Journal of Global Optimization (2025) 92:837–888
https://doi.org/10.1007/s10898-025-01489-2

MUSE-BB: a decomposition algorithm for nonconvex two-stage problems using strong multisection branching

Marco Langiu[1,2] · **Manuel Dahmen**[2] · **Dominik Bongartz**[3] · **Alexander Mitsos**[1,2,4]

Received: 28 June 2024 / Accepted: 30 March 2025 / Published online: 5 May 2025

Abstract
We present MUSE-BB, a branch-and-bound (B&B) based decomposition algorithm for the deterministic global solution of nonconvex two-stage stochastic programming problems. In contrast to three recent decomposition algorithms, which solve this type of problem in a projected form by nesting an inner B&B in an outer B&B on the first-stage variables, we branch on all variables within a single B&B tree. This results in a higher convergence order of the lower bounding scheme, avoids repeated consideration of subdomains, inherent to the nesting of B&B searches, and enables the use of cheaper subproblems. In particular, when branching on second-stage variables, we employ a multisection variant of strong-branching, in which we simultaneously consider one candidate variable from each scenario for branching. By our decomposable lower bounding scheme, the resulting subproblems are independent and can be solved in parallel. We then use strong-branching scores to filter less promising candidate variables and only generate child nodes corresponding to a multisection involving the remaining variables by combining the appropriate subproblem results. We prove finite ε_f-convergence, and demonstrate that the lower-bounding scheme of MUSE-BB has at least first-order convergence under the mild assumption of Lipschitz continuous functions and relaxations. MUSE-BB is implemented and made available open source, as an extension of our deterministic global solver for mixed-integer nonlinear programs, MAiNGO, with OpenMP-parallelization of the decomposable subroutines. Numerical results show that MUSE-BB requires less CPU time than solving the deterministic equivalent using the standard version

✉ Alexander Mitsos
alexander.mitsos@avt.rwth-aachen.de

Marco Langiu
marco.langiu@rwth-aachen.de

Manuel Dahmen
m.dahmen@fz-juelich.de

Dominik Bongartz
dominik.bongartz@kuleuven.be

1 Process Systems Engineering (AVT.SVT), RWTH Aachen University, 52074 Aachen, Germany

2 Institute of Climate and Energy Systems, Energy Systems Engineering (ICE-1), Forschungszentrum Jülich GmbH, 52425 Jülich, Germany

3 Department of Chemical Engineering, KU Leuven, 3001 Leuven, Belgium

4 JARA-ENERGY, 52425 Jülich, Germany

of MAiNGO; moreover, the parallelized decomposition allows for further reduction in wall time.

Keywords Two-stage stochastic programming · Decomposition · Multisection · Convergence analysis · Clustering

List of symbols

$\mathcal{F}$	Feasible set of an optimization problem
f	Objective function
$\boldsymbol{g}$	Generic constraint function vector
$\boldsymbol{h}$	Nonanticipativity constraints
$\mathcal{L}$	List of k scenarios for which both sibling subproblems are feasible, the associated second-stage variable instances will be branched, producing 2^k orthant nodes
$\mathcal{M}$	Map from scenarios producing exactly one infeasible sibling subproblem to the sibling n with a feasible subproblem, the associated second-stage variable instances will be branched, but do not add to the number of orthant nodes generated
N	Number, e.g., of first-stage variables (subscript x) or second-stage constraints (subscript II)
n	Node of the B&B tree
$\mathcal{N}$	Set of open nodes of the B&B tree
s	Scenario
$\check{\nabla}$	Subgradient
$\mathcal{X}$	Domain of $\boldsymbol{x}$
$\boldsymbol{x}$	First-stage variables
$\mathcal{Y}$	Domain of $\boldsymbol{y}$
$\boldsymbol{y}$	Second-stage variables

Subscripts

$\underline{\bullet}$	Related to lower bound
I	Related to first-stage
II	Related to second-stage

Superscripts

$\overline{\bullet}$	Related to upper bound
cv	Convex relaxation
†	Related to incumbent
*	Related to optimal solution

1 Introduction

A standard formulation for optimization under uncertainty is two-stage stochastic programming [1], typically applied when long-term ("here and now") decisions are taken prior to the realization of uncertain scenarios, and then recourse ("wait and see") decisions are taken in response to the realized scenario. This paradigm may also be applied in situations where future events can be expected to occur with a particular *frequency*, i.e., the scenarios do not represent an uncertain, but rather time-variable future, as in design and operation problems in process engineering [2, 3]. In the following we will not distinguish between the two cases and treat both via "probabilities".

1.1 Problem formulation and notation

The overall two-stage problem (TSP) takes the form

$$f^{\mathcal{X},\mathcal{Y}} := \min_{\boldsymbol{x}\in\mathcal{X}} f_{\mathrm{I}}(\boldsymbol{x}) + \sum_{s\in\mathcal{S}} w_s\, f_{\mathrm{II},s}^{\mathcal{Y}_s}(\boldsymbol{x}) \quad \text{s. t. } \boldsymbol{g}_{\mathrm{I}}(\boldsymbol{x}) \leq \boldsymbol{0}, \tag{TSP$^{\mathcal{X},\mathcal{Y}}$}$$

where $\mathcal{X} \subsetneq \mathbb{R}^{N_x}$, $\mathcal{Y}_s \subsetneq \mathbb{R}^{N_y}$, for each $s \in \mathcal{S}$, and $\mathcal{Y} \subsetneq \mathbb{R}^{N_s N_y}$, such that $\mathcal{X}$ and $\mathcal{Y}{:=}\bigtimes_{s\in\mathcal{S}} \mathcal{Y}_s$, are bounded hyperrectangles, and thus compact sets. The set of considered scenarios $\mathcal{S}$ is assumed to have finite cardinality $N_s{:=}|\mathcal{S}| \geq 1$, and each element $s \in \mathcal{S}$ is assigned a probability $w_s \in (0, 1]$, $\sum_{s\in\mathcal{S}} w_s = 1$. Throughout this work, we omit variable domains and other parameters in references to optimization problems, if they are irrelevant, e.g., as in TSP. The limitation to values in $(0, 1]$ makes the sum in the objective a convex combination, allowing for more concise definitions and proofs. Other weights can be equivalently used via appropriate scaling and redefinition of the objective.

For each scenario s, the value of *second-stage optimal value function* $f_{\mathrm{II},s}^{\mathcal{Y}_s}$, also known as *optimal recourse function* corresponds to the optimal objective value of the following recourse problem (RP) for a fixed value of $\boldsymbol{x}$, and second-stage domain $\mathcal{Y}_s$:

$$f_{\mathrm{II},s}^{\mathcal{Y}_s}(\boldsymbol{x}) := \min_{\boldsymbol{y}_s\in\mathcal{Y}_s} f_{\mathrm{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) \quad \text{s. t. } \boldsymbol{g}_{\mathrm{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) \leq \boldsymbol{0}. \tag{RP$_s^{\mathcal{Y}_s}(\boldsymbol{x})$}$$

The decisions that need to be made in the first and second stage are captured by the variable vectors $\boldsymbol{x}$ and $\boldsymbol{y}_s$, respectively. As a result, the TSP$^{\mathcal{X},\mathcal{Y}}$ and its optimal value $f^{\mathcal{X},\mathcal{Y}}$ are parameterized by the domains $\mathcal{X}$ and $\mathcal{Y}$, whereas RP$_s^{\mathcal{Y}_s}(\boldsymbol{x})$ and its optimal value $f_{\mathrm{II},s}^{\mathcal{Y}_s}(\boldsymbol{x})$ are parameterized by fixed first-stage variable values $\boldsymbol{x}$ and the domain $\mathcal{Y}_s$. $f_{\mathrm{I}} : \mathcal{X} \mapsto \mathbb{R}$ and $f_{\mathrm{II},s} : \mathcal{X} \times \mathcal{Y}_s \mapsto \mathbb{R}$ denote the scalar-valued first- and second-stage objective functions, and $\boldsymbol{g}_{\mathrm{I}} : \mathcal{X} \mapsto \mathbb{R}^{N_{\mathrm{I}}}$ and $\boldsymbol{g}_{\mathrm{II},s} : \mathcal{X} \times \mathcal{Y}_s \mapsto \mathbb{R}^{N_{\mathrm{II}}}$ the vector-valued first- and second-stage constraint functions. Note that we assume the number of second-stage variables N_y, and second-stage constraints N_{II}, to be equal for all scenarios. This assumption is naturally satisfied in many applications of two stage problems, e.g., system design and operation. While the generalization to different numbers $N_{y,s}$, and $N_{\mathrm{II},s}$, for each scenario s does not pose substantial complication, we only consider the simpler case for ease of exposition.

For conciseness, we aggregate the vectors $\boldsymbol{y}_s$ into the overall vector of second-stage variables $\boldsymbol{y} \in \mathcal{Y}$:

$$\boldsymbol{y} := \begin{pmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_{N_s} \end{pmatrix} = \begin{pmatrix} y_{1,1} \\ \vdots \\ y_{1,N_y} \\ \vdots \\ y_{N_s,N_y} \end{pmatrix} \tag{$\boldsymbol{y}$}$$

We denominate any scalar element $y_{s,i}$ of $\boldsymbol{y}$ or $\boldsymbol{y}_s$ as a *second-stage variable* and refer to the collection of elements at the same position i in $\boldsymbol{y}_s$ for different values of s as *instances of a second-stage variable*. Furthermore, we define the *scenario objective functions* $f_s : \mathcal{X} \times \mathcal{Y}_s \mapsto \mathbb{R}$

$$f_s(\boldsymbol{x}, \boldsymbol{y}_s) := f_{\mathrm{I}}(\boldsymbol{x}) + f_{\mathrm{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) \tag{f_s}$$

and the *overall objective function* $f : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$

$$\begin{aligned} f(\boldsymbol{x}, \boldsymbol{y}) &:= f_{\mathrm{I}}(\boldsymbol{x}) + \sum_{s \in \mathcal{S}} w_s \, f_{\mathrm{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) \\ &= \sum_{s \in \mathcal{S}} w_s \left(f_{\mathrm{I}}(\boldsymbol{x}) + f_{\mathrm{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) \right) \\ &= \sum_{s \in \mathcal{S}} w_s \, f_s(\boldsymbol{x}, \boldsymbol{y}_s), \end{aligned} \tag{f}$$

where the equalities follow from our assumptions on the weights w_s.

Using these definitions, $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$ can be equivalently stated as the following single-stage optimization problem, also known as the *'extensive form'* or the *'deterministic equivalent'*:

$$\begin{aligned} f^{\mathcal{X},\mathcal{Y}} = \min_{\substack{\boldsymbol{x} \in \mathcal{X} \\ \boldsymbol{y} \in \mathcal{Y}}} & f(\boldsymbol{x}, \boldsymbol{y}) \\ \text{s. t. } & \boldsymbol{g}_{\mathrm{DE}}(\boldsymbol{x}, \boldsymbol{y}), \end{aligned} \tag{$\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$}$$

where the vector-valued constraint function $\boldsymbol{g}_{\mathrm{DE}} : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}^{N_g^{\mathrm{DE}}}$, groups all $N_g^{\mathrm{DE}} := N_{\mathrm{I}} + N_s N_{\mathrm{II}}$ constraints in DE, and is defined as

$$\boldsymbol{g}_{\mathrm{DE}}(\boldsymbol{x}, \boldsymbol{y}) = \begin{pmatrix} g_{\mathrm{DE},1}(\boldsymbol{x}, \boldsymbol{y}) \\ \vdots \\ g_{\mathrm{DE},N_g^{\mathrm{DE}}}(\boldsymbol{x}, \boldsymbol{y}) \end{pmatrix} := \begin{pmatrix} g_{\mathrm{I},1}(\boldsymbol{x}) \\ \vdots \\ g_{\mathrm{I},N_{\mathrm{I}}}(\boldsymbol{x}) \\ g_{\mathrm{II},1,1}(\boldsymbol{x}, \boldsymbol{y}_1) \\ \vdots \\ g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\boldsymbol{x}, \boldsymbol{y}_{N_s}) \end{pmatrix}. \tag{$\boldsymbol{g}_{\mathrm{DE}}$}$$

The two problems $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$ and $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ are equivalent in the sense that their globally and locally optimal solution points and optimal objective values coincide if they exist, whereas if one of the formulations is infeasible or unbounded, so is the other, see e.g., [2]. We are interested in the case where all functions in DE may be nonconvex. We limit the theoretical considerations, implementation, and numerical results to continuous variables. Thus, we do not explicitly address issues pertaining to discrete variables in the following. The presence of discrete variables would however not pose substantial complication.

1.2 Literature overview

Solving $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$ by applying general-purpose branch and bound (B&B) solvers [e.g., [4–8]] to $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ is possible, typically amounting to solution of relaxations of $\mathrm{DE}^{\mathcal{X}^n,\mathcal{Y}^n}$ in every B&B node. Here $\mathcal{X}^n \in \mathbb{I}X$ and $\mathcal{Y}^n \in \mathbb{I}Y$, where $\mathbb{I}X$ and $\mathbb{I}Y$ denote the sets of nonempty, compact interval subsets of $\mathcal{X}$ and $\mathcal{Y}$. However, as B&B is intrinsically exponential in the number of (branched) variables, this approach has worst-case exponential runtime in the number of scenarios. This has motivated the development of decomposition algorithms capable of exploiting the special structure of TSP for a more efficient solution. In these algorithms, multiple independent subproblems are solved instead of instances of DE, which can result in a reduction of computational time required for the solution, as the subproblems are generally much smaller and thus cheaper to solve. In the best case, such decomposition algorithms achieve linear scaling with the number of scenarios N_s, i.e., an arithmetic complexity of

$\mathcal{O}(N_s)$. Furthermore, the subproblems are independent and may thus be solved in parallel, resulting in significant additional reductions of wall time.

