality and a large number of constraints because of the auxiliary variables and equality constraints added.

The method of McCormick [26], extended to multivariate compositions in [37], is a different method for the construction of valid convex and concave relaxations of factorable functions. In contrast to the AVM, the McCormick technique does not introduce any auxiliary variables when constructing convex and concave relaxations of a given function, thus always preserving the original dimension of the underlying function. Since the resulting McCormick relaxations may be nonsmooth, we use subgradient propagation [28] in order to construct valid affine under- and overestimators for the convex and concave McCormick relaxations. It is also possible to use the recently developed differentiable McCormick relaxations [23] in order to replace the computation of subgradients by the computation of derivatives, but they are not as tight as the nonsmooth McCormick relaxations and are computationally more costly to calculate. Furthermore, Cao et al. [6] have shown that affine subtangents used for the linearization of the convex relaxation of the lower bounding problem benefit from the favorable convergence order of McCormick relaxations. Thus, we stick to the nonsmooth McCormick relaxations and its subgradients throughout this article. As currently no algorithms exist for the selection of suitable linearization points for the McCormick relaxations,

the goal of this work lies in the development of algorithms for the computation of a set of linearization points. We briefly discuss the idea of direct propagation and combination of all subgradients in each intermediate factor. However, this approach results in the computation of a combinatorial number of affine functions and thus, we focus on methods for computing linearization points in the original variable space either iteratively or a priori. All methods developed for the linearization of multivariate McCormick relaxations are implemented on the basis of the open-source `MC++` library [7] and are available in the open-source deterministic global optimization solver MAiNGO [4].

Tawarmalani and Sahinidis [36] showed that for a specific set of linearization points, the linear outer approximation obtained via the AVM can be tighter than the linear outer approximation computed with McCormick relaxations. Later Tsoukalas and Mitsos demonstrated that the McCormick method extended to multivariate outer functions provides the same convex and concave relaxations if no reoccurring intermediate factors of a given function are recognized by the AVM. However, if the AVM recognizes reoccurring intermediate factors, it provides tighter convex relaxations than the multivariate McCormick approach. This problem, can, at least in princible, be solved by the addition of only a sufficient number of auxiliary optimization variables when applying the McCormick method, thus providing equally tight relaxations as the AVM. The authors thus suggest a combination of the AVM and the multivariate McCormick technique (Section 4 in [37]), but do not provide any computational insights or results. Modern computational representations of factorable functions through directed acyclic graphs (DAGs) allow for the recognition of factors occurring multiple times in a given optimization problem [7,32]. This allows for the development of a method that introduces common factors as auxiliary optimization variables together with the corresponding auxiliary constraints. Herein, we first briefly summarize the theory discussed in [37], showing that the AVM and the McCormick relaxations provide equally tight relaxations if common factors occurring at least twice are recognized. Then, we present a heuristic for the addition of intermediate factors occurring most often in a given DAG as auxiliary optimization variables in the lower bounding procedure and show that this hybridization of the AVM and the McCormick propagation technique results in large computational speed ups. We add only a limited number of auxiliary optimization variables in order get the advantage of tighter relaxations when common factors are recognized and keep the benefit of lower dimensional optimization problems when using the McCormick method. Last, we apply the linearization algorithms developed herein to the hybridization of the AVM and multivariate McCormick relaxations to come up for the increased dimensionality and resulting complexity of the lower bounding problems. Again, the implemented heuristic is available in the open-source solver MAiNGO [4].

The remainder of this article is structured as follows. Section 2 introduces the mathematical notation used throughout the manuscript. In Sect. 3 we present alternative algorithms for the computation of linearization points for McCormick relaxations a priori, iteratively and a combination of both supported by numerical results. Section 4 provides results on the application of the presented linearization algorithms to the hybridization of the McCormick technique and the auxiliary variable method. We summarize and discuss future work in Sect. 5.

2 Preliminaries

If not stated otherwise, we consider continuous functions $f : Z \to \mathbb{R}$ with $Z \in \mathbb{IR}^n$, where $\mathbb{IR}$ denotes the set of closed bounded intervals in $\mathbb{R}$. $Z \in \mathbb{IR}^n$, also called *box*, is defined as $Z \equiv [\mathbf{z}^L, \mathbf{z}^U] = [z_1^L, z_1^U] \times \cdots \times [z_n^L, z_n^U]$ with $\mathbf{z}^L, \mathbf{z}^U \in \mathbb{R}^n$ where the superscripts L and U always denote a lower and upper bound, respectively. We denote the range of f over Z by $f(Z) \in \mathbb{IR}$.

We call a convex function $f^{cv} : Z \to \mathbb{R}$ a convex relaxation (or convex underestimator) of f on Z if $f^{cv}(\mathbf{z}) \leq f(\mathbf{z})$ for every $\mathbf{z} \in Z$. Similarly, we call a concave function $f^{cc} : Z \to \mathbb{R}$ a concave relaxation (or concave overestimator) of f on Z if $f^{cc}(\mathbf{z}) \geq f(\mathbf{z})$ for every $\mathbf{z} \in Z$. We call the tightest convex and concave relaxations of f the convex and concave envelopes f_e^{cv}, f_e^{cc} of f on Z, respectively, i.e., it holds $f^{cv}(\mathbf{z}) \leq f_e^{cv}(\mathbf{z}) \leq f(\mathbf{z})$ and $f(\mathbf{z}) \leq f_e^{cc}(\mathbf{z}) \leq f^{cc}(\mathbf{z})$ for all $\mathbf{z} \in Z$ and all convex relaxations f^{cv} and concave relaxations f^{cc} of f on Z, respectively.

Definition 1 (Subgradients and linearization points). For a convex and concave function $f^{cv}, f^{cc} : Z \to \mathbb{R}$, we call $\mathbf{s}^{cv}(\bar{\mathbf{z}}), \mathbf{s}^{cc}(\bar{\mathbf{z}}) \in \mathbb{R}^n$ a convex and a concave subgradient of f^{cv}, f^{cc} at $\bar{\mathbf{z}} \in Z$, respectively, if

$$f^{cv}(\mathbf{z}) \geq f^{cv}(\bar{\mathbf{z}}) + (\mathbf{s}^{cv}(\bar{\mathbf{z}}))^T(\mathbf{z} - \bar{\mathbf{z}}), \qquad \forall \mathbf{z} \in Z, \tag{A1}$$

$$f^{cc}(\mathbf{z}) \leq f^{cc}(\bar{\mathbf{z}}) + (\mathbf{s}^{cc}(\bar{\mathbf{z}}))^T(\mathbf{z} - \bar{\mathbf{z}}), \qquad \forall \mathbf{z} \in Z. \tag{A2}$$

We denote the affine functions on the right-hand side of inequalities (A1), (A2) constructed with the convex and concave subgradient $\mathbf{s}^{cv}(\bar{\mathbf{z}}), \mathbf{s}^{cc}(\bar{\mathbf{z}})$ by $f^{cv,sub}(\bar{\mathbf{z}}, \mathbf{z})$ and $f^{cc,sub}(\bar{\mathbf{z}}, \mathbf{z})$, respectively.

Note that $f^{cv,sub}$ and $f^{cc,sub}$ are valid under- and overestimators of f on Z, respectively. We call the point $\bar{\mathbf{z}}$ in Definition 1 a *linearization point* of f.

2.1 McCormick relaxations and subgradient propagation

All relaxations considered herein are constructed with the use of the McCormick propagation rules originally developed by McCormick (Section 3 in [26]) and extended to multivariate compositions of functions by Tsoukalas and Mitsos (Theorem 2 in [37]). The corresponding subgradients are constructed as described by Mitsos et al. (Sections 2.4-3.1 in [28]) and Tsoukalas and Mitsos (Theorem 4 in [37]). We use the open-source library `MC++` [7] for the automatic computation of convex and concave McCormick relaxations and its subgradients at a single linearization point. Moreover, we extend the library to handle a set of linearization points directly by adapting the original implementation of McCormick propagation to vector valued calculations. This extension is available in the `MC++` library used in the open-source deterministic global solver MAiNGO [4].

3 Computation of linearization points for McCormick relaxations

Before describing the methods and algorithms used in this work, we briefly discuss the computation of linearization points for the AVM, its relationship with McCormick relaxations and the associated difficulties. We then begin the presentation of methods with an adaptation

of the iterative Kelley's cutting plane algorithm for convex optimization [21]. Subsequently, we develop a method for a priori computation of linearization points (before calculating relaxations). As a last point of this section, we present numerical results comparing the described methods and the combination of the two methods to a computationally cheap single-point approach and random selection.