Historically, decomposition strategies have predominantly been developed for certain subclasses of TSP, e.g., those restricted to linear functions and either only continuous [e.g., 9, 10] or mixed-integer variables [e.g., 11, 12], or those restricted to convex nonlinear functions [e.g., Generalized Benders Decomposition (GBD) [13]]. More recently, algorithms addressing subclasses of TSP allowing for certain nonconvexities, but imposing additional structural assumptions have also been proposed [e.g., [14–18]]. In the most general case, any of the functions in TSP may be nonconvex, and no additional structural assumptions are imposed. Two algorithm variants addressing this case are proposed by [19], however, both variants consider elements of $\boldsymbol{y}$ which introduce nonconvexity as complicating variables in addition to $\boldsymbol{x}$. Thus, in the worst case subproblems have a similar size as the original problem, diminishing the benefits of decomposition.

Three further recent algorithms all employ B&B exclusively on the first-stage variables: (i) [20] propose a modified Lagrangian relaxation in which so called nonancticipativity constraints (cf. Sect. 2) are dualized. The resulting Lagrangian problem is thus still a nonconvex two-stage problem but exhibits additional structure and can thus be solved in a decomposable manner using the algorithm proposed by [14, 15]. As a result, only the continuous first-stage variables need to be branched. (ii) [21] propose another B&B algorithm that obtains lower bounds in each node via global solutions to separate, but generally nonconvex scenario subproblems, resulting from simply dropping the nonanticipativity constraints. (iii) [22] use mixed integer linear or convex mixed integer nonlinear relaxations based on DE as lower bounding problems, which are solved via GBD. Cuts from Lagrangean subproblems are added to a Benders master problem and cutting planes for convexification are added to the Benders subproblems. All three algorithms [20–22], solve N_s independent subproblems on $\mathcal{X}^n \times \mathcal{Y}_s$ at each B&B node n, where $\mathcal{X}^n \in \mathbb{I}X$. While this implies that the computational work of the bounding operation scales linearly in N_s, and further, that the subproblems can be solved in parallel, linear scaling of the overall algorithms with N_s would additionally require that the number of nodes in the outer B&B search is independent of the number of scenarios. Note, however, that within a family of problems with variable number of scenarios, the quality of the lower bounds can be expected to depend on the number of scenarios. For a given tolerance, the number of nodes visited by the outer B&B search may therefore depend on the number of scenarios, despite branching only on $\boldsymbol{x}$. Of course, a similar argument also holds for other algorithms that are commonly thought to scale linearly with the number of scenarios, e.g., classical Lagrangian dualization for convex problems. While such algorithms are typically much more efficient for solving DE compared to general-purpose B&B, and empirically do exhibit linear scaling, they have not been rigorously proven to scale linearly with the number of scenarios in the general case.

1.3 Challenges of existing algorithms

Recently [23] observed that all three algorithms, addressing general nonconvex instances of TSP [20–22] fall into the category of *projection-based decomposition algorithms* (PBDAs). Algorithms in this category directly solve $\text{TSP}^{\mathcal{X},\mathcal{Y}}$ (which can be considered a projection of $\text{DE}^{\mathcal{X},\mathcal{Y}}$ onto the $\mathcal{X}$ space) by considering only the first-stage variables via second-stage optimal value functions $f_{\text{II},s}^{\mathcal{Y}_s}$. [24] argue that this approach likely suffers from the cluster effect, a phenomenon of some spatial B&B algorithms, where a large number of nodes may need to be visited near approximate global minimizers [25–27]. To avoid this effect, the

relaxations of both objective and constraints need to have a sufficiently high convergence order [28]. Note that throughout the article we refer to convergence order in the sense of Hausdorff, unless stated otherwise. The convergence order of relaxations typically used in algorithms for (mixed-integer) nonlinear programs has been analyzed in a series of articles [cf. [28–31]]. [24] show that as a result of performing search in the $\mathcal{X}$ domain only, PBDAs need to construct relaxations of the so-called *scenario value functions*:

$$f_s^{\mathcal{X},\mathcal{Y}_s}(\boldsymbol{x}) := \begin{cases} f_{\mathrm{I}}(\boldsymbol{x}) + f_{\mathrm{II},s}^{\mathcal{Y}_s}(\boldsymbol{x}), & \boldsymbol{x} \in \mathcal{F}_s^{\mathcal{X},\mathcal{Y}_s} \\ +\infty, & \text{otherwise} \end{cases},$$

where $\mathcal{F}_s^{\mathcal{X},\mathcal{Y}_s}$ are the feasible subsets of $\mathcal{X}$ in scenario s:

$$\mathcal{F}_s^{\mathcal{X},\mathcal{Y}_s} := \{\boldsymbol{x} \in \mathcal{X} \mid \boldsymbol{g}_{\mathrm{I}}(\boldsymbol{x}) \leq \boldsymbol{0}, \exists \boldsymbol{y}_s \in \mathcal{Y}_s : \boldsymbol{g}_{\mathrm{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) \leq \boldsymbol{0}\}.$$

Adopting the convention for the minimum of an infeasible problem to be infinite, the weighted sum over the scenario value functions is equivalent to the objective of TSP. [24] demonstrate that only branching on $\boldsymbol{x}$ generally causes $f_s^{\mathcal{X},\mathcal{Y}_s}$ to be nonsmooth, which in turn limits the achievable convergence order. In particular, even the ideal PBDA, which uses the tightest-possible relaxation for each $f_s^{\mathcal{X},\mathcal{Y}_s}$, i.e., the convex envelope, generally has a convergence order below 1, and only achieves first-order convergence if all $f_s^{\mathcal{X},\mathcal{Y}_s}$ are Lipschitz. On the other hand, they show that this ideal relaxation has second-order convergence if $f_s^{\mathcal{X},\mathcal{Y}_s}$ are twice continuously differentiable, and furthermore, that the algorithm of [22] is equivalent to using this ideal relaxation, if optimal dual multipliers $\boldsymbol{\lambda}_s^*$ are used. Note that in general, generating convex envelopes of arbitrary $f_s^{\mathcal{X},\mathcal{Y}_s}$ (via optimal dual multipliers or otherwise) is prohibitively expensive. Furthermore, even for convex f, $\boldsymbol{g}_{\mathrm{I}}$ and $\boldsymbol{g}_{\mathrm{II},s}$, and even in the absence of discrete variables, the $f_s^{\mathcal{X},\mathcal{Y}_s}$ are not guaranteed to be smooth, but rather only lower semi-continuous [cf., e.g., Theorem 35, Chapter 3 of 1]. In summary, using PBDAs, i.e., branching on $\boldsymbol{x}$ only, limits convergence order to below one in general. As a result [24] state that PBDAs are expected to suffer from clustering, and suggest to search for alternative decomposition approaches, rather than for better relaxations in PBDAs. While a higher convergence order can certainly be advantageous, we point out that this conclusion might be overly pessimistic, as the occurrence of clustering is determined by the interplay of both convergence order, and growth order of the objective and constraint functions [also see 28].

Nevertheless, the three aforementioned PBDAs [20–22] may potentially have further issues. First, for each node n with domain $\mathcal{X}^n \in \mathbb{X}$, visited by the outer B&B algorithm searching on $\mathcal{X}$, an inner algorithm searches on $\mathcal{X}^n \times \mathcal{Y}_s$ during the solution of the subproblems. The consideration of $\boldsymbol{x}$ in both levels will therefore result in repeated consideration of the same domain, constituting a duplication of work. Second, in the general case, where there are nonconvexities in the second stage (through nonconvex objectives or constraints, or integer variables), the lower bounding subproblems must at least occasionally be solved globally to guarantee convergence. In addition, [20] and [21] also solve their upper bounding problems globally, while [22] do not explicitly state whether their solutions are local or global. Finally, the nesting of these expensive bounding routines in an outer B&B algorithm, bears resemblance to early ideas for solving general mixed-integer nonlinear programming problems, which considered branching on the integer variables and globally solving a continuous nonconvex problem in each node. However, such ideas have been abandoned since nested exponential approaches are considered computationally unfavorable [32].

1.4 A new decomposition algorithm for TSP

To improve convergence orders of the relaxations, and to avoid duplication of work and the nesting of expensive search routines, we propose an alternative decomposition algorithm for TSP. Similar to solving DE via a classical B&B algorithm, we explicitly branch on first- and second-stage variables, however, we still make use of the structure inherent to TSP to obtain decomposable bounding subproblems for each scenario. We call our proposed algorithm *MUSE-BB*, as it combines classical scenario decomposition with **mu**lti**se**ction [33] in a **B&B** algorithm. Efficient branching on multiple instances of a particular second-stage variable is made possible by the fact that bounding subproblems for each scenario are independent of second-stage variable instances from other scenarios: While branching a node on N_s second-stage variables results in 2^{N_s} child nodes, only $2\,N_s$ independent subproblems need to be solved to update their lower bounds. Each child node can then be generated by combining bounds and variable domains from N_s out of the $2\,N_s$ independent subproblems. To limit memory requirements as well as the number of generated child nodes with poor lower bounds, we filter the N_s candidate bisections based on strong-branching scores, and allow for selecting a further subset of these bisections, ensuring an upper limit on the total number of generated child nodes.

Like classical B&B algorithms, MUSE-BB searches the full variable space. Thus in the worst-case, its runtime is expected to be exponential in N_s. However, the combination of decomposition with multisection allows for a more efficient exploration of the search space than with classical algorithms. Moreover, we analyze the convergence order of the lower bounding scheme used in MUSE-BB. We show that while this convergence order is generally lower than in classical B&B algorithms, it is at least as high as in PBDAs, and can be strictly larger when the scenario value functions $f_s^{\mathcal{X},\mathcal{Y}_s}$ are not Lipschitz. In particular we show that the lower bounding scheme of MUSE-BB is (at least) first-order convergent if all functions and convex relaxations are Lipschitz. While our lower bounding scheme is generally not second-order convergent, we discuss a possible extension of MUSE-BB, whose lower bounding scheme achieves second-order convergence at unconstrained minimizers by dualizing nonanticipativity constraints instead of dropping them. Overall, the results indicate that MUSE-BB and its extension at least partially avoid issues with the cluster effect.

The remainder of the article is structured as follows: Sect. 2 gives a brief review of the decomposable bounding subproblems used in scenario decomposition algorithms for TSP. In Sect. 3 we motivate the use of multisection branching of second-stage variables. Following this, we outline two alternative variants of multisection that allow for efficient incorporation of decomposable bounding problems in a B&B algorithm, branching on both $\boldsymbol{x}$ and $\boldsymbol{y}$. Section 4 presents the MUSE-BB algorithm, incorporating one variant of multisection branching. It includes implementation details followed by a formal statement of the MUSE-BB algorithm and subroutines. In Sect. 5 we present convergence results for both our lower bounding problems, and the overall algorithm. We show that under mild conditions MUSE-BB converges to an ε_f-optimal solution in finite time for any $\varepsilon_f > 0$. Section 6 presents the results of computational experiments on a small test problem, highlighting the effect of different parameters on MUSE-BB, and Sect. 7 summarizes the results and gives an outlook on future work.

2 Decomposable bounding subproblems for TSP

In this section we review how bounds on TSP can be obtained from separate subproblems for each scenario. Since this approach trivially enables both parallelization and linear scaling of the computational work for bounding with N_s, its variants are the basis of many existing decomposition algorithms, as well as for MUSE-BB. The principal idea for decomposable bounding routines is that first-stage variables are complicating, because they appear in the objectives and constraints of all scenarios. Therefore, the problem can be decoupled by scenario, by either introducing independent copies of $\boldsymbol{x}$, or fixing its value. As shown in the following, these two cases result in subproblems which respectively provide lower and upper bounds on the optimal objective value $f^{\mathcal{X},\mathcal{Y}}$ of $\text{TSP}^{\mathcal{X},\mathcal{Y}}$.