3.1 Auxiliary variable method and McCormick relaxations

For the sake of demonstration, let us consider a simple abstract box-constrained minimization of $f : \mathbb{R}^n \supset X \to \mathbb{R}$

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{x} \in X \in \mathbb{R}^n. \end{aligned}$$

When applying the AVM to a given optimization problem, each intermediate factor is replaced by an auxiliary variable together with an appropriate equality constraint. For now, let us assume that the AVM introduces exactly one auxiliary variable z_1 for the intermediate factor g with range $g(X) = [g^L, g^U]$ providing the equivalent optimization problem

$$\begin{aligned} \min_{\mathbf{x}, z_1} \quad & \tilde{f}(z_1) \\ \text{s.t.} \quad & z_1 = g(\mathbf{x}) \\ & \mathbf{x} \in X \in \mathbb{R}^n \\ & z_1 \in [g^L, g^U], \end{aligned} \tag{AVM}$$

with $\tilde{f} : [g^L, g^U] \to \mathbb{R}$, $g : X \to [g^L, g^U]$ and $\tilde{f}(g(\mathbf{x})) = f(\mathbf{x})$ for all $\mathbf{x} \in X$. Then, convex and concave relaxations $\tilde{f}^{cv}$, g^{cv}, g^{cc} of $\tilde{f}$ and g over $[g^L, g^U]$ and X, respectively, are constructed. Note that in general the range set of g overestimates the true image of g under X, but to keep this example simple, we assume that the exact image of g under X is known. Finally, a set of linearization points $\mathscr{L}_{AVM}$ with $|\mathscr{L}_{AVM}| = N_{\mathscr{L}}$ is computed, e.g., with the sandwich algorithm and one of its different rules (Section 4.2 in [35]), and all convex and concave relaxations are replaced by their affine under- and overestimators computed at points in $\mathscr{L}_{AVM}$. For the above optimization problem (AVM), when assuming that every function is linearized the same number of times, we end up with $N_{\mathscr{L}}$ affine inequalities consisting of $\frac{N_{\mathscr{L}}}{3}$ for the convex relaxation of the objective, $\frac{N_{\mathscr{L}}}{3}$ for the convex relaxation, and $\frac{N_{\mathscr{L}}}{3}$ for the concave relaxation of the auxiliary function g. The linearized lower bounding problem is given as

$$\begin{aligned} \min_{\mathbf{x}, z_1, \eta} \quad & \eta \\ \text{s.t.} \quad & \eta \geq \tilde{f}^{cv}(\bar{z}_1) + (\mathbf{s}_{\tilde{f}}^{cv}(\bar{z}_1))^T (z_1 - \bar{z}_1), && \forall \left(\bar{\mathbf{x}}^{cv}, \bar{\mathbf{x}}^{cc}, \bar{z}_1\right) \in \mathscr{L}_{AVM} \\ & z_1 \geq g^{cv}(\bar{\mathbf{x}}^{cv}) + (\mathbf{s}_g^{cv}(\bar{\mathbf{x}}^{cv}))^T (\mathbf{x} - \bar{\mathbf{x}}^{cv}), && \forall \left(\bar{\mathbf{x}}^{cv}, \bar{\mathbf{x}}^{cc}, \bar{z}_1\right) \in \mathscr{L}_{AVM} \\ & z_1 \leq g^{cc}(\bar{\mathbf{x}}^{cc}) + (\mathbf{s}_g^{cc}(\bar{\mathbf{x}}^{cc}))^T (\mathbf{x} - \bar{\mathbf{x}}^{cc}), && \forall \left(\bar{\mathbf{x}}^{cv}, \bar{\mathbf{x}}^{cc}, \bar{z}_1\right) \in \mathscr{L}_{AVM} \\ & \mathbf{x} \in X \in \mathbb{R}^n \\ & z_1 \in [g^L, g^U]. \end{aligned} \tag{AVM$_{\text{LP}}$}$$

It is desirable to achieve the same tightness of the lower bounding linear optimization problem when using McCormick relaxations. Since no auxiliary variables are introduced when

McCormick's theorems are applied, this amounts to computing the projection of (AVM_{LP}) to the original variable space X. This is done by combining each of the affine under- or overestimators of function g with each of the affine relaxations of f. The choice whether the affine under- or overestimators of g have to be used, depends on the monotonicity of the affine relaxations of f (Corollary 3 in [37]). This combination results in a linear optimization problem with $(N_{\mathscr{L}})^2$ linear inequalities in the original variables $\mathbf{x}$ and only one (free) auxiliary variable η. Generalizing, if $F > 1$ auxiliary variables are introduced in the AVM, we end up with $\mathcal{O}\left((N_{\mathscr{L}})^{(F+1)}\right)$ affine inequalities for the McCormick method. The estimation of number of inequalities is not exact since some of the resulting affine inequalities may be redundant. Thus, propagating all affine inequalities with the McCormick method is not a promising idea as solely the computation of all affine functions would require an exponentional computational time.

For the AVM, we can say that the method pays for the lower number of affine inequalities with the increase of the dimension of the final linear problem (AVM_{LP}). This increase of number of variables however seems acceptable. The resulting LP is typically solved with (variants of the) simplex method [10]. In practice, such methods perform very well for large problems, and does not require anywhere near the worst-case exponential complexity. When propagating all combinations with the McCormick method, we have to compute all facets explicitly and thus, also all vertices of the underlying polytope. In contrast to the computation of vertices in the linear programming simplex algorithm, this propagation thus inevitably results in exponential computational runtime.

In order to achieve a similar tightness without the necessity of propagating all combinations of the affine functions and avoiding the combinatorial complexity, the following possibility can be considered. We choose only a limited number of what we think are *promising* linearizations in each factor and propagate only a small part of all possible combinations. The difficulty emerging with this approach is the decision making in a given factor f_j over domain X_{f_j} of the original function f, i.e., telling what is a *promising* linearization point, since only local information about the factor f_j on its domain X_{f_j} and not on the final function f over X_f is available. This local information is unsatisfactory because it is not possible to know which affine under- and overestimator of f_j is the best to be used for further propagation, since f may have many nonlinear factors following f_j. Still, several heuristics to overcome this issue can be applied to the concave and convex relaxations. First, we can choose the affine underestimator whose minimum value is the closest to the minimum value of the convex factor f_j can be considered. As an alternative, we can compute the value at the minimum point of the factor f_j over X_{f_j} among all propagated affine underestimators and save the tightest underestimator for further propagation. Finally, we can choose the affine underestimators within each intermediate factor randomly. Unfortunately, in preliminary experiments none of the above methods provides results which could compete with the algorithms and ideas described throughout the rest of this manuscript. Thus, we focus the scope of this paper on the iterative and direct computation of the set of linearization points $\mathscr{L}_{MC}$ providing the linear lower bounding problem in the space of original variables

$$\begin{aligned} \min_{\mathbf{x},\eta} \quad & \eta \\ \text{s.t.} \quad & \eta \geq f^{cv}(\bar{\mathbf{x}}) + (\mathbf{s}_f^{cv}(\bar{\mathbf{x}}))^T(\mathbf{x} - \bar{\mathbf{x}}), \quad \forall\, \bar{\mathbf{x}} \in \mathscr{L}_{MC} \\ & \mathbf{x} \in X \in \mathbb{R}^n. \end{aligned} \tag{MC_{LP}}$$

3.2 Adaptation of Kelley's algorithm

Cutting plane or localization methods are commonly used in continuous convex optimization [16] and are often also called bundle methods. The probably best known methods are the algorithm by Kelley [21] and the method by Cheney and Goldstein [8]. These algorithms may be inefficient in the sense that they may require the solution of multiple small linear programs until they stabilize but are computationally very cheap. To improve the efficiency, extensions of the methods by Kelley and Cheney and Goldstein have been developed during the last decade. The center of gravity method proposed by Newman [30] and Levin [24] computes the center of gravity of the given polyhedron approximating the convex set of interest. Although it is proven to decrease the volume of the polyhedron by a large part in each iteration, it is computationally not practicable. The maximum volume inscribed ellipsoid method [34] and the analytic center method [13] both require the solution of subsequent convex optimization problems in order to obtain a set of linearization points. As we have to compute possibly multiple linearization points in each node of the B&B algorithm, it is intuitive to avoid this additional computational overhead. The Chebyshev center cutting-plane method [11] requires the solution of a linear program to compute a linearization point (Chapter 8 in [5]) but it is prone to scaling and coordinate transformations. Since we have to compute multiple linearization points in each B&B node, the two algorithms by Kelley and Cheney and Goldstein seem acceptable in terms of computational time and resulting tightness of the lower bounding problem. Next, we present the method we use in this work and stick to the notion of *Kelley's algorithm*. We adapt the original algorithm to be able to directly use McCormick relaxations and the corresponding subgradients instead of computing tangents of (smooth) convex functions.