An equivalent representation of $\text{DE}^{\mathcal{X},\mathcal{Y}}$ and thus $\text{TSP}^{\mathcal{X},\mathcal{Y}}$ is the lifting obtained by introducing a copy $\boldsymbol{x}_s$ of $\boldsymbol{x}$ for each scenario s and enforcing the equality of these copies, resulting in the following *nonanticipativity problem*.

$$\begin{aligned} f^{\mathcal{X},\mathcal{Y}} = \min_{\substack{\boldsymbol{x}_s \in \mathcal{X} \\ \boldsymbol{y} \in \mathcal{Y}}} & \sum_{s\in\mathcal{S}} w_s f_s(\boldsymbol{x}_s, \boldsymbol{y}_s) \\ \text{s. t.} \quad & \sum_{s\in\mathcal{S}} \boldsymbol{H}_s \boldsymbol{x}_s = \boldsymbol{0} \\ & \boldsymbol{g}_{\text{I}}(\boldsymbol{x}_s) \leq \boldsymbol{0} \quad \forall s \in \mathcal{S} \\ & \boldsymbol{g}_{\text{II},s}(\boldsymbol{x}_s, \boldsymbol{y}_s) \leq \boldsymbol{0} \quad \forall s \in \mathcal{S}. \end{aligned} \tag{$\text{DE}^{\mathcal{X},\mathcal{Y}}_{\text{NAC}}$}$$

In DE_{NAC}, the first set of constraints enforces equality of all $\boldsymbol{x}_s$, thus, the coupling is moved to these so called *nonanticipativity constraints* (NACs), where $\boldsymbol{H}_s$ are appropriately shaped, sparse matrices. For simplicity, we assume the following, specific form of the NACs, also used, e.g., in [22]:

$$\boldsymbol{x}_1 - \boldsymbol{x}_s = \boldsymbol{0} \quad \forall s \in \mathcal{S}\backslash\{1\}. \tag{NACs}$$

Due to the linearity of the NACs, dualizing them with $N_s - 1$ multiplier vectors $\boldsymbol{\pi}_s \in \mathbb{R}^{N_x}$, $s \in \mathcal{S}\backslash\{1\}$ removes the coupling, as it allows to define the vector $\boldsymbol{\lambda} := (\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_{N_s})$, consisting of scenario-specific multiplier subvectors

$$\begin{aligned} \boldsymbol{\lambda}_1 &:= -\sum_{s\in\mathcal{S}\backslash\{1\}} \boldsymbol{\pi}_s / w_s, \\ \boldsymbol{\lambda}_s &:= \boldsymbol{\pi}_s / w_s \quad s \in \mathcal{S}\backslash\{1\}. \end{aligned}$$

Note that inherently,

$$\sum_{s\in\mathcal{S}} w_s \boldsymbol{\lambda}_s = \boldsymbol{0}. \tag{1}$$

The resulting dualization gives rise to the *Lagrangian relaxation*

$$\begin{aligned} f^{\mathcal{X},\mathcal{Y}}_{\text{LR}}(\boldsymbol{\lambda}) := \min_{\substack{\boldsymbol{x}_s \in \mathcal{X} \\ \boldsymbol{y} \in \mathcal{Y}}} & \sum_{s\in\mathcal{S}} w_s \left[f_s(\boldsymbol{x}_s, \boldsymbol{y}_s) + \boldsymbol{\lambda}_s{}^{\intercal} \boldsymbol{x}_s \right] \\ \text{s. t.} \quad & \boldsymbol{g}_{\text{I}}(\boldsymbol{x}_s) \leq \boldsymbol{0} \quad \forall s \in \mathcal{S} \\ & \boldsymbol{g}_{\text{II},s}(\boldsymbol{x}_s, \boldsymbol{y}_s) \leq \boldsymbol{0} \quad \forall s \in \mathcal{S}. \end{aligned} \tag{$\text{LR}^{\mathcal{X},\mathcal{Y}}$}$$

By weak duality, the value $f^{\mathcal{X},\mathcal{Y}}_{\text{LR}}(\boldsymbol{\lambda})$ provides a lower bound to $f^{\mathcal{X},\mathcal{Y}}$ for any $\boldsymbol{\lambda}$ satisfying Eq. (1) [cf. e.g., 34]. Furthermore, this bound can be obtained by solving the N_s separate

Lagrangian subproblems

$$f_{\mathrm{LSP},s}^{\mathcal{X},\mathcal{Y}_s}(\boldsymbol{\lambda}_s) := \min_{\substack{\boldsymbol{x}_s\in\mathcal{X}\\ \boldsymbol{y}_s\in\mathcal{Y}_s}} f_s(\boldsymbol{x}_s,\boldsymbol{y}_s) + \boldsymbol{\lambda}_s{}^{\intercal}\boldsymbol{x}_s \quad \text{s. t. } \boldsymbol{g}_{\mathrm{I}}(\boldsymbol{x}_s) \le \mathbf{0},\ \boldsymbol{g}_{\mathrm{II},s}(\boldsymbol{x}_s,\boldsymbol{y}_s) \le \mathbf{0}, \qquad \mathrm{LSP}_s^{\mathcal{X},\mathcal{Y}_s}$$

and calculating the Lagrangian relaxation based lower bound as

$$f_{\mathrm{LR}}^{\mathcal{X},\mathcal{Y}}(\boldsymbol{\lambda}) := \sum_{s\in\mathcal{S}} w_s\, f_{\mathrm{LSP},s}^{\mathcal{X},\mathcal{Y}_s}(\boldsymbol{\lambda}_s) \le f^{\mathcal{X},\mathcal{Y}}. \tag{LRLB}$$

The best such bound is obtained by solving the Lagrangian dual, which can be written as

$$f_{\mathrm{LR}}^{\mathcal{X},\mathcal{Y}}(\boldsymbol{\lambda}^*) := \max_{\substack{\boldsymbol{\lambda}\in\mathbb{R}^{N_S N_x}\\ \sum_{s\in\mathcal{S}}\boldsymbol{\lambda}_s=\mathbf{0}}} f_{\mathrm{LSP},s}^{\mathcal{X},\mathcal{Y}_s}(\boldsymbol{\lambda}_s). \qquad \mathrm{L}^{\mathcal{X},\mathcal{Y}}$$

It can be shown that if the sets $\mathcal{F}_s^{\mathcal{X},\mathcal{Y}_s}$ have a nonempty intersection, the resulting bound corresponds to the minimum of the weighted sum of convex envelopes of scenario value functions, [24], i.e:

$$f_{\mathrm{LR}}^{\mathcal{X},\mathcal{Y}}(\boldsymbol{\lambda}^*) = \min_{\boldsymbol{x}\in\mathcal{X}} \sum_{s\in\mathcal{S}} w_s \operatorname{conv} f_s^{\mathcal{X},\mathcal{Y}_s}(\boldsymbol{x}).$$

In that sense $f_{\mathrm{LR}}^{\mathcal{X},\mathcal{Y}}(\boldsymbol{\lambda}^*)$ constitutes the best bound obtainable via convex relaxation in the framework of scenario decomposition. Unfortunately, obtaining optimal dual multipliers $\boldsymbol{\lambda}^*$ is both computationally expensive and numerically challenging [35]. We therefore only consider the implications of updating the dual multipliers in Sect. 5, whereas in the remainder of this work, we focus on the simpler case, also considered by [21], where all multipliers are fixed to zero. In that case, the scenario relaxation of $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ consists of N_s *scenario problems* of the form

$$f_{\mathrm{SP},s}^{\mathcal{X},\mathcal{Y}_s} := \min_{\substack{\boldsymbol{x}_s\in\mathcal{X}\\ \boldsymbol{y}_s\in\mathcal{Y}_s}} f_s(\boldsymbol{x}_s,\boldsymbol{y}_s) \quad \text{s. t. } \boldsymbol{g}_{\mathrm{I}}(\boldsymbol{x}_s) \le \mathbf{0},\ \boldsymbol{g}_{\mathrm{II},s}(\boldsymbol{x}_s,\boldsymbol{y}_s) \le \mathbf{0}. \qquad \mathrm{SP}_s^{\mathcal{X},\mathcal{Y}_s}$$

In Sect. 4.1 we will introduce further subproblems, obtained from additional relaxations of SP_s. To distinguish the different optimal objective values, we use corresponding subscripts. The globally optimal objective values $f_{\mathrm{SP},s}^{\mathcal{X},\mathcal{Y}_s}$ of problems $\mathrm{SP}_s^{\mathcal{X},\mathcal{Y}_s}$ can be used to obtain a lower bound $f_{\mathrm{SP}}^{\mathcal{X},\mathcal{Y}}$ on the optimal objective value $f^{\mathcal{X},\mathcal{Y}}$ of $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$, i.e.,

$$f_{\mathrm{SP}}^{\mathcal{X},\mathcal{Y}} := \sum_{s\in\mathcal{S}} w_s\, f_{\mathrm{SP},s}^{\mathcal{X},\mathcal{Y}_s} \le f^{\mathcal{X},\mathcal{Y}}. \tag{SPLB}$$

While the resulting first-stage solutions obtained for each scenario will generally differ from each other, the bound can be made arbitrarily tight by exhaustive branching on $\boldsymbol{x}$. PBDAs like [20–22] use this fact: while they branch on $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ during the global solution of the subproblems SP_s, the outer B&B search only requires branching on $\boldsymbol{x}$ to ensure convergence. As shown by [24], however, the convergence order of such lower bounding schemes is

inherently limited due to the nonsmoothness of $f_{\mathrm{II},s}^{\mathcal{Y}_s}(\boldsymbol{x})$, incurred by projection, also cf. TSP and Sect. 1.

Upper bounds on $f^{\mathcal{X},\mathcal{Y}}$ can generally be obtained by evaluating any feasible point. Fixing $\boldsymbol{x}$ to an arbitrary point $\widetilde{\boldsymbol{x}} \in \mathcal{X}$ that is feasible with respect to $\boldsymbol{g}_{\mathrm{I}}$, gives rise to N_s instances of RP_s. If each of these problems has at least one feasible point $\widetilde{\boldsymbol{y}}_s$, the function values $f_{\mathrm{II},s}(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}}_s)$ provide an upper bound on $f_{\mathrm{II},s}^{\mathcal{Y}_s}(\widetilde{\boldsymbol{x}})$, and thus the upper bounding function $\overline{f}$, defined as

$$\overline{f}(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}}) := f_{\mathrm{I}}(\widetilde{\boldsymbol{x}}) + \sum_{s\in\mathcal{S}} w_s\, f_{\mathrm{II},s}(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}}_s) \geq f^{\mathcal{X},\mathcal{Y}} \tag{UB}$$

provides an upper bound on the optimal objective value $f^{\mathcal{X},\mathcal{Y}}$. Thus, given a candidate for $\widetilde{\boldsymbol{x}}$, values for $\widetilde{\boldsymbol{y}}_s$ can be obtained by local or global solutions of $\mathrm{RP}_s^{\mathcal{Y}_s}(\widetilde{\boldsymbol{x}})$. Candidates proposed in the literature are commonly based on the individual solutions $\boldsymbol{x}_s^*$ from the lower bounding subproblems. A common candidate is the (w_s-weighted) average $\widetilde{\boldsymbol{x}} = \boldsymbol{x}^{\mathrm{avg}} = \sum_{s\in\mathcal{S}} w_s \boldsymbol{x}_s^*$ [20–22]. However, since the feasible set of $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$ is generally nonconvex, this point may be infeasible. An alternative candidate that is at least guaranteed to be feasible with respect to $\boldsymbol{g}_{\mathrm{I}}$, and $\boldsymbol{g}_{\mathrm{II},s^{\mathrm{rep}}}$, is $\widetilde{\boldsymbol{x}} = \boldsymbol{x}_{s^{\mathrm{rep}}}^*$, such that s^{rep} is a *representative* scenario for which $\boldsymbol{x}_{s^{\mathrm{rep}}}^*$ is closest to $\boldsymbol{x}^{\mathrm{avg}}$ with respect to the relative Euclidean distance, i.e,

$$s^{\mathrm{rep}} \in \arg\min_{s\in\mathcal{S}} \sum_{i=1}^{N_x} \left(\frac{x_i^{\mathrm{avg}} - x_{s,i}^*}{\overline{x}_i - \underline{x}_i} \right)^2, \tag{s^{rep}}$$

where $\overline{x}_i$, and $\underline{x}_i$ denote the original lower and upper bounds of x_i [22]. Note that while $\boldsymbol{x}_{s^{\mathrm{rep}}}^*$ is trivially feasible in s^{rep}, it is generally not in other scenarios. Furthermore, if a candidate $\boldsymbol{x}_{s^{\mathrm{rep}}}^*$ does happen to be feasible, there is no guarantee that local solutions to the corresponding instances of RP_s are found.

As with any spatial B&B method, in the general nonconvex case, a guarantee to find a feasible point allowing for termination is only given if the feasible set of $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ has a nonempty interior at a global minimizer, also compare with the analysis for single-stage programs in [36]. Furthermore, Example 3.1 in [36], where upper bounds are obtained simply from feasible lower bounding solutions, shows that even when the interior is nonempty, certain combinations of problem instances, branching and node selection rules can lead to sequences of lower bounding solutions that never include a feasible point. In such cases, an adaption of the tested candidates, such as the approach proposed by [36] may be necessary. Unfortunately such approaches may not address the more general situation of an empty interior, e.g., due to the presence of equality constraints, although there are approaches that produce upper bounds without guaranteeing to find feasible points [37]. On the other hand, feasible, and even (approximately) globally optimal solutions can often be produced relatively easily for many applications. Because of this, we neither implement the methods presented in [36] in MUSE-BB, nor analyze this issue further. Instead we follow the common approach to perform upper bounding via local solutions from candidate points, and concentrate this work on the issues pertaining to lower bounding.