The idea of bundle methods is to approximate a given convex function by a set or a so-called bundle of affine underestimators. The bundle is constructed by iteratively generating and adding new affine functions. The pseudo-code of the procedure can be found in Algorithm 1. In particular, we start with an initial set of linearization points $\mathscr{L}$, e.g., the mid-point of the underlying interval domain (Line 4). We construct the lower bounding linear optimization problem L (MC_{LP}) with the use of McCormick relaxations by using subgradient propagation [28] with reference points defined in $\mathscr{L}$ (Line 5). We solve L to obtain the optimal solution point $\mathbf{x}^*$ and a corresponding solution value f^* (Line 9). We add the solution point $\mathbf{x}^*$ to the set of linearization points $\mathscr{L}$. We now extend the linear problem L with the inequalities which we get by the computation of subgradients of the McCormick relaxations at the new point $\mathbf{x}^*$ of every function participating in the original problem (Lines 16-17). We then re-solve L to get a new solution point $\mathbf{x}^*_{new}$ and its corresponding solution value f^*_{new} and repeat the procedure until either a maximum number N_K of linearizations has been reached (Line 8) or the difference of the solution values f^* and f^*_{new} reach a given threshold (Line 13). The implementation of the described method is available in the open-source solver MAiNGO [4].

3.3 Computation of linearization points as vertices of an n-simplex

We have shown how to construct lower bounding linear problems iteratively with a simple adaptation of well-known bundle methods in the previous section. In this section we discuss how to compute a *promising* set of linearization points a priori, i.e., before the first lower bounding linear optimization problem is solved.

First, we need to clarify what we denote as a *promising* set of linearization points. A *promising* set of linearization points consists of points, that are well-distributed among the

Algorithm 1: Adaptation of Kelley's algorithm for the iterative computation of linearizations for McCormick relaxations.

N_K - maximum number of additional linearizations;
μ^a - minimum absolute difference between solution values;
μ^r - minimum relative difference between solution values;
Given an initial set of linearization points $\mathscr{L}$;
Compute subgradients of McCormick relaxations at points in $\mathscr{L}$ and construct corresponding linear problem L (MC_{LP});
k←1;
$f_{old} = f_{new} = -\infty$;
while $k \leq N_K$ **do**
  Solve L;
  $\mathbf{x}^* \leftarrow$ optimal solution point of L;
  $f_{old} \leftarrow f_{new}$;
  $f_{new} \leftarrow$ optimal solution value of L;
  if $(f_{new} - f_{old}) < \min(\mu^r \cdot |f_{new}|, \mu^a)$ **then**
    return;
  end
  Compute subgradient $\mathbf{s}(\mathbf{x}^*)$ of McCormick relaxations;
  Extend L by $f^{cv,sub}(\mathbf{x}^*\mathbf{x})$ and $f^{cc,sub}(\mathbf{x}^*, \mathbf{x})$;
  k← k+1;
end

n-dimensional interval domain of the optimization variables and at the same time is not too large. Intuitively, since the nonlinear n-dimensional function $f^{cv} : X \in \mathbb{R}^n \rightarrow \mathbb{R}$, which we want to approximate through affine estimators, is convex, it should be enough to approximate the shape of f with the use of $n + 1$ affine underestimators. This intuition comes from the fact that a set $\mathscr{L}$ of $n+1$ points is enough to encapsulate any interior point $\mathbf{x}^* \in X$ within the convex hull of the points $\mathscr{L}$ in the domain space X. Moreover, since the function we aim to underestimate is convex, a set of points which is well-distributed among the domain appears to be more favorable when computing the lower bound rather than a possibly clustered set of linearization points. In practice, we cannot expect the linearization points based on this set to be exact at the minimum of f, but we hope that the affine linearizations defined by the set of linearization points, which is well-distributed among the domain of the optimization variables, can approximate f well enough. Additionally, the computation of only a limited number of affine underestimators does not increase the computational cost too much allowing for a possible trade-off between accuracy of the affine approximation and computational time.

In order to obtain a well-distributed set of linearization points, it is also possible to use well-known space covering algorithms such as Monte-Carlo methods [15] or quasi-random sequences such as the Sobol or the Halton sequence [9,19]. These methods cover the given space well for a large number of samples. However, we desire only a very limited number of linearization points when constructing lower bounding problems, making these methods unsuitable for our procedure. Additionally, Monte-Carlo methods and quasi-random sequences obviously posses the randomness factor, whereas the methods presented herein do not. Thus, we propose a different approach based on the computation of all vertices of an n-simplex. An n-simplex is defined as an n-dimensional polytope which also is the convex hull of its $n + 1$ vertices. The idea is to compute all $n + 1$ vertices of an n-simplex, with all vertices lying on an n-dimensional Ball centered at $\mathbf{0}$ in $[-1, 1]^n$ with radius $r \in (0, 1]$. These points can then be rescaled to the original interval domains of the optimization variables and then used for the computation of subgradients and McCormick relaxations. The $n + 1$ points

Algorithm 2: Algorithm for the computation of all $n+1$ vertices of an n-simplex lying on the n-dimensional ball with radius $r \in (0, 1)$.

n - dimension of the simplex, has to be ≥ 2;
r - radius $\in (0, 1)$;
$\mathscr{V}$ - set of all vertices $\mathbf{v}$ of the n-simplex;
Fill $\mathscr{V}$ with $n+1$ $\mathbf{0}$ vectors;
$v_{1,1} \leftarrow r$;
$v_{1,i} \leftarrow -\frac{r}{n}, i = 2, \ldots, n+1$;
for *i=1,...,n* **do**
  $t \leftarrow 0$;
  for *j=1,...,i* **do**
    $t \leftarrow t + v_{j,i+1}^2$;
  end
  $v_{i,i+1} \leftarrow \sqrt{r^2 - t}$;
  for *j=i+1,...,n* **do**
    $v_{i,j} \leftarrow -\frac{v_{i,i+1}}{n-i}$;
  end
end

are well-distributed in a sense that each vertex has the same distance to every other vertex in the $[-1, 1]^n$ box, namely $2 \cdot r\frac{\sqrt{3}}{2}$. Since subgradients of McCormick relaxations are steeper at the boundary of the domain if the convex relaxation has its minimum in the interior of the domain and thus, tend to contribute less to the tightness of the resulting linear program it is favorable to choose $r < 1$. It is also reasonable to choose the radius larger than 0 to avoid a possible clustering of the linearization set resulting in possibly many similar subgradients of the affine underestimators and finally redundant constraints. Thus, we recommend to choose $r \in (0.7, 0.9)$. The pseudo-code of the described procedure can be found in Algorithm 2. In the pseudo-code, the index (i, j) of a vertex $\mathbf{v}$ denotes the ith row of the jth vertex in the set $\mathscr{V}$. Throughout the algorithm we make use of the fact that all edges of the n-simplex have the same length, meaning that the sum $\sum_{i=1}^{n} v_{i,j}$ has to equal 0 for a fixed j. After the computation of the $n+1$ vertices of the n-simplex, it is advisable to rotate the polytope by a given angle γ with the help of a standard rotation matrix to avoid the computation of points with having multiple equal coordinates. It is not recommended to rotate the matrix with respect to each axis due to high computational effort in high dimensions, e.g., $n > 100$, but rather rotate with respect to every kth axis to avoid a large number of calculations. The case $n = 1$ is a special case, where we always use 3 points equally spread among the 1-dimensional interval $X = [x^L, x^U]$, being

$$x_1 = \frac{1}{3} \cdot (x^U - x^L) + x^L,\ x_2 = \frac{1}{2} \cdot (x^U - x^L) + x^L,\ \text{and}\ x_3 = \frac{2}{3} \cdot (x^U - x^L) + x^L.$$

We also experienced that using all $n+1$ points is not optimal if each function in the optimization problem only depends on a subset of the optimization variables. If the maximum number of variables participating non-linearly in any function of the optimization problem is $p < n$, we filter the points further and choose only p out of $n+1$ vertices. Note that when a p-simplex is computed, we obtain $p+1$ p-dimensional points. Just because there are at most p variables participating nonlinearly in some function f_i, it is not excluded that subsets of these variables and some additional variables participate nonlinearly in some other function f_j with $i \neq j$. It is possible to rescale these p vertices to the n variables, but then the same simplex vertices for multiple variables are used which could intuitively lead to possibly very

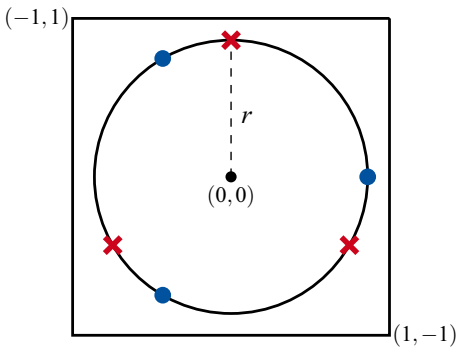

Fig. 1 Example of the n-simplex algorithm described in Sect. 3.3. 2-simplex vertices on the 2 dimensional ball with radius r. The original points computed by Algorithm 2 are depicted as blue circles. The rotated original points by an angle of 30° are depicted as red crosses