In summary, by solving instances of the separable subproblems $\mathrm{SP}_s^{\mathcal{X},\mathcal{Y}_s}$ and $\mathrm{RP}_s^{\mathcal{Y}_s}(\widetilde{\boldsymbol{x}})$, we can bound the desired optimal solution value of the original problem $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ from below and above:

$$f_{\mathrm{SP}}^{\mathcal{X},\mathcal{Y}} \leq f^{\mathcal{X},\mathcal{Y}} \leq \overline{f}(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}}).$$

Assuming that arbitrarily good feasible points are found during the successive partitioning of the variable domains, the bounds can be tightened until some satisfactory accuracy $\varepsilon_f > 0$

is reached. The upper bound $\overline{f}$ then serves as an ε-optimal solution to problem $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$. In the following section we present a special branching scheme that efficiently combines the decomposable subproblems with partitioning of both $\mathcal{X}$ and $\mathcal{Y}$.

3 Multisection branching for decomposable bounding schemes

To avoid several issues associated with the nested branching of PBDAs (cf. Sect. 1), we propose to combine decomposable bounding schemes with explicit branching of both first- and second-stage variables. As argued below, standard branching of individual variables would eliminate some of the benefits of decomposable bounding schemes. We therefore propose a special branching scheme that either partitions a single first-stage variable or multiple second-stage variable instances in each iteration. To refer to the partition elements containing the lower/upper part of a branched variable domain, we say the respective variable was *branched down/up*.

We first present the concept of multisection as used in the MUSE-BB in Sect. 3.1. Following this, in Sect. 3.2 we outline an alternative idea that may be seen as a hybrid between the variant presented in Sect. 3.1, and existing PBDAs.

3.1 Multisection in MUSE-BB

In a B&B algorithm for TSP using separable lower and upper bounding problems, branching on elements of $\boldsymbol{x}$ and $\boldsymbol{y}$ has different implications for the resulting nodes: each node n is characterized by the domains $\mathcal{X}^n \subset \mathcal{X}$ and $\mathcal{Y}^n \subset \mathcal{Y}$, where $\mathcal{Y}^n := \times_{s\in\mathcal{S}} \mathcal{Y}_s^n$; $\mathcal{Y}_s^n \subset \mathcal{Y}_s$. To obtain a lower bound on n, variants of the N_s subproblems $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$ are solved. While branching on an element of $\boldsymbol{x}$ bisects $\mathcal{X}^n$ into two subdomains, e.g., $\mathcal{X}^d$ and $\mathcal{X}^u$, which generally results in changed bound contributions from all subproblems (compare cases **a** and **b** in Fig. 1), branching on an element of $\boldsymbol{y}$, e.g., $y_{s,i}$, only bisects the second-stage variable domain $\mathcal{Y}_s^n$ of the associated scenario s (compare cases **b** and **c** in Fig. 1). Thus, if we were to only branch on $y_{s,i}$, each of the two resulting child nodes would have $N_s - 1$ unchanged subproblems with respect to n. An example for this situation is given by the case **c** of Fig. 1.

In the parallel setting, where at least two subproblems can be solved simultaneously, this implies that standard branching on second-stage variables leaves some processing capacity unused. In other words, we could only exploit the parallelizable solution of subproblems when processing nodes obtained from branching on first-stage variables.

To enable parallelism when processing nodes produced from second-stage branching, we can branch on all N_s instances of a particular second-stage variable, instead of a single one. Note that such a multisection is equivalent to N_s sequential bisections, i.e., it splits the original node into 2^{N_s} child nodes instead of two, also see the cases **d)** through **g)** of Fig. 1. Multisection has previously been used in different B&B algorithms for general nonlinear problems. Mostly this was in the form of branching the domain of a single variable at multiple points (also called 'multisplitting') [38–40], but there are also examples of using multisection in the present sense, i.e., branching once on multiple variables [33]. While these works showed that multisplitting and multisection can result in better computational performance than bisection, the considered B&B algorithms used standard bounding procedures and thus needed to process all of the resulting nodes individually. In contrast, when solving TSP, the use of separable bounding subproblems such as SP_s^n allows us to generate bounds for the exponential number of nodes resulting from multisection without explicitly processing

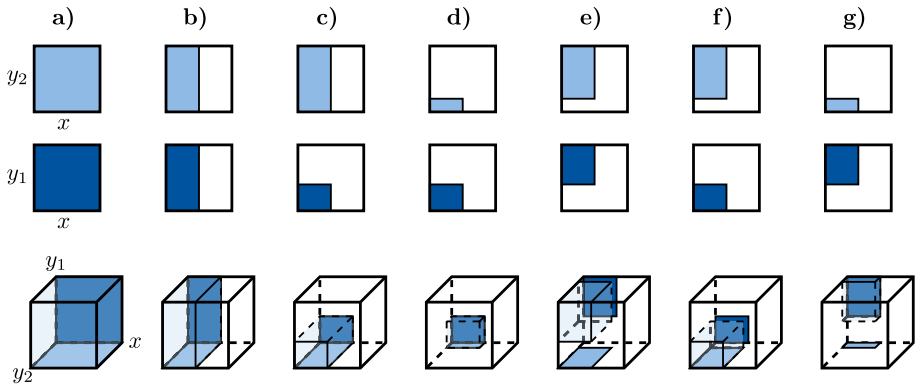


Fig. 1 Implications of branching in scenario decomposition. We consider nodes from solving an instance of DE with $N_x = N_y = 1$, and $N_s = 2$. In this case, each node corresponds to a 3D domain (bottom) and updating the lower bounds requires solving two bounding subproblems on a 2D domain (top). These subproblems can be considered projections on the $\mathcal{X} \times \mathcal{Y}_1$ and $\mathcal{X} \times \mathcal{Y}_2$ faces of the node domain (dark and light blue colors, respectively). **b**: Branching the node from **a** on x affects both subproblem domains. **c**: Branching the node from **b** on a single instance of y, here y_1, only affects the associated subproblem domain, while the subproblem for the second scenario remains unchanged. **d** through **g**: Alternatively to **c**, branching on all instances of y simultaneously results in four nodes instead of two. However, out of the eight subproblems associated with these nodes only four are distinct. When processing two complementary nodes, e.g., node **d** (where both y_1 and y_2 are branched down), and node **e** (where both y_1 and y_2 are branched up), all distinct subproblems are solved. Thus, explicitly processing the remaining nodes, i.e., **f** and **g** in our example, is unnecessary. Instead, bounds for these nodes can be generated by combining the results from the subproblems solved for **d** and **e**. (Color figure online)

each one individually: for each scenario s, branching on the associated second-stage variable instance bisects the domain $\mathcal{Y}_s^n$ into two subdomains, $\mathcal{Y}_s^d$, and $\mathcal{Y}_s^u$. Combining these new domains with the unchanged domain $\mathcal{X}^n$ therefore results in two different subproblems (i.e., $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^d}$, and $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^u}$) per scenario, i.e., multisection of second-stage variables only results in $2\,N_s$ distinct subproblems. Each child node simply corresponds to one of the possible combinations of selecting one of the two subproblems for each scenario. This means that to update lower bounds on all 2^{N_s} child nodes, only the $2\,N_s$ distinct subproblems need to be solved. Note that this can be achieved by processing any two of the 2^{N_s} nodes that contain complementary subproblems. One such choice consists of the pair of nodes resulting from branching all instances of the selected second-stage variable down, or up. In the following we respectively call these two nodes the *lower and upper sibling nodes*.

In summary, if a first-stage variable is selected for branching, we perform standard bisection resulting in two child nodes, whereas if a second-stage variable is selected, we instead perform multisection branching of all associated variable instances for different scenarios, resulting in 2^{N_s} nodes. In both cases, only two nodes need to be processed after branching a given node n: after first-stage branching, these two nodes are simply the child nodes with domains $(\mathcal{X}^d, \mathcal{Y}^n)$ and $(\mathcal{X}^u, \mathcal{Y}^n)$. After second-stage branching, we process the sibling nodes, with domains $(\mathcal{X}^n, \mathcal{Y}^d)$ and $(\mathcal{X}^n, \mathcal{Y}^u)$, where $\mathcal{Y}^d := \bigtimes_{s\in\mathcal{S}} \mathcal{Y}_s^d$ and $\mathcal{Y}^u := \bigtimes_{s\in\mathcal{S}} \mathcal{Y}_s^u$. While theoretically one could generate 2^{N_s} nodes after each second-stage branching, this poses several issues in an actual implementation. To address this, it is possible to filter the candidate bisections contributing to the final multisection, i.e., to keep only a "promising" subset and thus produce a small number of high quality nodes. The process we use for this will be presented in Sect. 4.4.

We point out that while the proposed multisection procedure may appear to avoid a computational cost for node processing that is exponential in N_s, this is only true at the level of an individual iteration. Whereas the cost for generating the child nodes from multisection branching is indeed linear in N_s, their number, and thus the overall computational cost for further processing is still exponential in N_s. In the following, we outline an alternative use of our multisection idea that may avoid this exponential scaling but bears resemblance to PBDAs. While we do not pursue this idea further in the present work, we believe it to be fruitful for future research.

3.2 Projected multisection

The only conceivable path to avoid exponential scaling with N_s in the context of multisection-based second-stage branching is to avoid the explicit generation of the resulting child nodes. One approach for this is to maintain information related to second-stage variable domains and objective bounds at the subproblem level, without combining this information from different scenarios into individual B&B nodes. One can still compute a lower bound related to the first-stage domain by combining the lowest lower bounds from each scenario. Furthermore this bound can be refined by further partitioning of the resulting subproblem domains and organizing the resulting subproblem nodes in a separate B&B tree for each scenario. Similar to PBDAs, this results in 'nested' B&B trees: The outer tree contains nodes based on first-stage domains, each of which maintains a state of progress for the search in the second-stage domains and associated lower bounds in N_s separate second-stage trees.

In contrast to existing PBDAs, branching of first- and second-stage variables is carried out exclusively in the outer and inner trees, respectively. This not only avoids duplication of first-stage variables, but also the exhaustive exploration of the second-stage trees in each iteration, since their state is passed on to child nodes from branching on first-stage variables in the outer tree.

The number of nodes in the outer tree is exponential in N_x. In the worst case, each such outer node is associated to N_s full second-stage trees, the size of which is exponential in N_y. Thus a total of $N_s a^{N_x} b^{N_y}$ nodes may need to be processed, i.e., it appears that this approach would avoid the exponential scaling with N_s and indeed scale linearly with N_s.

While this linear scaling appears promising, avoiding the generation of nodes from explicit combinations of subproblem data results in lower bounds based on the lowest lower bounds from the second-stage trees. Since this may be seen as a 'projection' of bounding information, it is unclear whether this approach can be considered a full-space or rather a projected search. We suspect the convergence order of this approach to be limited as with PBDAs, and therefore focus the remainder of this work on the previously presented idea.

4 Proposed algorithm

We now present the spatial multisection B&B algorithm MUSE-BB for the solution of $\text{TSP}^{\mathcal{X},\mathcal{Y}}$. Algorithm 1 presents a formal statement of MUSE-BB; the relevant subroutines will be presented in the following. For conciseness, we assume throughout this section that given a node n, we have access to its domains $\mathcal{X}^n$ and $\mathcal{Y}^n$, and lower bound $\underline{f}^n$, as well as the domains $\mathcal{X}_s^n$ and $\mathcal{Y}_s^n$, and lower bounds $\underline{f}_s^n$ of the corresponding subproblems. Under this assumption, it suffices to provide nodes to the subroutines instead of all associated data. If a node n can be fathomed by infeasibility, we set its lower bound $\underline{f}^n$ to ∞.

Algorithm 1: MUSE-BB

Input : Instance of TSP$^{\mathcal{X},\mathcal{Y}}$, tolerance ε_f, maximum number of effective bisections $k_{\max}$, strong-branching threshold η
Output: Incumbent point $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger)$, incumbent objective value $\overline{f}$, certificate $\underline{f}$

1 $n \leftarrow \mathcal{X} \times \mathcal{Y}$; $\underline{f}^n \leftarrow -\infty$; $\mathcal{N} \leftarrow \{n\}$; $\overline{f} \leftarrow \infty$; $\mathcal{M}_{\text{sib}} \leftarrow$ empty Map;
2 **while** $\mathcal{N} \neq \emptyset$ **do** `// there are nodes to be processed`
3 $n \leftarrow$ select a node and remove from $\mathcal{N}$;
4 **if** $n \in \mathcal{M}_{\text{sib}}$ **then** `// do a "sibling iteration"`
 `// n is the previously processed parent node, re-entered into` $\mathcal{N}$ `as a placeholder for the sibling nodes`
5 $(d, u) \leftarrow \mathcal{M}_{\text{sib}}[n]$; `// recover the sibling nodes to be processed`
 `// see Subroutine 3 in Section 4.3`
6 $(\underline{f}^n, \overline{f}^n, \boldsymbol{x}^n, \boldsymbol{y}^n) \leftarrow$ `processSiblings(`n, d, u`)`;
7 **if** $\overline{f}^n < \overline{f}$ **then** $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger, \overline{f}) \leftarrow (\boldsymbol{x}^n, \boldsymbol{y}^n, \overline{f}^n)$;
8 **if** $\underline{f}^n < \overline{f}$ **then**
 `// see Subroutine 4 in Section 4.4`
9 $(\mathcal{M}, \mathcal{L}) \leftarrow$ `filteredMultiSection(n, d, u)`;
10 **foreach** $i \in \{0, \ldots, 2^{|\mathcal{L}|} - 1\}$ **do**
 `// see Subroutine 5 in Section 4.4`
11 $o \leftarrow$ `generateOrthantNode(`$i, n, d, u, \mathcal{M}, \mathcal{L}$`)`;
 `// see Subroutine 1 in Section 4.3`
12 **if** $\underline{f}^o < \overline{f}$ **then** `branchNode(`o`)`;
13 **end**
14 **end**
15 **else** `// do a "normal iteration"`
 `// see Subroutine 2 in Section 4.3`
16 $(\underline{f}^n, \overline{f}^n, \boldsymbol{x}^n, \boldsymbol{y}^n) \leftarrow$ `processNode(`n`)`;
17 **if** $\overline{f}^n < \overline{f}$ **then** $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger, \overline{f}) \leftarrow (\boldsymbol{x}^n, \boldsymbol{y}^n, \overline{f}^n)$;
 `// see Subroutine 1 in Section 4.3`
18 **if** $\underline{f}^n < \overline{f}$ **then** `branchNode(`n`)`;
19 **end**
20 $\underline{f} \leftarrow \min_{n \in \mathcal{N}} \underline{f}^n$; `// update lowest lower bound`
21 **if** $\underline{f} + \varepsilon_f > \overline{f}$ **then return** $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger, \overline{f}, \underline{f})$;
22 **end**
23 **return** $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger, \overline{f}, \underline{f})$;

On a high level, MUSE-BB only differs from a standard B&B algorithm in the use of different kinds of iterations for nodes obtained from branching on first- and second-stage variables. In both cases, the bounds of unprocessed nodes are updated, and the nodes are either fathomed (by infeasibility or value dominance, i.e., $\underline{f}^n \geq \overline{f}$) or branched.

Each iteration begins with the selection of a node n from a list of nodes $\mathcal{N}$ (Line 3 in Algorithm 1). The selected node is either an unprocessed node (the root node or one of the two child nodes obtained from standard bisection of a first-stage variable) which is processed via a *"normal iteration"*, similar as in a standard B&B algorithm (Lines 15–18), or it is a placeholder for two unprocessed sibling nodes that will be addressed via a special *"sibling iteration"* (Lines 4–14). In the latter case, n is the parent of the sibling nodes that was processed and multisected, (i.e., branched on N_s second-stage variables as presented in Sect. 3) in a previous iteration. In that case, there is an entry in $\mathcal{M}_{\text{sib}}$, mapping n to the sibling nodes d and u (Lines 4 and 5), which are processed together, using some of the bound and domain information from their parent, n (Line 6). During this step we also search for

an upper bound within the domain of the parent node. If possible, we use such a bound to update the best known upper bound (Line 7) and if this does not allow the parent node to be fathomed (Line 8), we use the results from the sibling iteration to generate processed child nodes whose number is exponential in the number of branched variables. However, instead of using all N_s bisections of the original multisection, we filter them, selecting only a subset for the final multisection (Line 9, also see Sect. 4.4). We only consider branching via partitioning of the original domain through hyperplanes, orthogonal to the branched variable dimensions. Because of this and the related concept of an orthant, i.e., the intersection of k mutually orthogonal half-spaces in k-dimensional Euclidean space, we refer to the nodes resulting from the filtered multisection as *"orthant nodes"* in the following. The map $\mathcal{M}$ and the list $\mathcal{L}$, returned from the filtered multisection, determine the subproblem data (from n, d, or u) to be used for a particular orthant node o. The number k of orthant nodes to be generated is determined by the length of $\mathcal{L}$ (Line 10). As the orthant nodes are already in a processed state, we immediately branch them, provided they cannot be fathomed (Line 12).

In the course of the algorithm, unprocessed nodes are either fathomed or branched, until the lower and upper bounds converge to ε_f optimality (Line 21) or the list $\mathcal{N}$ is exhausted (Line 23, possibly indicating infeasibility of $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$). On termination, MUSE-BB either provides an incumbent $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger)$, with an associated objective value $\overline{f} = f(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger)$ that is at most ε_f larger than the global lower bound $\underline{f}$, or a certificate of infeasibility ($\underline{f} = \infty$).

We implement MUSE-BB as an extension of our deterministic global optimization solver and open-source project MAiNGO [6]. In Sects. 4.1–4.2 we detail how lower bounding and range reduction schemes available in MAiNGO are adapted to subproblems from Sect. 2 to obtain the decomposable bounding schemes used in the processing subroutines. Since node processing comprises the main computational work, the respective subroutines are parallelized in our implementation. The main theoretical results we present in Sect. 5 do not depend on the presented bounding schemes, i.e., alternative ones may be employed analogously. Next, we discuss the branching of first- and second-stage variables (Subroutine 1) via standard bisection and the multisection from Sect. 3, and detail how the resulting nodes are respectively processed in "normal" and "sibling iterations" (Subroutines 2 and 3) in Sect. 4.3. Finally, we present the subroutines for the filtered multisection and orthant node generation in Sect. 4.4.

4.1 Lower and upper bounding

Our deterministic global solver MAiNGO [6] employs a general-purpose B&B algorithm with lower bounding problems obtained via McCormick-based relaxation techniques [30, 41–45]. When solving TSP via equivalence to DE, we generate and solve such relaxations based on $\mathrm{DE}^{\mathcal{X}^n,\mathcal{Y}^n}$ for each node n. In the following, we abbreviate $\mathrm{DE}^{\mathcal{X}^n,\mathcal{Y}^n}$ as DE^n.

PBDAs like [21], on the other hand, only branch on the first-stage variables and solve N_s subproblems $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$ (or variants thereof) in each node. To ensure convergence, the three reviewed algorithms [20–22] at least occasionally solve these subproblems to global optimality. This generally also requires branching on $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$, albeit not in the outer algorithm.

In MUSE-BB we also generate lower bounds based on SP_s, however, we partition both the $\mathcal{X}$ and $\mathcal{Y}$ domains in the same B&B tree and thus consider subproblems based on $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$ (abbreviated as SP_s^n in the following) instead of $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$. In contrast to PBDAs, the explicit partitioning of the $\mathcal{Y}$ domain, renders global solution of subproblems unnecessary for convergence. We therefore further relax the subproblems SP_s^n, resulting in cheaper lower bounding

problems. In particular, we make use of the available relaxation techniques in MAiNGO to construct the following McCormick based convex relaxations of SP_s^n:

$$\begin{aligned} f_{\text{MC},s}^n := \min_{\substack{\boldsymbol{x}_s \in \mathcal{X}^n \\ \boldsymbol{y}_s \in \mathcal{Y}_s^n}} & f_s^{\text{cv},n}(\boldsymbol{x}_s, \boldsymbol{y}_s) \\ \text{s. t. } & \boldsymbol{g}_{\text{I}}^{\text{cv},n}(\boldsymbol{x}_s) \leq \boldsymbol{0} \\ & \boldsymbol{g}_{\text{II},s}^{\text{cv},n}(\boldsymbol{x}_s, \boldsymbol{y}_s) \leq \boldsymbol{0}, \end{aligned} \qquad \text{MC}_s^n$$

where $f_s^{\text{cv},n}$, $\boldsymbol{g}_{\text{I}}^{\text{cv},n}$, and $\boldsymbol{g}_{\text{II},s}^{\text{cv},n}$ are the McCormick based convex relaxations of the functions f_s, $\boldsymbol{g}_{\text{I}}$, and $\boldsymbol{g}_{\text{II},s}$, on $\mathcal{X}^n \times \mathcal{Y}_s^n$, respectively [41, 42, 45]. These problems are further linearized based on subtangents at one or more linearization points [cf. 46]. By default (and in all experiments in Sect. 6) we only linearize at the midpoint of the node domain. The resulting lower bounding problems take the form:

$$\begin{aligned} f_{\text{LP},s}^n := \min_{\substack{\boldsymbol{x}_s \in \mathcal{X}^n \\ \boldsymbol{y}_s \in \mathcal{Y}_s^n \\ v \in \mathbb{R}}} & v \\ \text{s. t. } & \text{sub}_{f_s}^n\left(\boldsymbol{x}_s, \boldsymbol{y}_s\right) \leq v \\ & \text{sub}_{\boldsymbol{g}_{\text{I}}}^n\left(\boldsymbol{x}_s\right) \leq \boldsymbol{0} \\ & \text{sub}_{\boldsymbol{g}_{\text{II},s}}^n\left(\boldsymbol{x}_s, \boldsymbol{y}_s\right) \leq \boldsymbol{0} \end{aligned} \qquad \text{LP}_s^n$$

Here, sub_ϕ^n are subtangents of the convex relaxation of the function ϕ at the center of the domain of node n, i.e.,

$$\text{sub}_\phi^n(\bullet) := \phi^{\text{cv},n}(\boldsymbol{m}_\bullet) + \check{\nabla}\phi^{\text{cv},n}(\boldsymbol{m}_\bullet)^\intercal(\bullet - \boldsymbol{m}_\bullet) \qquad \text{(subtangent)}$$

where the superscript 'cv,n' denotes the corresponding convex relaxation, $\boldsymbol{m}_\bullet$ denotes the midpoint of either $\mathcal{X}^n$ or $\mathcal{X}^n \times \mathcal{Y}_s^n$ (depending on the passed variables), and $\check{\nabla}$ denotes a subgradient, i.e., $\check{\nabla}\phi^{\text{cv},n}(\boldsymbol{m}_\bullet) \in \partial\phi^{\text{cv},n}(\boldsymbol{m}_\bullet)$, where $\partial\phi^{\text{cv},n}(\boldsymbol{m}_\bullet)$ is the subdifferential of $\phi^{\text{cv},n}$ at $\boldsymbol{m}_\bullet$. Since $f_{\text{LP},s}^n$ are valid lower bounds on the globally optimal objective values $f_{\text{SP},s}^{\mathcal{X}^n,\mathcal{Y}_s^n}$ of SP_s^n, they provide a valid lower bound for node n, i.e:

$$f_{\text{LP}}^n := \sum_{s \in \mathcal{S}} w_s \, f_{\text{LP},s}^n \leq f_{\text{SP}}^n \leq f^{\mathcal{X}^n,\mathcal{Y}^n}. \qquad \text{(LPLB}^n\text{)}$$

Evidently this bound is generally weaker than the one obtained via global solution (see SPLB), but it is also much cheaper to compute.

For upper bounding, we solve instances of the form $\text{RP}_s^{\mathcal{Y}_s^n}(\widetilde{\boldsymbol{x}}^n)$ (abbreviated as RP_s^n in the following), instead of $\text{RP}_s^{\mathcal{Y}_s}(\widetilde{\boldsymbol{x}}^n)$ as in PBDAs. Furthermore, in contrast to [20] and [21] who solve their upper bounding problems globally, we again aim to reduce computational cost by solving RP_s^n locally. We obtain $\widetilde{\boldsymbol{x}}^n$ from the lower bounding solution corresponding to s^{rep} (see Sect. 2). If the corresponding local solutions of RP_s^n result in a feasible $\widetilde{\boldsymbol{y}}^n = (\widetilde{\boldsymbol{y}}_1, \ldots, \widetilde{\boldsymbol{y}}_{N_s})$, the corresponding objective values $f_{\text{II},s}\left(\widetilde{\boldsymbol{x}}^n, \widetilde{\boldsymbol{y}}_s^n\right) \geq f_{\text{II},s}^{\mathcal{Y}_s^n}(\widetilde{\boldsymbol{x}}^n)$ can be aggregated to a globally valid upper bound $\overline{f}^n$, via the upper bounding function (UB), i.e:

$$\overline{f}^n := \overline{f}(\widetilde{\boldsymbol{x}}^n, \widetilde{\boldsymbol{y}}^n) \geq f^{\mathcal{X},\mathcal{Y}} \qquad \text{(UB}^n\text{)}$$

If $\overline{f}^n$ is smaller than the previously best upper bound $\overline{f}$, the incumbent $(\boldsymbol{x}^\dagger, \boldsymbol{y}^\dagger)$ and $\overline{f}$ are updated with $(\widetilde{\boldsymbol{x}}^n, \widetilde{\boldsymbol{y}}^n)$ and $\overline{f}^n$, respectively.

4.2 Range reduction

In this section we discuss decomposable range reduction routines for tightening variable bounds in B&B algorithms for TSP. We first consider two general points, namely, how the NACs can enable fathoming by infeasibility after application of these routines, and how dominance rules give rise to scenario-specific objective cuts. We then present the specific routines employed in MUSE-BB. While range reduction is not necessary from a theoretical standpoint, it can improve the efficiency of the algorithm by reducing the search domain.

Based on decomposable bounding problems such as SP_s^n or LP_s^n, one can obtain decomposable range reduction routines by applying reduction techniques to the subproblems instead of the full problem DE^n. Standard techniques for feasibility-based reductions can be applied. As will be discussed in the following, however, optimality-based reductions require modified upper bounds instead of objective values of points, feasible in the subproblems. In both cases, the independence of subproblems allows for parallel updates of the bounds for the variables $(\boldsymbol{x}_s, \boldsymbol{y}_s)$ of each scenario s.