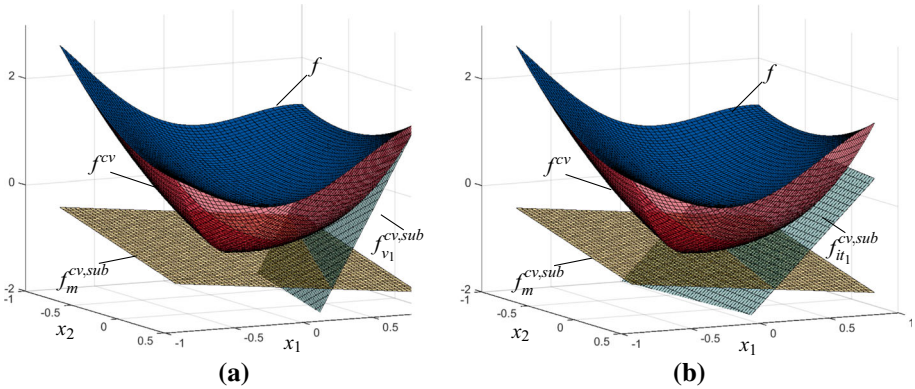

Fig. 2 Figures showing the affine underestimators of the three-hump camel function over $[-0.75, 1] \times [-1, 0.25]$ at linearization points obtained with the adaptation of Kelley's algorithm (Figure (a)) described in Sect. 3.2 and the n-simplex algorithm (Figure (b)) described in Sect. 3.3. **a** The linearizations are conctructed starting at the middle point $\mathbf{m}$ and conducting one iteration it_1 of the adapted Kelley's algorithm. **b** The linearizations are constructed at the middle point $\mathbf{m}$ and the point $\mathbf{v}_1$ obtained with the n-simplex algorithm

similar and often redundant affine underestimators. Moreover, since the computation of the n-simplex is performed only once in the global optimization procedure, the computation of a larger simplex is negligible. We choose only p points instead of $p+1$ and one of these points is always the mid point of the domain, since this choice has shown the best results in the numerical case studies performed in this work. Since the mid point is always used, we choose the $p-1$ points with the highest absolute improvement in interval tightness when applying the subgradient interval heuristic presented in our previous work [29] for computation of McCormick relaxations of the given optimization problem.

3.4 Illustrative example

The following example illustrates application of Algorithm 2 in practice. Consider the three-hump camel function

$$f : [-0.75, 1] \times [-1, 0.25] \to \mathbb{R}, (x_1, x_2) \mapsto 2 \cdot x_1^2 - 1.05 \cdot x_1^4 + \frac{x_1^6}{6} + x_1 \cdot x_2 + x_2^2.$$

First, we compute the vertices of the n-simplex lying on the 2 dimensional ball $\mathscr{B}$ with radius $r = 0.725$. The points of the 2-simplex on $\mathscr{B}$ are

$$\begin{aligned}\mathbf{v}_1 &= (0.725, 0),\\ \mathbf{v}_2 &= \left(-0.3625, \sqrt{r^2 - (-0.3625)^2}\right),\\ \mathbf{v}_3 &= \left(-0.3625, -\sqrt{r^2 - (-0.3625)^2}\right).\end{aligned}$$

Then, we rotate the points by 30° counterclockwise and finally obtain

$$\begin{aligned}\mathbf{v}_1 &= \left(\sqrt{r^2 - (-0.3625)^2}, -0.3625\right),\\ \mathbf{v}_2 &= (0, 0.725),\\ \mathbf{v}_3 &= \left(-\sqrt{r^2 - (-0.3625)^2}, -0.3625\right).\end{aligned}$$

Note that the similarity of the rotated points with respect to the original points is a special case of the 2 dimensional case. Figure 1 shows the (not) rotated points of the 2-simplex computed on the ball with radius $r = 0.725$.

Next, we scale the points to the domain $[-0.75, 1] \times [-1, 0.25]$ of f and get

$$\begin{aligned}\mathbf{v}_1 &\approx (0.674, -0.601),\\ \mathbf{v}_2 &= (0.125, 0.078125),\\ \mathbf{v}_3 &\approx (-0.424, -0.601).\end{aligned}$$

As final linearization points, we choose the mid point $\mathbf{m} = (0.125, -0.375)$ and the point $\mathbf{v}_i$ providing the most improvement when using the subgradient interval heuristic presented in our previous work [29], given by $\mathbf{v}_1$ in this example. Finally, we compute the linearization of convex McCormick relaxations of f at $\mathscr{L} = \{\mathbf{m}, \mathbf{v}_1\}$ (cf. Fig. 2a). Figure 2b shows the resulting affine underestimators of f when using the adaptation of Kelley's algorithm described in the previous subsection. The difference between the linearizations with respect to the resulting lower bound of the respective linear program is only marginal in this example although different linearization points are used.

3.5 Numerical results

We compare the presented linearization strategies by the solution of various problems with the deterministic global optimization solver MAiNGO [4] using CPLEX v12.8 [18] for the solution of linear lower bounding problems, IPOPT v3.12.0 [39] for local optimization in the pre-processing step, and the SLSQP solver found in the NLOPT package v2.5.0 [20] for local optimization within the B&B algorithm. For range reduction, we use optimization-based bound tightening with the trivial filtering algorithm with filtering value 0.001 [12], duality-based bound tightening using dual values obtained from CPLEX [31], and basic constraint propagation. All calculations are conducted on an Intel® Core™ i5-3570 CPU with 3.4 GHz running Fedora Linux 30.

We solve 9 problems of varying sizes with up to 57 variables chosen from the MINLPLIB2 library; 4 case studies of a combined-cycle power plant presented in Sections 5.2 and 5.2 of [1] where we either maximize the power output of the net or minimize the levelized cost of electricity; 2 case studies of the William-Otto process with 4 and 5 degrees of freedom, respectively [2]; and 3 case studies of chemical processes presented in [3], where we consider

the example of a flash modeled with the NRTL mixture model as well as the optimization of a methanol production process from H_2 and CO_2, once with original degrees of freedom (case c) and once additionally using the CO_2 inlet flowrate as degree of freedom making the case study somewhat more complex. All problem statistics can be found in Table 6 in Appendix A.

In the 9 problems found in the MINLPLIB2 library, we set the absolute and relative tolerances to $\epsilon = 10^{-4}$ and for the process engineering case studies, we set both optimality tolerances to $\epsilon = 10^{-2}$. We allow for a maximum of 8 hours. We consider the computational performance of the B&B algorithm for five cases:

– *Mid*, where every convex and/or concave relaxation is linearized at the middle point of the optimization variable domain. This is the setting used in all of our previous work.

– *Kelley*, where every convex and/or concave relaxation is linearized iteratively using the adaptation of Kelley's algorithm described in Sect. 3.2. We allow for at most p linearizations, where p is the maximum number of variables participating nonlinearly in any function of the optimization problem.

– *Simplex*, where every convex and/or concave relaxation is linearized at points determined through the application of the n-simplex algorithm described in Sect. 3.3. We choose p linearizations, where p is the maximum number of variables participating nonlinearly in any function of the optimization problem. We choose the p points with the highest absolute improvement in interval tightness when applying the subgradient interval heuristic presented in our previous work [29] for computation of McCormick relaxations of the given optimization problem.

– *S+K*, where every convex and/or concave relaxation is linearized iteratively using the adaptation of Kelley's algorithm with the initial set of linearization points $\mathscr{L}$ being determined with the use of the n-simplex algorithm. We add at most 5 additional linearizations with Kelley's algorithm.

– *Random*, where every convex and/or concave relaxation is linearized at p points determined randomly, where p is the maximum number of optimization variables participating nonlinearly in any function found in the optimization problem. Here we report the results for the best random seed among 5 different random seeds we used in C++.

In all numerical experiments, we set the maximum number of iterations in the Kelley's Algorithm 1 to p denoting the maximum number of optimization variables participating non-linearly in any function found in the optimization problem and the minimum difference thresholds to $\mu^a = \epsilon^a \cdot 10$ and $\mu^r = 10^{-2}$. For the determination of n-simplex vertices, we set the ball radius to $r = 0.725$ and rotate the n-simplex by $\gamma = 30°$ counterclockwise. Furthermore, in any algorithm, in addition to the at most p or $n + 1$ linearization points, we always use the mid point of the interval domain as linearization point. The detailed CPU times and number of iterations can be found in Table 7 in Appendix A. Since performance plots can be easily misinterpreted [14], we present the computational results with the use of the shifted geometric mean defined as

$$\exp\left(\sum_{i=1}^{n} \frac{\log(\max(1, t_i + s))}{n}\right) - s, \tag{1}$$

where n is the number of instances, t_i the reported computational time and s a constant shift set to $s = 10$ in this work. We also report the shifted geometric mean with t_i denoting the number of iterations needed. If a method reaches the maximum computational time of 8 hours, we use $t_i = 28800$ s or $t_i = 2 \cdot 10^6$ iterations (this is the maximum iterations needed among all runs that converged within the time limit of 8 hours). Table 1 shows the

Table 1 Tables reporting the correlation of the shifted geometric mean (1) between the considered methods with respect to CPU seconds and number of iterations