After each round of range reduction, the NACs can be used to tighten the first-stage variable bounds. More explicitly, if $\mathcal{X}_s^n$ denotes the tightened first-stage variable domain for node n and scenario s after any of the presented decomposable range reduction routines, a valid reduction of the overall domain $\mathcal{X}^n$ is evidently given by the intersection $\mathcal{X}^{n\prime}$:

$$\mathcal{X}^{n\prime} := \bigcap_{s\in\mathcal{S}} \mathcal{X}_s^n \qquad (\mathcal{X}_s^n \text{ aggregation})$$

In particular, if $\mathcal{X}^{n\prime}$ is empty, node n can be fathomed by infeasibility.

If an upper bound $\overline{f}$ is known, dominance rules can be used to derive objective cuts for range reduction routines. Since in a decomposable bounding scheme objective values obtainable in any particular node n are limited from below by the local lower bound f_{SP}^n, all nodes for which $f_{\mathrm{SP}}^n > \overline{f}$ holds can be *fathomed by dominance*. To derive a scenario-specific cutoff based on a given value of $\overline{f}$, we rewrite the dominance condition in terms of scenario-specific lower bounds. Using (SPLB), a node is dominated if

$$\sum_{s\in\mathcal{S}} w_s \, f_{\mathrm{SP},s}^{\mathcal{X}^n,\mathcal{Y}_s^n} > \overline{f}.$$

Note that replacing any $f_{\mathrm{SP},s}^{\mathcal{X}^n,\mathcal{Y}_s^n}$ by a smaller value (say $\underline{f}_{\mathrm{SP},s}^n$) results in an even stronger condition, that implies the above. Thus for any particular scenario s, the node is dominated if

$$\underline{f}_{\mathrm{SP},s}^n > \frac{\overline{f} - \sum_{s'\in\mathcal{S}\setminus\{s\}} w_{s'} \, \underline{f}_{\mathrm{SP},s'}^n}{w_s} =: \overline{f}_s^n \qquad (s\text{-domination})$$

The above approach is a slight generalization to the scenario-specific upper bounds proposed by [47] for the 'primal problems' in their decomposition method. In particular, any valid lower bounds $\underline{f}_{\mathrm{SP},s}^n$ can be used. In MUSE-BB we use the maximum of $f_{\mathrm{LP},s}^n$ and an interval arithmetic based lower bound based on the objective of SP_s^n.

MAiNGO implements three range reduction techniques: constraint propagation [CP, cf. e.g. 48], optimization-based bounds tightening [OBBT, cf. e.g. 49], duality-based bounds tightening and probing [both referred to as DBBT in the following, cf. e.g. 50].

CP essentially refers to the inverse propagation of feasible intervals of the constraint values, i.e., $(-\infty, 0]$ in our case, to the variables [48]. This allows to determine conservative

variable ranges for which the constraints can be fulfilled and thus enables domain reduction by intersecting the variable domains with these valid ranges. Thus, applying CP to the subproblems SP_s^n instead of DE^n directly gives a decomposable routine.

The OBBT procedure consists of minimizing or maximizing a selected variable v subject to the (relaxed) constraints of the original problem [49]. In our case, we consider scenario-specific OBBT-problems, based on the lower bounding subproblems LP_s^n, i.e., they take the form:

$$\begin{aligned} \underline{v} \backslash \overline{v} = \min \backslash \max_{\substack{\boldsymbol{x}_s \in \mathcal{X}^n \\ \boldsymbol{y}_s \in \mathcal{Y}_s^n}} \; & v \\ \text{s. t.} \quad & \mathrm{sub}_{f_s}^n\left(\boldsymbol{x}_s, \boldsymbol{y}_s\right) \leq \overline{f}_s^n \\ & \mathrm{sub}_{\boldsymbol{g}_\mathrm{I}}^n\left(\boldsymbol{x}_s\right) \leq \mathbf{0} \\ & \mathrm{sub}_{\boldsymbol{g}_{\mathrm{II},s}}^n\left(\boldsymbol{x}_s, \boldsymbol{y}_s\right) \leq \mathbf{0} \end{aligned} \qquad \mathrm{OBBT}_{s,v}^n$$

While no finite upper bound $\overline{f}$ is known, the first constraint is dropped. For each iteration, we initially consider all variables for OBBT, and apply a variant of the *trivial filtering heuristic* from [49] after each pass. Similar OBBT based problems have been proposed, e.g., by [20, 21, 51] for their respective algorithms.

DBBT uses objective bounds and duality information from the node subproblems that are typically solved in spatial B&B algorithms [50] to tighten variable domains. In our case, if all subproblems LP_s^n are feasible, the solutions $(\widetilde{\boldsymbol{x}}_s, \widetilde{\boldsymbol{y}}_s)$, associated reduced cost multipliers $(\boldsymbol{r}_{x,s}, \boldsymbol{r}_{y,s})$, and lower bounds $f_{\mathrm{LP},s}^n$ are available. If in addition a finite upper bound $\overline{f}$ is known, we can compute scenario-specific $\overline{f}_s^n$ values from s-domination and perform DBBT. For variables v for which the solution value v^* corresponds to the respective lower or upper bound, the complementary bound may be tightened:

$$\begin{aligned} &\textbf{if } v^* = \underline{v}, \ \textbf{set } \overline{v} = \min\left(\overline{v}, \underline{v} + \frac{\overline{f}_s^n - f_{\mathrm{LP},s}^n}{r}\right) \\ &\textbf{if } v^* = \overline{v}, \ \textbf{set } \underline{v} = \max\left(\underline{v}, \overline{v} + \frac{\overline{f}_s^n - f_{\mathrm{LP},s}^n}{r}\right) \end{aligned} \qquad \text{(DBBT)}$$

where r is the corresponding entry in $\boldsymbol{r}_{x,s}$ or $\boldsymbol{r}_{y,s}$, which must be positive in the first case and negative in the second one. For variables for which the solution lies between the bounds, two probing variants of LP_s^n can be solved: in these probing LPs, the variable is temporarily fixed to one of its bounds and DBBT is applied based on the new reduced cost multipliers and optimal objective values. As probing is relatively expensive, it is deactivated by default (and in all experiments of Sect. 6).

Since each subproblem contains only part of the information of DE^n, the presented range-reduction routines will generally be less effective than their full space counterparts. Thus, the use of parallelized range reduction needs to result in sufficiently large reductions of wall time to warrant the looser variable bounds. In comparison to the solution time of a lower bounding problem, CP is computationally very cheap, which makes its decomposable variant less appealing. Nevertheless it must be used when processing sibling nodes obtained from multisection branching (cf. Sect. 3), as the resulting domains are needed for the generation of orthant nodes, also see Subroutine 5 in Sect. 4. OBBT on the other hand is a relatively expensive procedure. This typically causes OBBT to dominate the computational work done per iteration and thus makes a decomposable OBBT variant more appealing. Finally, the

Subroutine 1: `branchNode(`n`)`

```
1  v ← select a variable from (x, y) maximizing largest relative domain width × branching priority;
2  if v ∈ {x_i, | i ∈ {1, ··· , N_x}} then // v corresponds to some x_i
3  |  (d, u) ← bisect n along the domain of v;
4  |  N ← N ∪ {d, u};
5  else // v corresponds to y_{s',i} for some s'
6  |  i ← index for which v = y_{s',i};
7  |  d ← n; u ← n; // Initialize d and u as copies of n
   |  foreach s ∈ S do // branch d/u down/up on all instances of v
8  |  |  v ← y_{s,i};
9  |  |  d ← lower half of bisecting d along the domain of v;
10 |  |  u ← upper half of bisecting u along the domain of v;
11 |  end
   |  // Since n, d, and u all share the same lower bound, the former is
   |     re-entered into N and the latter two are stored in M_sib
12 |  N ← N ∪ {n};
13 |  M_sib[n] ← (d, u);
14 end
```

use of decomposable lower bounding problems inherently requires the use of decomposable DBBT, as duality information necessary for a full space variant is not available.

4.3 Branching and node processing

In this section, we present the branching and processing routines of Algorithm 1. In Subroutine 1, we first present how processed nodes are branched, as this determines the kind of iteration that will be performed for the child nodes. Following this, we present the processing of nodes obtained from first- and second-stage branching in Subroutines 2 and 3.

Any processed node n that is not fathomed is branched on either a first-stage variable or on multiple second-stage variable instances, as outlined in Subroutine 1. For this, we select some first- or second-stage variable v, maximizing the product of *relative domain width* (i.e., current over original interval width) and branching priority (assumed to be nonzero), to ensure exhaustive partitioning. If v is an element of $\boldsymbol{x}$, i.e., $v = x_i$, we bisect the associated domain $\mathcal{X}_i^n = [\underline{x}_i, \overline{x}_i]$ at some branching point x_i^{b}, and add the two resulting nodes with the lower and upper part of the original domain (i.e., $[\underline{x}_i, x_i^{\mathrm{b}}]$ and $[x_i^{\mathrm{b}}, \overline{x}_i]$) to the list of open nodes (Lines 3 and 4 in Subroutine 1). In MUSE-BB, x_i^{b} always corresponds to the center of the interval, i.e., $0.5\,(\underline{x}_i + \overline{x}_i)$.

If instead, v is an element of $\boldsymbol{y}$, i.e., $v = y_{s',i}$, for some s', we perform the proposed multisection branching. As pointed out in Sect. 3, the child nodes of this multisection can subsequently be generated from the results of two complementary nodes. Therefore we only need to generate the lower and upper sibling node at this point. Taking the example from Fig. 1: multisecting a parent node p, corresponding to node **b)** in Fig. 1, results in sibling nodes d, and u, corresponding to nodes **d)**, and **e)**, respectively, which we create by branching all N_s instances of the selected second-stage variable $y_{s,i}$ down / up (Lines 6–11).

For a practical algorithm, we need to limit the number of nodes that will be generated in the sibling iterations, as will be outlined in Sect. 4.4. This is done via a filtered multisection which requires domain and bound data from the parent node n as well as the sibling nodes. We therefore return the parent node to the list of open nodes and create the mapping $n \mapsto (d, u)$

Subroutine 2: `processNode(n)`

1 do CP based on DE^n; fathom by dominance or infeasibility;
2 do OBBT based on LP_s^n; fathom by dominance; apply $\mathcal{X}_s^n$ aggregation, fathom by infeasibility ;
3 solve LP_s^n and set $\underline{f}_s^n \leftarrow f_{\text{LP},s}^n \ \forall s \in \mathcal{S}$, use LPLB^n, set $\underline{f}^n \leftarrow f_{\text{LP}}^n$, fathom by dominance or infeasibility ;
4 $\widetilde{\boldsymbol{x}}^n \leftarrow$ solution of LP_s^n with $s = s^{\text{rep}}$;
5 $(\widetilde{\boldsymbol{y}}_s^n, \overline{f}_s^n) \leftarrow$ solution and objective value of $\text{RP}_s^n \ \forall s \in \mathcal{S}$, update $(\widetilde{\boldsymbol{y}}^n, \overline{f}^n)$ via UB^n, fathom by dominance ;
6 do DBBT based on LP_s^n, fathom by dominance;
7 **return** $(\underline{f}^n, \overline{f}^n, \widetilde{\boldsymbol{x}}^n, \widetilde{\boldsymbol{y}}^n)$

in $\mathcal{M}_{\text{sib}}$ (Lines 12 and 13). When the node n is selected a second time in Algorithm 1, this is detected via a lookup in $\mathcal{M}_{\text{sib}}$ and we perform a sibling iteration instead.

For the root node and all nodes resulting from first-stage branching, we do a "normal iteration", i.e., the respective node is processed as specified in Subroutine 2, and either fathomed, or branched as specified in Subroutine 1. The only difference of Subroutine 2 with respect to a standard B&B algorithm is the possible use of decomposable bounding and range reduction routines from Sect. 4.1 and 4.2. In our implementation, we solve scenario subproblems for OBBT (Line 2 of Subroutine 2), lower and upper bounding (Lines 3 and 5), and DBBT (Line 6) in parallel, while the computationally cheap CP (Line 1) is done using the full problem, DE^n. To generate a candidate solution $\widetilde{\boldsymbol{x}}^n$ for upper bounding (Line 4), we use a representative scenario s^{rep} as outlined in Sect. 2.

With sibling nodes, obtained from second-stage branching, we do a "sibling iteration". Before we give the formal statement of the combined processing of siblings in Subroutine 3, we recall that child nodes from multisection can be generated by combining the results from different subproblems of both siblings (cf. Sect. 3). In contrast to Subroutine 2, the use of decomposable range reduction and bounding routines is thus mandatory in Subroutine 3. Moreover, we cannot perform $\mathcal{X}_s^n$ aggregation after doing range reduction routines on $n \in \{d, u\}$, because the resulting tightening would only be valid for the respective sibling node. However, we can first propagate results form range reduction of both siblings to the parent node p, whose multi section resulted in d and u, and then back to the siblings: let $\mathcal{X}_s^d$, $\mathcal{X}_s^u$ and $\mathcal{Y}_s^d$, $\mathcal{Y}_s^u$ denote the tightened variable domains obtained after applying some range reduction to the subproblems of d and u for scenario s. Then the unions of the first- and second-stage domains are a valid tightening of the corresponding domains from the parent node p, i.e:

$$\begin{aligned} \mathcal{X}_s^p &\leftarrow \text{conv}\left(\mathcal{X}_s^d \cup \mathcal{X}_s^u\right) \\ \mathcal{Y}_s^p &\leftarrow \text{conv}\left(\mathcal{Y}_s^d \cup \mathcal{Y}_s^u\right) \end{aligned} \qquad \text{(parent } s\text{-domain tightening)}$$

Here, the use of the convex hull of the unions is purely for ease of implementation, as it ensures the resulting domains are representable as hyperrectangles. Once we have applied parent s-domain tightening for all scenarios, we can use the resulting $\mathcal{X}_s^p$ for $\mathcal{X}_s^p$ aggregation. Intersecting the resulting $\mathcal{X}^{p\prime}$ with $\mathcal{X}_s^d$ and $\mathcal{X}_s^u$ results in a valid tightening of the sibling domains:

$$\begin{aligned} \mathcal{X}_s^{d\prime} &\leftarrow \mathcal{X}_s^d \cap \mathcal{X}^{p\prime} \\ \mathcal{X}_s^{u\prime} &\leftarrow \mathcal{X}_s^u \cap \mathcal{X}^{p\prime} \end{aligned} \qquad \text{(sibling } s\text{-domain tightening)}$$

Subroutine 3: `processSiblings` (p, d, u)

1 **foreach** $s \in \mathcal{S}$ **do**
2 **foreach** $n \in \{d, u\}$ **do**
3 CP based on SP_s^n; fathom by dominance or infeasibility
4 **end**
5 apply parent s-domain tightening; fathom by infeasibility;
6 **end**
7 apply $\mathcal{X}_s^p$ aggregation, apply sibling s-domain tightening $\forall s \in \mathcal{S}$, fathom by infeasibility;
8 **foreach** $s \in \mathcal{S}$ **do**
9 **foreach** $n \in \{d, u\}$ **do**
10 OBBT based on LP_s^n; fathom by dominance
11 **end**
12 apply parent s-domain tightening; fathom by infeasibility;
13 **end**
14 apply $\mathcal{X}_s^p$ aggregation, apply sibling s-domain tightening $\forall s \in \mathcal{S}$, fathom by infeasibility;
15 **foreach** $s \in \mathcal{S}$ **do**
16 **foreach** $n \in \{d, u\}$ **do**
17 solve LP_s^n, set $\underline{f}_s^n \leftarrow f_{\text{LP},s}^n$, and fathom by infeasibility
18 **end**
19 check for s-domination; fathom by dominance;
20 **end**
21 **foreach** $s \in \mathcal{S}$ **do**
22 **foreach** $n \in \{d, u\}$ **do**
23 do DBBT based on LP_s^n; fathom by dominance
24 **end**
25 apply parent s-domain tightening; fathom by infeasibility;
26 **end**
27 apply $\mathcal{X}_s^p$ aggregation, apply sibling s-domain tightening $\forall s \in \mathcal{S}$, fathom by infeasibility;
28 $\widetilde{\boldsymbol{x}}^p \leftarrow$ solution of LP_s^n with s from a variant of s^{rep} that considers all feasible scenarios for $n \in \{d, u\}$;
29 **foreach** $s \in \mathcal{S}$ **do**
30 $(\widetilde{\boldsymbol{y}}_s^p, \overline{f}_s^p) \leftarrow$ solution and objective value of RP_s^p, check for s-domination; fathom by dominance;
31 **end**
32 update $(\widetilde{\boldsymbol{y}}^p, \overline{f}^p)$ via UB^p;
33 **return** $(\underline{f}^p, \overline{f}^p, \widetilde{\boldsymbol{x}}^p, \widetilde{\boldsymbol{y}}^p)$

With this in place we can now review Subroutine 3. For each scenario, we execute the range reduction and lower bounding routines for the corresponding subproblem of both siblings. Any of these routines may indicate that either d or u can be fathomed because the subproblem for some scenario s is dominated or infeasible. However, the results from the remaining subproblems of the fathomable sibling can still be combined with the results of the subproblem for s from the other sibling to generate child nodes. Thus we continue the sibling iteration as long as for each scenario there is at least one feasible, undominated subproblem from either sibling. For lower bounding (Lines 15–20 in Subroutine 3) we solve the subproblems LP_s^n, using the associated domains after CP (Lines 1–6) and OBBT (Lines 8–13). Following this, we perform DBBT (Lines 21–26). We perform all range reduction (Lines 1–6, Lines 8–13, and Lines 21–26), as well as bounding (Lines 15–20, and Lines 29–31) in parallel. Based on the final variable domains and objective bounds, we can generate processed orthant nodes as detailed in Sect. 4.4. In analogy to Subroutine 2, we could solve one upper bounding problem for each such orthant node, however, this would result in an exponential number of upper bounding problems. Instead, we choose to solve only a single set of upper bounding problems $\text{RP}_s^{\mathcal{Y}_s^p}(\widetilde{\boldsymbol{x}}^p)$ (Lines 29–31 in Subroutine 3), using the $\mathcal{Y}_s^p$

domains, resulting from parent s-domain tightening after DBBT. We select $\tilde{\boldsymbol{x}}^n$ to be one of the first-stage solutions of the feasible subproblems of both siblings, based on a representative scenario s^{rep}, that takes into account the subproblems of both siblings.

4.4 Filtered multisection

In this section we present a filtered multisection that addresses issues pertaining to the inherently exponential number of child nodes resulting from multisection branching, as presented in Sect. 3. After motivating this filtered multisection we give a formal statement in Subroutine 4. Following this, we comment on the possibility of adapting a related approach used in [21], for branching on first-stage variables. Finally we present the generation of orthant nodes in Subroutine 5.

The ability to generate 2^{N_s} bounded child nodes by processing and recombining the results from just two sibling nodes may seem attractive, however, handling an exponential number of nodes for arbitrary N_s can quickly become an issue in practice. Consider for instance a simple problem with $N_x = N_y = 1$; simply storing the variable bounds of child nodes from a single second-stage branching as 8 byte double values requires $16\,(1 + N_s)\,2^{N_s}$ bytes, e.g., more than two terabytes of memory for $N_s = 32$. At least in principle, we could avoid this memory issue by generating the nodes on demand in later iterations, however, doing this in an appropriate order, e.g., by increasing lower bound would require additional computations. More importantly, it is possible that for some of the bisections neither of the two subproblems improves the lower bound of the parent node significantly. This can result in a large number of nodes with weak objective bounds that all need to be processed separately, slowing down the algorithm.

To address this issue, we can select a subset of the N_s bisection candidates that allows for a significant increase of the lower bound or reduction of the overall domain size, compared to the parent node. We then revert the original multisection in favor of a second, filtered multisection, comprising only the variable instances corresponding to the selected bisection candidates. For this, we use Subroutine 4, which will be presented in the following. Note that each bisection candidate corresponds to a particular scenario s, a branched variable instance $y_{s,i}$, and two associated sibling subproblems with complementary domains for $y_{s,i}$. For each bisection candidate we get one of three results:

Case 1) Both subproblems are infeasible, this immediately implies infeasibility of the parent node p.

Case 2) Exactly one subproblem is infeasible, only the domain of the feasible subproblem can contribute to the generation of feasible orthant nodes, i.e., selecting this bisection candidate does not increase their number.

Case 3) Both subproblems are feasible, selecting this bisection candidate doubles the resulting number of orthant nodes.

Since Case (1) is already addressed by the fathoming rules in Subroutine 3, Subroutine 4 only needs to address Cases (2) and (3). We select all bisection candidates from Case (2) (Lines 4–7 in Subroutine 4), as they effectively result in a domain reduction, without affecting the number of generated nodes. The feasible subproblems associated with these bisection candidates are stored in the map $\mathcal{M}$. The remaining bisection candidates, corresponding to Case (3), are collected in $\mathcal{L}$ (Line 9).

As the number $k \leq N_s$ of bisection candidates selected from Case (3) determines the resulting number of child nodes, we call k the *"effective number of bisections"*. To determine which bisection candidates should be selected, we use a heuristic based on strong-branching

Subroutine 4: `filteredMultiSection(`p, d, u`)`

```
1  M ← empty map;                    /* mapping s with single feasible subproblem to
                                        the corresponding sibling */
2  L ← empty list;                   /* containing s for which both sibling subproblems
                                        are feasible */
3  foreach s ∈ S do
4      if f_s^d = ∞ then
5          M[s] = u;                 // variable corresponding to s will be branched
6      else if f_s^u = ∞ then
7          M[s] = d;                 // variable corresponding to s will be branched
8      else
9          append s to L;            /* variable corresponding to s might be
                                        branched (see Lines 16-17) */
10     end
11 end
12 σ_max = max_{s∈L} σ_s;
13 if σ_max ≤ ε_σ^2 then
14     replace σ_s and σ_max with scores based on relative widths of variable domains
15 end
16 delete all s for which σ_s ≤ ησ_max from L;
17 delete all but the k_max best entries from L;
18 return (M, L);
```

scores [52, 53]: given sibling nodes d and u obtained from the parent node p, each bisection candidate, i.e., each scenario s, is assigned a strong-branching score σ_s. For this, we employ the default scoring function of SCIP, proposed in [54], which is calculated as

$$\sigma_s := \max(\underline{f}_s^d - \underline{f}_s^p, \varepsilon_\sigma) \max(\underline{f}_s^u - \underline{f}_s^p, \varepsilon_\sigma) \tag{σ_s}$$

Here the constant ε_σ ensures a nonnegative score for cases where only one sibling improves upon the parent bound.

We only keep scenarios from $\mathcal{L}$ with a score of at least $\eta\sigma_{\max}$, where $\eta \in (0, 1]$ and $\sigma_{\max}$ is the largest of the scores Lines 12 and 15. Additionally, a *maximum number of effective bisections* $k_{\max}$ is imposed to ensure that the filtered multisection produces at most $2^{k_{\max}}$ child nodes (Line 17). If all scores are smaller than ε_σ^2, we instead rank and select bisection candidates based on relative widths of the associated variable domains (Lines 12 and 15). This ensures exhaustive partitioning in the limit, necessary for the convergence of MUSE-BB, also see Lemma 1 and Corollary 3 in Sect. 5. A visualization of the proposed multisection branching procedure is given in Fig. 2.

The use of strong-branching scores in Subroutine 4 suggests a relation between filtered multisection and standard strong-branching, where *alternative* bisections of a set of N_v variables are considered. While standard strong-branching requires processing $2\,N_v$ *full nodes* to select a single bisection, i.e., generate 2 child nodes, we only process $2\,N_s$ *subproblems* (equivalent to 2 full nodes) and may generate an exponential number of nodes in each filtered multisection. Nevertheless, standard strong-branching might also be useful in MUSE-BB, as indicated by its use in the related algorithm of [21] for the selection of first-stage variables: in each iteration, the authors consider all elements of $\boldsymbol{x}$ via strong-branching, solving LP relaxations of the associated instances of DE^n for the $2\,N_x$ child nodes. For the two nodes of the selected bisection, they then perform the global solution of the subproblems SP_s, required for the convergence of their algorithm. While a similar approach could also be

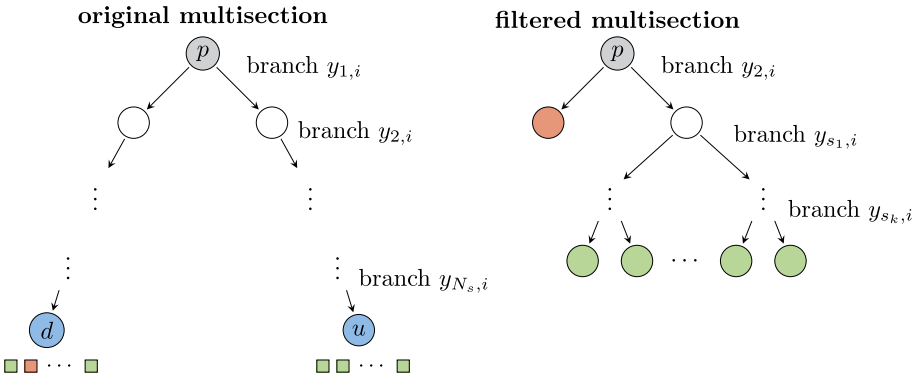


Fig. 2 Example for multisection branching and filtered multisection. In the original multisection (left) the parent node p is branched on all second stage variable instances $y_{s,i}$ for a given variable index i. Instead of generating all 2^{N_s} nodes, we only generate the leftmost and rightmost node, corresponding to branching all variable instances down (d) or up (u), respectively. We process these sibling nodes (blue) by solving the resulting subproblems (squares). In the example, the subproblem for $s = 2$ of d is infeasible (red) while all other subproblems are feasible (green). Right: based on the subproblem results, we perform a second, filtered multisection of p, involving a subset k of the original N_s bisection candidates (right). This can be interpreted as generating a new tree of k sequential bisections: we keep all bisections producing exactly one feasible subproblem (here only the bisection of $y_{2,i}$), as they do not increase the total number of child nodes. For the bisections resulting in two feasible subproblems, we consider the bound improvement w.r.t. the corresponding subproblems of p to compute the associated strong-branching scores σ_s. The bisection candidates are then filtered based on the values of σ_s and the algorithm parameters η and $k_{\max}$. We reject bisections for which improvement is considered insufficient, i.e., those with $\sigma_s < \eta\sigma_{\max}$, for a threshold $\eta \in (0, 1]$. The remaining ones are sorted by descending strong-branching score, resulting in an ordering of the associated scenarios (i.e., $s_1, ..., s_k$). Of these bisections we keep at most $k_{\max}$. Finally, we generate the corresponding 2^k orthant nodes (green) using appropriate combinations of domains and bounds from the feasible sibling subproblems. (Color figure online)

adopted in MUSE-BB, we do not require expensive global bounding routines for convergence; hence solving full-space LP relaxations based on DE^n is relatively expensive in our case. Alternatively we could solve the decomposable LP relaxations LP^n_s, and aggregate the strong-branching scores σ_s, e.g., via a w_s-weighted sum. As pointed out above, this would require to process $2\,N_x$ nodes instead of just 2. Due to the importance of first-stage branching for TSP (also see Sect. 6), this effort may in fact be warranted, however, we do not consider this idea further here, and instead branch only on individual first-stage variables as indicated in Subroutine 1.