CPU sec.	Mid	Kelley	Simplex	S+K	Random
Geom mean	1.63	1.07	1	1.07	1.03
Iterations	Mid	Kelley	Simplex	S+K	Random
Geom mean	5.37	1.62	1.21	1	1.5

speed up differences among the considered methods. We see that the simple method using a single linearization point performs worst with respect to the number of iterations. It also performs worst with respect to the computational time in all but the smallest case studies. All methods are at least 50% faster than the single middle point linearization and need up to 5 times less iterations. The n-simplex method seems to perform best on the considered test set with respect to computational time, but all other presented method are only slightly weaker. Indeed the n-simplex method solves more instances than the other considered methods in the given time frame. The hybrid method has the lowest number of iterations but performs slightly worse in terms of computational time. Kelley's algorithm performs best regarding computational time for the methanol production processes but performs worse than the n-simplex algorithm on the other process engineering case studies. It also almost always needed strictly less than p iterations to converge to the preset tolerances. This may be an indicator that Kelley's algorithm is already good enough for our purposes but obviously this cannot be guaranteed. Note that for the minimization of the levelized cost of electricity (LCOE) in case 3, the methods which did not converge in time have a very small remaining gap. This may be misleading as for this particular problem, the last few percent needed to close the optimality gap are the most time consuming, meaning that the remaining $\approx 1\%$ would most likely need more than 1 additional hour to converge. It is also interesting that the random generation of points performed slightly better than the adaptation of Kelley's algorithm and the hybrid of n-simplex and Kelley's. This can be easily explained with the fact that the random seed we chose in C++ seems to work well for the instances considered. Unfortunately, this behavior cannot be guaranteed if a different machine or a different random seed is chosen, making the random generation of linearization points undesirable in general. We can conclude that except for the easiest instances or the instances with already very tight relaxations, the computation of additional linearization points seems favorable.

4 Hybridization of the auxiliary variable method and McCormick relaxations

Tsoukalas and Mitsos already discussed the relationship between the auxiliary variable method and multivariate McCormick relaxations (Section 4 in [37]). Their conclusion was that McCormick relaxations can be modified to provide equally tight relaxations as the AVM. The modification consists of the introduction of auxiliary optimization variables and corresponding equality constraints for common intermediate factors. We first briefly present the main result again herein. Since the introduction of additional optimization variables together with the corresponding equality constraints increases the complexity of the lower bounding problems, in a B&B framework it is favorable to use the linearization algorithms presented in the previous section. Then, we conduct additional numerical experiments to check the

impact of the introduction of common factors as auxiliary optimization variables when using multivariate McCormick relaxations.

4.1 Tightness of McCormick relaxations and the auxiliary variable method

Consider the functions

$$f_1 : \mathbb{R}^2 \ni X_1 \to \mathbb{R}, \qquad f_2 : \mathbb{R} \ni X_2 \to \mathbb{R},$$
$$f_3 : \mathbb{R} \ni X_3 \to \mathbb{R}, \qquad f_4 : \mathbb{R} \ni X_4 \to \mathbb{R},$$

and the composition

$$g : \mathbb{R} \ni X \to \mathbb{R}, x \mapsto f_1\left(f_3(x), f_4(x)\right) + f_2\left(f_3(x)\right),$$

with $X \subset X_3$, $X \subset X_4$, $f_3(X_3) \times f_4(X_4) \subset X_1$ and $f_3(X_3) \subset X_2$. When minimizing g, the AVM could formulate the problem in two different ways, depending on whether the intermediate factor f_3 is recognized as a common term (Formulation 2) or not (Formulation 1).

Formulation 1

$$\begin{aligned} \min_{\substack{x\in X, z_1\in X_1\\ z_2\in X_2, z_3\in X_3\\ z_3'\in X_3, z_4\in X_4}} \quad & z_1 + z_2 \\ \text{s.t.} \quad & z_1 = f_1(z_3, z_4) \\ & z_2 = f_2(z_3') \\ & z_4 = f_4(x) \\ & z_3 = f_3(x) \\ & z_3' = f_3(x) \end{aligned}$$

Formulation 2

$$\begin{aligned} \min_{\substack{x\in X, z_1\in X_1\\ z_2\in X_2, z_3\in X_3\\ z_4\in X_4}} \quad & z_1 + z_2 \\ \text{s.t.} \quad & z_1 = f_1(z_3, z_4) \\ & z_2 = f_2(z_3) \\ & z_4 = f_4(x) \\ & z_3 = f_3(x) \end{aligned}$$

with the corresponding convex relaxations used for the lower bounding problem

Formulation 1

$$\begin{aligned} \min_{\substack{x\in X, z_1\in X_1\\ z_2\in X_2, z_3\in X_3\\ z_3'\in X_3, z_4\in X_4}} \quad & z_1 + z_2 \\ \text{s.t.} \quad & f_1^{cv}(z_3, z_4) \le z_1 \le f_1^{cc}(z_3, z_4) \\ & f_2^{cv}(z_3') \le z_2 \le f_2^{cc}(z_3') \\ & f_4^{cv}(x) \le z_4 \le f_4^{cc}(x) \\ & f_3^{cv}(x) \le z_3 \le f_3^{cc}(x) \\ & f_3^{cv}(x) \le z_3' \le f_3^{cc}(x) \end{aligned}$$

Formulation 2

$$\begin{aligned} \min_{\substack{x\in X, z_1\in X_1\\ z_2\in X_2, z_3\in X_3\\ z_4\in X_4}} \quad & z_1 + z_2 \\ \text{s.t.} \quad & f_1^{cv}(z_3, z_4) \le z_1 \le f_1^{cc}(z_3, z_4) \\ & f_2^{cv}(z_3) \le z_2 \le f_2^{cc}(z_3) \\ & f_4^{cv}(x) \le z_4 \le f_4^{cc}(x) \\ & f_3^{cv}(x) \le z_3 \le f_3^{cc}(x) \end{aligned}$$

Formulation 1 is in general a relaxation of Formulation 2 and results in a worse lower bound if the nonlinear convex problem is solved. Although, theoretically the better lower bound cannot be guaranteed when the formulations are linearized, in practice it is still expected that the linearization of Formulation 2 will result in a tighter bound.

When using multivariate McCormick relaxations we obtain the same bound for Formulation 1

$$\min_{x\in X}\left\{\begin{array}{ll}\min\limits_{\hat{z}_1,\hat{z}_2} & \hat{z}_1+\hat{z}_2\\ \text{s.t.} & \begin{pmatrix}\min\limits_{\hat{z}_3,\hat{z}_4} & f_1^{cv}(\hat{z}_3,\hat{z}_4)\\ \text{s.t.} & f_3^{cv}(x)\le\hat{z}_3\le f_3^{cc}(x)\\ & f_4^{cv}(x)\le\hat{z}_4\le f_4^{cc}(x)\end{pmatrix}\le\hat{z}_1\le\begin{pmatrix}\max\limits_{\hat{z}_3,\hat{z}_4} & f_1^{cc}(\hat{z}_3,\hat{z}_4)\\ \text{s.t.} & f_3^{cv}(x)\le\hat{z}_3\le f_3^{cc}(x)\\ & f_4^{cv}(x)\le\hat{z}_4\le f_4^{cc}(x)\end{pmatrix}\\ & \begin{pmatrix}\min\limits_{\hat{z}'_3} & f_2^{cv}(\hat{z}'_3)\\ \text{s.t.} & f_3^{cv}(x)\le\hat{z}'_3\le f_3^{cc}(x)\end{pmatrix}\le\hat{z}_2\le\begin{pmatrix}\min\limits_{\hat{z}'_3} & f_2^{cc}(\hat{z}'_3)\\ \text{s.t.} & f_3^{cv}(x)\le\hat{z}'_3\le f_3^{cc}(x)\end{pmatrix}\end{array}\right\}.$$

Note that in the above multivariate McCormick formulation, we have only one optimization variable x. Now, if the intermediate factor $f_3(x)$ is recognized and treated as a common variable, it is necessary to introduce one additional optimization variable z_3. The alternative formulation we get then is

$$\min_{x\in X,z_3\in Z_3}\left\{\begin{array}{ll}\min\limits_{\hat{z}_1,\hat{z}_2} & \hat{z}_1+\hat{z}_2\\ \text{s.t.} & \begin{pmatrix}\min\limits_{\hat{z}_4} & f_1^{cv}(z_3,\hat{z}_4)\\ \text{s.t.} & f_4^{cv}(x)\le\hat{z}_4\le f_4^{cc}(x)\end{pmatrix}\le\hat{z}_1\le\begin{pmatrix}\max\limits_{\hat{z}_4} & f_1^{cc}(z_3,\hat{z}_4)\\ \text{s.t.} & f_4^{cv}(x)\le\hat{z}_4\le f_4^{cc}(x)\end{pmatrix}\\ & f_2^{cv}(x)\le\hat{z}_2\le f_2^{cc}(x)\\ & f_3^{cv}(x)\le z_3\le f_3^{cc}(x)\end{array}\right\}$$

providing the same bound as Formulation 2 in the AVM by only introducing a single additional auxiliary optimization variable. This also means that the tightness of the linear lower bounding problem obtained through linearization of the convex multivariate McCormick relaxation is equal to the bound obtained by the linearized AVM Formulation 2 if the same linearization points are used (which in general we do not recommend due to the combinatorial complexity (Sect. 3.1)). Although not explicitly mentioned in the original publication [37], the above formulation already suggests that a hybridization of the AVM and the multivariate McCormick relaxations may result in computational advantages. In this work, instead of trying to achieve the tightness of the AVM, we experiment with the addition of only a few auxiliary optimization variables in order to achieve best computational results.

4.2 Numerical results

Modern implementations of directed acyclic graphs such as the one in `MC++` [7] allow for the recognition of common factors. The recognition is done automatically and has a quadratic complexity, since in the worst-case all previous factors have to be checked whenever a new factor is introduced. When comparing the AVM with McCormick relaxations, it is of interest to investigate the impact of the intermediate common factors on computational time in a B&B algorithm. Note that the addition of all intermediate factors, also those occurring only once, results in the AVM.

In our experiments, we always choose the intermediate nonlinear factors participating most often in the optimization problems and replace them by an auxiliary optimization variable and a corresponding equality constraint in the lower bounding problem. We use the same optimization problems as in Sect. 3.5 with the exact same computational setup. In our experiments we add up to 3 auxiliary variables and solve the problems with the same 5 presented linearization strategies as in Sect. 3.5. However, we do not branch on these auxiliary variables but rather only apply bound tightening heuristics to them. Initial valid bounds for

Table 2 Tables reporting the correlation of the shifted geometric mean (1) between the considered methods with respect to CPU seconds and number of iterations when exactly 1, 2 or 3 auxiliary optimization variables are added

Geometric mean w.r.t. CPU seconds					
1 aux added	Mid	Kelley	Simplex	S+K	Random
Geom mean	1.47	1.05	1	1.14	1.03
2 aux added	Mid	Kelley	Simplex	S+K	Random
Geom mean	1.48	1.006	1	1.25	1.01
3 aux added	Mid	Kelley	Simplex	S+K	Random
Geom mean	1.41	1	1.06	1.18	1.02
Geometric mean w.r.t. number of iterations					
1 aux added	Mid	Kelley	Simplex	S+K	Random
Geom mean	4.22	1.61	1.12	1	1.16
2 aux added	Mid	Kelley	Simplex	S+K	Random
Geom mean	4	1.43	1.02	1	1.05
3 aux added	Mid	Kelley	Simplex	S+K	Random
Geom mean	3.92	1.44	1.1	1	1.06

aux denotes auxiliary variable(s)

Table 3 Tables reporting the correlation of the shifted geometric mean (1) between the presented methods considering all numbers of additional auxiliary variables (1–3) with respect to CPU seconds and number of iterations

Geometric mean considering all numbers of additional aux. vars. (1–3)					
CPU sec.	Mid	Kelley	Simplex	S+K	Random
Geom mean	1.42	1	1.001	1.16	1.003
Iterations	Mid	Kelley	Simplex	S+K	Random
Geom mean	1.72	1.17	1.03	1	1.03

the introduced variables are obtained through interval arithmetic and constraint propagation. We report only the problems where at least one common intermediate factor has been found. Again, we use the shifted geometric mean (cf. Sect. 3.5) to compare the different linearization strategies for a fixed number of additional auxiliary optimization variables and also to compare the same linearization strategies for the different numbers of introduced auxiliary optimization variables (cf. Tables 2, 4). Table 3 reports the shifted geometric mean comparing the different linearization strategies over all numbers of additional auxiliary variables (1–3). Table 5 reports the shifted geometric mean comparing the different linearization strategies and all different number of auxiliary variables (0–3).

In the cases where 1 or 2 auxiliary optimization variables are added, the n-simplex method outperforms all other considered procedures and that it is the only one to solve all considered numerical examples in the time frame of 8 hours (cf. Tables 8, 9, 10 found in "Appendix A") with up to 2 auxiliary variables. The adaptation of Kelley's algorithm performs best when 3 auxiliary variables are added, but in general all procedures perform worse than when less auxiliaries are used. Overall, Kelley's and the n-simplex algorithm perform very similar when auxiliary variables are added (cf. Tables 3, 5). The introduction of auxiliary variables allows an improvement of up to $\approx 20\%$ with respect to solution times for all procedures (cf.

Table 4 Tables reporting the correlation of the shifted geometric mean (1) when only the mid point of the domain box is used as linearization point; when the adapted Kelley's algorithm (Sect. 3.2) is used to linearize the lower bounding problem; the n-simplex algorithm (Sect. 3.3) is used compute the linearization points; and when the adapted Kelley's algorithm (Sect. 3.2) with the initial set of linearization points computed with the n-simplex algorithm (Sect. 3.3) is used to linearize the lower bounding problem; and when random linearization points are used. We consider the case when 0–3 auxiliary variables are introduced and compare the each method individually in terms of CPU seconds and number of iterations

Geometric mean w.r.t. CPU seconds

Mid	0 aux	1 aux	2 aux	3 aux
Geom mean	1.14	1.02	1.03	1
Kelley	0 aux	1 aux	2 aux	3 aux
Geom mean	1.19	1.03	1	1.006
Simplex	0 aux	1 aux	2 aux	3 aux
Geom mean	1.22	1	1.009	1.09
S+K	0 aux	1 aux	2 aux	3 aux
Geom mean	1.21	1	1.1	1.06
Random	0 aux	1 aux	2 aux	3 aux
Geom mean	1.19	1.01	1	1.02

Geometric mean w.r.t. number of iterations

Mid	0 aux	1 aux	2 aux	3 aux
Geom mean	1.82	1.19	1.12	1
Kelley	0 aux	1 aux	2 aux	3 aux
Geom mean	1.97	1.12	1.09	1
Simplex	0 aux	1 aux	2 aux	3 aux
Geom mean	1.93	1.12	1.02	1
S+K	0 aux	1 aux	2 aux	3 aux
Geom mean	1.99	1.11	1.1	1
Random	0 aux	1 aux	2 aux	3 aux
Geom mean	2.06	1.21	1.08	1

aux denotes auxiliary variable(s)

Table 4). It is also comprehensible that linearization strategies using multiple linearizations achieve computational advantages when introducing auxiliary optimization variables when compared to the simple single point linearization due to the increased dimensionality and thus also an increased complexity of the optimization problem. Unfortunately, we cannot say that introducing more auxiliary variables always results in best performance as, e.g., when introducing auxiliary variables for the minimization of the levelized cost of electricity in case 3 [1], we achieve the best computational performance when adding only 1 auxiliary variable. However, the number of iterations needed decreases as more auxiliary optimization variables are added, confirming the improved tightness of the resulting lower bounding relaxations. This may be an indicator that replacing the common factors occurring the most in the optimization problem is not always the best heuristic but rather different properties are crucial for the computational advantage. It is also possible that it is required to branch on the auxiliary variables to possibly further improve the computational performance. The improvement of the choice of intermediate factors and the development of specialized heuristics for the hybridization of McCormick relaxations with the AVM requires further investigation and remains an active research topic of the authors.

The present results for the NRTL flash case (NRTL_RS_IdealGasFlash) also provide an explanation for the differences in performance of the model formulations considered by

Table 5 Tables reporting the correlation of the shifted geometric mean (1) when only the mid point of the domain box is used as linearization point; when the adapted Kelley's algorithm (Sect. 3.2) is used to linearize the lower bounding problem; the n-simplex algorithm (Sect. 3.3) is used compute the linearization points; and when the adapted Kelley's algorithm (Sect. 3.2) with the initial set of linearization points computed with the n-simplex algorithm (Sect. 3.3) is used to linearize the lower bounding problem; and when random linearization points are used. We consider the case when 0–3 auxiliary variables are introduced and compare all combinations of method and number of auxiliary variables in terms of the CPU seconds and number of iterations. *aux* denotes auxiliary variable(s)

Geometric mean w.r.t. CPU seconds

	0 aux	1 aux	2 aux	3 aux
Mid	1.69	1.47	1.49	1.44
Kelley	1.21	1.05	1.01	1.02
Simplex	1.22	1	1.009	1.09
S+K	1.39	1.14	1.26	1.21
Random	1.21	1.03	1.02	1.04

Geometric mean w.r.t. number of iterations

	0 aux	1 aux	2 aux	3 aux
Mid	7.15	4.7	4.41	3.92
Kelley	2.85	1.79	1.58	1.44
Simplex	2.14	1.25	1.13	1.1
S+K	1.99	1.11	1.1	1
Random	2.2	1.29	1.15	1.06

Bongartz and Mitsos [3]. In the version of the case study used herein, only the liquid mole fractions of the flash are optimization variables while the vapor mole fractions are calculated from the isofugacity equations. Bongartz and Mitsos [3] observed that adding the vapor mole fractions as optimization variables, thus turning the isofugacity equations into constraints, improved computational performance. When allowing the addition of auxiliary variables, the variables added herein correspond exactly to these vapor mole fractions, showing that their addition to the problem results in tighter relaxations because they are re-occurring factors.

5 Conclusion

Algorithms for the computation of linearization points for the auxiliary variable method are available and are widely used in state-of-the-art deterministic global optimization solvers [35]. No such methods have been proposed for the (multivariate) McCormick relaxations. We discuss difficulties and limits of methods for the computation of affine underestimators of McCormick relaxations when the idea of subgradient propagation is considered [28]. An iterative method based on well-known cutting plane algorithms from the field of convex optimization [8,21] and an own algorithm based on the computation of all vertices of an n-simplex for the computation of linearization points a priori are developed. We combine both methods and compare all procedures in numerical experiments. The n-simplex algorithms performs best on the considered test set, but all method perform up to $\approx 60\%$ better than the single point linearization strategy used in our previous works. Furthermore we tie on the discussion on the relaxation tightness and introduction of auxiliary variables in the AVM and McCormick relaxations presented in [37]. The theoretical results are applied in the numerical experiments by recognizing common intermediate factors in the directed acyclic graph and introducing additional optimization variables while still using the McCormick propagation

rules. The introduction of a limited number of auxiliary variables provides large improvements in the computational times. We deduce that it is not always best (w.r.t. computational time) to introduce all commonly occurring factors as auxiliary optimization variables. The development of methods for the determination of when to replace a common factor by an auxiliary optimization variable remains an active field of research of the authors.

Acknowledgements We would like to thank Dr. Benoît Chachuat for providing MC++ and for discussions on polyhedral relaxations. We would like to thank Prof. Angelos Tsoukalas for discussions on localization methods. This project has received funding from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) *Improved McCormick Relaxations for the efficient Global Optimization in the Space of Degrees of Freedom* MA 1851/4-1. We gratefully acknowledge additional funding by the German Federal Ministry of Education and Research (BMBF) within the "Kopernikus Project P2X: Flexible use of renewable resources exploration, validation and implementation of 'Power-to-X' concepts".

Funding Open Access funding enabled and organized by Projekt DEAL.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

A Appendix

See Tables 6, 7, 8, 9 and 10.

Table 6 Table summarizing the problems for the numerical studies

Name	#var	#ineq	#eq
bearing	13	3	9
case2_lcoe [1]	5	12	1
case2_wnet [1]	5	7	1
case3_lcoe [1]	8	22	1
case3_wnet [1]	8	14	1
chenery	43	6	32
ex5_4_4	27	0	19
ex6_1_3	12	0	9
ex6_2_10	6	0	3
ex6_2_7	10	0	3
ex7_2_3	8	6	0
ex7_3_4	12	10	7
ex8_2_1b	57	31	2
NRTL_RS_IdealGasFlash [3]	6	1	4
Process_Detailed [3]	51	6	46
Process_Detailed_with_CO2 [3]	52	6	46
William_Otto_4 [2]	13	1	9
William_Otto_5 [2]	14	1	9

#var represents the number of variables, #ineq stands for the number of inequalities and #eq for the number of equalities

Table 7 The numerical results for different linearization strategies conducted as described in Sect. 3.5

Name	Mid		Kelley		Simplex		S+K		Random	
	#iter	CPU[s]	#iter	CPU[s]	#iter	CPU[s]	#iter	CPU[s]	#iter	CPU[s]
bearing	10,603	8.305	156	0.954	219	1.22	125	1.142	10520	16.204
case2_lcoe [1]	3783	4.987	3117	7.95	2677	6.501	2353	9.358	2733	6.652
case2_wnet [1]	145	0.207	77	0.259	69	0.186	55	0.274	91	0.232
case3_lcoe [1]	97.81%	28,800	98.18%	28,800	1.6e6	27713.4	98.99%	28800	98.63%	28,800
case3_wnet [1]	10,935	20.077	6611	30.583	4123	18.897	3783	27.477	4345	19.829
chenery	2045	38.792	75	3.359	209	14.175	29	3.273	217	14.775
ex5_4_4	429,169	4847.05	20,087	413.141	20,265	378.376	11061	284.45	18,595	350.642
ex6_1_3	3633	10.108	1975	14.559	1413	10.613	1399	15.446	1423	11.244
ex6_2_10	26475	32.316	24,057	49.608	19,237	42.527	18897	60.125	18371	41.496
ex6_2_7	14.77%	28,800	15.59%	28,800	16.12%	28,800	16.59%	28,800	17.83%	28,800
ex7_2_3	71,017	71.69	57,113	77.221	55,609	73.1	49,459	88.745	45,701	59.22
ex7_3_4	23,7991	754.934	10,873	56.669	559	3.18	487	3.969	485	2.748
ex8_2_1b	3	14.738	1	15.037	1	15.243	1	15.282	1	15.088
NRTL_RS_IdealGasFlash [3]	2721	5.791	1855	9.568	1773	7.294	1741	11.372	1911	7.815
Process_Detailed [3]	23,691	1501.02	3301	765.178	1141	979.333	1263	1303.33	1191	1094.24
Process_Detailed_with_CO2 [3]	186,059	13604.2	25,389	5898.19	12,061	11,525.6	12,287	13,767.2	9413	9634.69
William_Otto_4 [2]	165,165	879.264	44,087	571.624	38607	529.542	35425	678.719	39073	531.333
William_Otto_5 [2]	36.59%	28,800	63.68%	28,800	70.66%	28,800	64.66%	28,800	79.18%	28,800

#iter denotes the number of iterations. If the maximum CPU time of 8 hours = 28800 seconds has been reached, we report the final ratio computed as lower bound divided by upper bound

Table 8 The numerical results for different linearization strategies conducted as described in Sect. 3.5 when exactly 1 auxiliary variables are introduced as described in Sect. 4

Name	Mid		Kelley		Simplex		S+K		Random	
	#iter	CPU[s]	#iter	CPU[s]	#iter	CPU[s]	#iter	CPU[s]	#iter	CPU[s]
case2_lcoe [1]	3669	5.848	2801	8.703	2703	7.859	2349	10.741	2705	7.893
case2_wnet [1]	157	0.271	79	0.321	85	0.256	57	0.301	75	0.245
case3_lcoe [1]	98.37%	28,800	98.61%	28,800	975,789	13885.3	816969	13509	1345460	20586.3
case3_wnet [1]	2457	5.776	1257	6.898	909	5.125	759	6.646	945	5.387
ex5_4_4	275,673	3019.53	14,339	310.056	10,003	200.567	8567	233.442	12,255	255.115
ex6_2_10	26,569	36.385	23,907	60.611	20,385	54.378	20,007	76.941	18,255	48.461
NRTL_RS_IdealGasFlash [3]	201	0.545	117	0.796	103	0.705	101	1.051	141	0.868
Process_Detailed [3]	23,677	1652.18	3287	853.992	1149	1079.67	1217	1373.75	1181	1177.06
Process_Detailed_with_CO2 [3]	186,011	15272.6	24,851	6514.94	12,643	12,668.5	13571	15958.5	10511	11499.9
William_Otto_4 [2]	139,101	807.127	32,313	469.433	32,155	515.297	26307	587.63	33877	536.424
William_Otto_5 [2]	41.65%	28,800	96.74%	28,800	1,092,550	24,819.2	916,705	27,341.1	925,387	20,822.9

#iter denotes the number of iterations. If the maximum CPU time of 8 hours = 28800 seconds has been reached, we report the final ratio computed as lower bound divided by upper bound

Table 9 The numerical results for different linearization strategies conducted as described in Sect. 3.5 when exactly 2 auxiliary variables are introduced as described in Sect. 4

Name	Mid		Kelley		Simplex		S+K		Random	
	#iter	CPU[s]	#iter	CPU[s]	#iter	CPU[s]	#iter	CPU[s]	#iter	CPU[s]
case2_lcoe [1]	3333	6.438	2169	8.269	2499	8.709	2051	11.894	2335	8.821
case2_wnet [1]	145	0.341	69	0.329	75	0.296	51	0.335	83	0.316
case3_lcoe [1]	98.28%	28,800	98.58%	28,800	1,330,320	23,133.4	1117790	24224.9	98.93%	28800
case3_wnet [1]	1759	4.827	839	5.591	625	4.356	527	5.693	581	4.096
ex5_4_4	149,065	1639.14	7787	175.221	5341	116.505	5109	150.293	4657	110.833
ex6_2_10	26,551	42.223	23,413	68.426	20,851	63.723	20,161	89.76	18,063	57.55
NRTL_RS_IdealGasFlash [3]	199	0.631	113	0.882	105	0.77	101	1.078	133	0.902
Process_Detailed [3]	36,345	2897.9	4065	1048.83	1145	1077.94	2653	2962.65	1209	1188.84
Process_Detailed_with_CO2 [3]	184,219	14,753.2	21,049	5311.69	11,635	11,631.6	12261	14273.2	9727	10567.4
William_Otto_4 [2]	140,131	919.677	32,459	539.216	33941	610.43	29,237	735.929	33,887	615.038
William_Otto_5 [2]	41%	28,800	92.47%	28,800	881,471	21,993.1	815,493	25329	925,929	22,366.5

#iter denotes the number of iterations. If the maximum CPU time of 8 hours = 28800 seconds has been reached, we report the final ratio computed as lower bound divided by upper bound

Table 10 The numerical results for different linearization strategies conducted as described in Sect. 3.5 when exactly 3 auxiliary variables are introduced as described in Sect. 4

Name	Mid		Kelley		Simplex		S+K		Random	
	#iter	CPU[s]	#iter	CPU[s]	#iter	CPU[s]	#iter	CPU[s]	#iter	CPU[s]
case2_lcoe [1]	3263	8.336	2071	10.354	2547	12.305	2225	17.362	2289	11.71
case2_wnet [1]	143	0.394	67	0.407	69	0.329	55	0.429	83	0.404
case3_lcoe [1]	98.33%	28,800	98.63%	28,800	98.83%	28,800	98.87%	28800	98.88%	28800
case3_wnet [1]	1759	6.015	785	6.498	567	5.015	505	6.608	547	4.914
ex5_4_4	69,035	837.047	3747	98.505	2637	69.46	2639	86.686	2219	62.303
ex6_2_10	26,621	56.092	23,239	88.102	19,497	80.074	18,913	113.152	18,013	75.718
NRTL_RS_IdealGasFlash [3]	189	0.733	123	1.063	105	0.911	97	1.199	139	1.093
Process_Detailed [3]	23,333	1813.59	3291	933.238	1341	1333.75	1195	1384.82	1197	1260.21
Process_Detailed_with_CO2 [3]	184,249	16,887	21,317	6389.03	12,937	14,107.1	10331	13427.6	8917	10603.4
William_Otto_4 [2]	140,339	1067.94	32,443	607.578	31,937	647.733	28593	800.386	33943	702.403
William_Otto_5 [2]	40.73%	28,800	88.62%	28,800	948,137	26522.7	822,053	28,800	927,037	24,698.5

#iter denotes the number of iterations. If the maximum CPU time of 8 hours = 28800 seconds has been reached, we report the final ratio computed as lower bound divided by upper bound

References

1. Bongartz, D., Mitsos, A.: Deterministic global optimization of process flowsheets in a reduced space using mccormick relaxations. J. Global Optim. **69**(4), 761–796 (2017)
2. Bongartz, D., Mitsos, A.: Infeasible Path Global Flowsheet Optimization Using McCormick Relaxations. In: A. Espuña, M. Graells, L. Puigjaner (eds.) Computer Aided Chemical Engineering: 27th European Symposium on Computer Aided Chemical Engineering (ESCAPE 27), pp. 631–636 (2017). https://doi.org/10.1016/B978-0-444-63965-3.50107-0
3. Bongartz, D., Mitsos, A.: Deterministic global flowsheet optimization: between equation-oriented and sequential-modular methods. AIChE J. **65**(3), 1022–1034 (2019)
4. Bongartz, D., Najman, J., Sass, S., Mitsos, A.: MAiNGO - McCormick-based Algorithm for mixed-integer Nonlinear Global Optimization. Tech. rep., Process Systems Engineering (AVT.SVT), RWTH Aachen University (2018). http://permalink.avt.rwth-aachen.de/?id=729717, MAiNGO Git: https://git.rwth-aachen.de/avt.svt/public/maingo
5. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
6. Cao, H., Song, Y., Khan, K.A.: Convergence of subtangent-based relaxations of nonlinear programs. Processes **7**(4), 221 (2019). https://doi.org/10.3390/pr7040221
7. Chachuat, B., Houska, B., Paulen, R., Perić, N., Rajyaguru, J., Villanueva, M.: Set-theoretic approaches in analysis, estimation and control of nonlinear systems. IFAC-PapersOnLine **48**(8), 981–995 (2015)
8. Cheney, E.W., Goldstein, A.A.: Newton's method for convex programming and Tchebycheff approximation. Numer. Math. **1**(1), 253–268 (1959)
9. Chi, H., Mascagni, M., Warnock, T.: On the optimal Halton sequence. Math. Comput. Simul. **70**(1), 9–21 (2005)
10. Dantzig, G.B., Orden, A., Wolfe, P., et al.: The generalized simplex method for minimizing a linear form under linear inequality restraints. Pac. J. Math. **5**(2), 183–195 (1955)
11. Elzinga, J., Moore, T.G.: A central cutting plane algorithm for the convex programming problem. Math. Prog. **8**(1), 134–145 (1975)
12. Gleixner, A., Berthold, T., Müller, B., Weltge, S.: Three enhancements for optimization-based bound tightening. J. Global Optim. **67**, 731–757 (2017)
13. Goffin, J.L., Vial, J.P.: On the computation of weighted analytic centers and dual ellipsoids with the projective algorithm. Math. Prog. **60**(1–3), 81–92 (1993)
14. Gould, N., Scott, J.: A note on performance profiles for benchmarking software. ACM Trans. Math. Softw. **43**(2), 15:1–15:5 (2016)
15. Hastings, W.K.: Monte carlo sampling methods using markov chains and their applications. Biometrika **57**(1), 97–109 (1970)
16. Hiriart-Urruty, J.B., Lemaréchal, C.: Convex Analysis and Minimization Algorithms II Advanced Theory and Bundle Methods, vol. 306. Springer, Berlin (1993). https://doi.org/10.1007/978-3-662-06409-2
17. Horst, R., Tuy, H.: Global Optimization: Deterministic Approaches. Springer, Berlin (2013)
18. International Business Machines Corporation: IBM ILOG CPLEX v12.8. Armonk, NY (2017)
19. Joe, S., Kuo, F.Y.: Constructing Sobol sequences with better two-dimensional projections. SIAM J. Sci. Comput. **30**(5), 2635–2654 (2008)
20. Johnson, S.: The NLopt nonlinear-optimization package (2016). http://ab-initio.mit.edu/nlopt. Last Accessed 13 Sept 2018
21. Kelley Jr., J.E.: The cutting-plane method for solving convex programs. J. Soc. Ind. Appl. Math. **8**(4), 703–712 (1960)
22. Khajavirad, A., Sahinidis, N.V.: A hybrid LP/NLP paradigm for global optimization relaxations. Math. Prog. Comput. **10**(3), 383–421 (2018)
23. Khan, K., Watson, H., Barton, P.: Differentiable McCormick relaxations. J. Global Optim. **67**(4), 687–729 (2017)
24. Levin, A.Y.: An algorithm for minimizing convex functions. In: Doklady Akademii Nauk, vol. 160, pp. 1244–1247. Russian Academy of Sciences (1965)
25. Locatelli, M., Schoen, F.: Global optimization: theory, algorithms, and applications, vol. 15. SIAM (2013)
26. McCormick, G.: Computability of global solutions to factorable nonconvex programs: part I-convex underestimating problems. Math. Prog. **10**, 147–175 (1976)
27. Misener, R., Floudas, C.: ANTIGONE: algorithms for coNTinuous/integer global optimization of nonlinear equations. J. Global Optim. **59**, 503–526 (2014)
28. Mitsos, A., Chachuat, B., Barton, P.: McCormick-based relaxations of algorithms. SIAM J. Optim. **20**(2), 573–601 (2009)
29. Najman, J., Mitsos, A.: Tighter McCormick relaxations through subgradient propagation. J. Global Optim. (2019). https://doi.org/10.1007/s10898-019-00791-0

30. Newman, D.J.: Location of the maximum on unimodal surfaces. JACM **12**(3), 395–398 (1965)
31. Ryoo, H., Sahinidis, N.: Global optimization of nonconvex NLPs and MINLPs with applications in process design. Comput. Chem. Eng. **19**(5), 551–566 (1995)
32. Schichl, H., Neumaier, A.: Interval analysis on directed acyclic graphs for global optimization. J. Global Optim. **33**(4), 541–562 (2005)
33. Smith, E.M., Pantelides, C.C.: Global optimisation of nonconvex MINLPs. Comput. Chem. Eng. **21**, 791–796 (1997)
34. Tarasov, S.: The method of inscribed ellipsoids. Soviet Math. Doklady **37**, 226–230 (1988)
35. Tawarmalani, M., Sahinidis, N.: Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming. Kluwer Academic Publishers, Dordrecht (2002)
36. Tawarmalani, M., Sahinidis, N.V.: A polyhedral branch-and-cut approach to global optimization. Math. Prog. **103**(2), 225–249 (2005)
37. Tsoukalas, A., Mitsos, A.: Multivariate McCormick relaxations. J. Global Optim. **59**, 633–662 (2014)
38. Vigerske, S., Gleixner, A.: SCIP: global optimization of mixed-integer nonlinear programs in a branch-and-cut framework. Optim. Methods Softw. **33**(3), 563–593 (2018)
39. Wächter, A., Biegler, L.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Prog. **106**(1), 25–57 (2006)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.