The map $\mathcal{M}$, and list $\mathcal{L}$, returned by Subroutine 4 are used within Subroutine 5 for the generation of individual orthant nodes. For this, we collect the appropriate variable domains and subproblem objective values for each orthant from one of the siblings or the parent node (Lines 16–18 in Subroutine 5). For each scenario s, the respective node is determined, based on whether the associated bisection was selected ($s \in \mathcal{M}$ or $s \in \mathcal{L}$) or not (Lines 4–15). If s is in the map $\mathcal{M}$, we only use the data from the feasible subproblem of the associated sibling node (Lines 4 and 5). If instead, the scenario is in $\mathcal{L}$, appropriate subproblem data is taken based on the orthant index i (cf. Line 10 of Algorithm 1) to determine the sibling node from which to use data (Lines 6–12 in Subroutine 5). Otherwise the bisection is rejected, i.e., we use the data from the parent (Line 14). Note that the latter case does not imply that the solution of the associated subproblems was in vain, as it may still result in tightened variable bounds due to parent s-domain tightening (Lines 5, 12 and 25 in Subroutine 3).

Subroutine 5: `generateOrthantNode`$(i, p, d, u, \mathcal{M}, \mathcal{L})$

1 $\boldsymbol{b} \leftarrow$ vector of $|\mathcal{L}|$ bits, representing i;
2 $\mathcal{X}^o \leftarrow \mathcal{X}^p$;
3 **foreach** $s \in \mathcal{S}$ **do**
4 **if** $s \in \mathcal{M}$ **then**
  `// use data from the feasible subproblem of bisection` s
5 $n \leftarrow \mathcal{M}[s]$;
6 **else if** $s \in \mathcal{L}$ **then**
  `// use data from the sibling subproblem corresponding to orthant id` i
7 $j \leftarrow$ position of s in $\mathcal{L}$;
8 **if** $b_j = 0$ **then**
9 $n \leftarrow d$;
10 **else**
11 $n \leftarrow u$
12 **end**
13 **else**
  `// use parent data (bisection` s `was filtered in Lines 16–17 of Subroutine 4)`
14 $n \leftarrow p$;
15 **end**
16 $\mathcal{X}^o \leftarrow \mathcal{X}^o \cap \mathcal{X}^n_s$;
17 $\mathcal{Y}^o_s \leftarrow \mathcal{Y}^n_s$;
18 $\underline{f}^o_s \leftarrow \underline{f}^n_s$;
19 **end**
20 $\mathcal{Y}^o \leftarrow \bigtimes_{s\in\mathcal{S}} \mathcal{Y}^o_s$;
21 $\underline{f}^o \leftarrow \sum_{s\in\mathcal{S}} w_s \underline{f}^o_s$;
22 **if** $\mathcal{X}^o = \emptyset$ or $\underline{f}^o > \overline{f}$ **then** $\underline{f}^o = \infty$;
23 **return** o;

Once data for all scenarios has been collected, we aggregate the overall second-stage domain and scenario-weighted lower bound (Lines 20 and 21). Finally we test whether the orthant node is infeasible or dominated and return it (Line 22).

5 Theoretical results

In this section we present convergence results for the lower bounding schemes used in MUSE-BB, and highlight the connection to the convergence of the algorithm itself. When applied to the domains of individual B&B nodes, the lower bounding problems presented in Sect. 4.1 give rise to different *lower bounding schemes* (LBSs). Their quality is determined by their underestimation of the true optimal value, and their capacity to quickly detect infeasible subdomains. In the following we analyze the asymptotic behavior of these two qualities for LBSs relevant to MUSE-BB, as the size of B&B nodes diminishes. In particular, we consider the LBSs based on: (i) dropping or dualizing the NACs, corresponding to the subproblems SP^n_s, or LSP^n_s, respectively, (ii) the McCormick relaxations of subproblems from (i), and (iii) the linear programming relaxations, resulting from subtangent relaxation of subproblems from (ii). Formally, the asymptotic behavior of a LBS for a sequence of descendant nodes is quantified by the convergence order [55]. We first introduce additional notation and definitions related to this convergence order in Sect. 5.1, and then present conditions under which different LBSs achieve first- and second-order convergence, respectively in Sects. 5.2

and 5.3. As a result of the first-order convergence, we show that MUSE-BB guarantees finite termination with an ε_f-optimal solution in Sect. 5.2. In Sect. 5.3 we analyze an extension of MUSE-BB in which the NACs are dualized instead of dropped. We show that employing this dualization within MUSE-BB is equivalent to adding the terms $\boldsymbol{\lambda}_s{}^{\mathsf{T}}\boldsymbol{x}_s$, to the objective function relaxations in the subproblems LP_s^n, and performing dual iterations to update the multipliers $\boldsymbol{\lambda}_s$. Provided optimal multipliers $\boldsymbol{\lambda}_s^*$ are obtained, we show that this results in stronger convergence properties, with implications for the so-called cluster effect [25, 26]. In particular, while theoretical results of [28] indicate that the current implementation of MUSE-BB may mitigate clustering around typical constrained minimizers, mitigating clustering around typical unconstrained minimizers may require an extension such as the one presented in Sect. 5.3.

Before we give the formal definition of a LBS and the associated convergence order, we highlight the impact of the quality of lower bounds via two examples. For this we define the width of an interval.

Definition 1 (*width of a multidimensional interval*) A measure for the width of a multidimensional interval $\mathcal{V} = \times_{i\in\{1,\dots,m\}} \left[\underline{v}_i, \overline{v}_i\right] \subset \mathbb{R}^m$ is given by:

$$\mathrm{W}(\mathcal{V}) := \max_{i\in\{1,\dots,m\}} \left(\overline{v}_i - \underline{v}_i\right)$$

As shown by [28], the occurrence of clustering is related to the convergence order of the LBS, which in turn is defined in terms of the 'size of B&B nodes', i.e., the width of the domain of branched variables, measured by Definition 1 [also see 55]. In algorithms like PBDAs, this node size is given by $\mathrm{W}(\mathcal{X}^n)$, whereas in algorithms like MUSE-BB it is given by the width of the overall variable domain, i.e., $\mathrm{W}(\mathcal{Z}^n)$. While MUSE-BB will of course require more branching than PBDAs to reach a given node size, the LBSs used in MUSE-BB may achieve a higher convergence order than the scheme $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$, used in PBDAs.

The following example illustrates this situation for LBSs based on SP_s, i.e., the simplest scenario relaxation, corresponding to dropping the NACs from $\mathrm{DE}_{\mathrm{NAC}}^{\mathcal{X},\mathcal{Y}}$: while the scheme $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$, where only $\mathcal{X}$ is partitioned, results in an absolute optimality gap that diminishes with $\sqrt{\mathrm{W}(\mathcal{X}^n)}$, the gap produced by the scheme $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$, which additionally partitions $\mathcal{Y}$, diminishes with $\mathrm{W}(\mathcal{Z}^n)$.

Example 1 Consider the following instance of $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ with $N_x = N_y = 1$, $N_s = 2$ and an original domain with $\mathcal{X} = \mathcal{Y}_1 = \mathcal{Y}_2 = [0, 2]$. Take

$$\begin{aligned} w_1\, f_1(x_1, y_1) &= -y_1; \quad \boldsymbol{g}_{\mathrm{II},1}(x_1, y_1) = -x_1 + y_1^2 \\ w_2\, f_2(x_1, y_2) &= 2\, y_2; \quad \boldsymbol{g}_{\mathrm{II},2}(x_2, y_2) = x_2 - y_2^2 \end{aligned}$$

The objectives imply that at the optimum $\boldsymbol{z}^{\mathrm{DE}^n} = (x_1^{\mathrm{DE}^n}, y_1^{\mathrm{DE}^n}, y_2^{\mathrm{DE}^n})$ of DE^n, $y_1^{\mathrm{DE}^n}$ is maximized and $y_2^{\mathrm{DE}^n}$ is minimized. For any feasible node n with $\mathcal{Z}^n = [\underline{x}^n, \overline{x}^n] \times [\underline{y}_1^n, \overline{y}_1^n] \times [\underline{y}_2^n, \overline{y}_2^n]$, the bounds and constraints imply $y_1^{\mathrm{DE}^n} \leq \min\{\sqrt{x^{\mathrm{DE}^n}}, \overline{y}_1^n\}$ and $y_2^{\mathrm{DE}^n} \geq \max\{\sqrt{x^{\mathrm{DE}^n}}, \underline{y}_2^n\}$. We have $y_1^{\mathrm{DE}} = y_2^{\mathrm{DE}} = \sqrt{x^{\mathrm{DE}}}$ on the original domain, and thus $f(x^{\mathrm{DE}}, y_1^{\mathrm{DE}}, y_2^{\mathrm{DE}}) = \sqrt{x^{\mathrm{DE}}}$, which is minimized at $\boldsymbol{z}^{\mathrm{DE}} = (x^{\mathrm{DE}}, y_1^{\mathrm{DE}}, y_2^{\mathrm{DE}}) = (0, 0, 0)$, with objective value 0.

Now consider the lower bounds generated by lower bounding schemes based on SP_s on any nested sequence of nodes converging to the optimum $\boldsymbol{z}^{\mathrm{DE}}$. Since all nodes in such sequences satisfy $\underline{x}^n = \underline{y}_s^n = 0$, the optimal solutions of the associated instance of SP_s

satisfy $y_1^{\mathrm{SP}^n} = \min\{\sqrt{\overline{x}^n}, \overline{y}_1^n\}$ and $y_2^{\mathrm{SP}^n} = \max\{\sqrt{\underline{x}^n}, \underline{y}_2^n\} = 0$, and thus from the constraint $\boldsymbol{g}_{\mathrm{II},2}$, we have $x_2^{\mathrm{SP}^n} = y_2^{\mathrm{SP}^n} = 0$.

In $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$, only x is branched, and the width of a node n corresponds to $W^n = \mathrm{W}(\mathcal{X}^n) = \overline{x}^n$, while $\mathrm{W}\left(\mathcal{Y}_s^n\right) = \overline{y}_s^n = 2$ remains constant. Since $\overline{x}^n < 2$, we have: $y_1^{\mathrm{SP}^n} = \sqrt{W^n}$, and thus $f_{\mathrm{DE}}^n - f_{\mathrm{SP}}^n = \sqrt{W^n}$.

In $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$, both x and y_s are branched, and the width of a node n corresponds to $W^n = \mathrm{W}(\mathcal{Z}^n)$. For a given width W^n, the largest value for $f_{\mathrm{DE}}^{n'} - f_{\mathrm{SP}}^{n'}$ over all nodes n' with $\mathrm{W}\left(\mathcal{Z}^{n'}\right) = W^n$ will be produced by the node n with $\overline{x}^n = \overline{y}_s^n = W^n$. Once $W^n < 1$, we have that $\sqrt{W^n} > W^n$, and thus $y_1^{\mathrm{SP}^n} = W^n$, and $f_{\mathrm{DE}}^n - f_{\mathrm{SP}}^n = W^n$.

While Example 1 shows that for certain problems the scheme $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$ will produce weaker bounds than $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$ for a given node width, the following example demonstrates that this is not always the case, i.e., both LBSs may produce absolute optimality gaps that diminish linearly (and not better) with the node width.

Example 2 Take Example 1, but change the constraints to

$$\boldsymbol{g}_{\mathrm{II},1}(x_1, y_1) = -x_1 + y_1; \qquad \boldsymbol{g}_{\mathrm{II},2}(x_2, y_2) = x_2 - y_2.$$

which implies that $y_1^{\mathrm{DE}} = y_2^{\mathrm{DE}} = x^{\mathrm{DE}}$ on the original domain, and thus $f(x^{\mathrm{DE}}, y_1^{\mathrm{DE}}, y_2^{\mathrm{DE}}) = x^{\mathrm{DE}}$. This is again minimized at $\boldsymbol{z}^{\mathrm{DE}} = (x^{\mathrm{DE}}, y_1^{\mathrm{DE}}, y_2^{\mathrm{DE}}) = (0, 0, 0)$, with objective value 0. Now for both $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$, and $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$, it is easy to see that $x_2^{\mathrm{SP}^n} = y_2^{\mathrm{SP}^n} = 0$, and $x_1^{\mathrm{SP}^n} = y_1^{\mathrm{SP}^n} = W^n$, resulting in $f_{\mathrm{DE}}^n - f_{\mathrm{SP}}^n = W^n$, i.e., an optimality gap that decreases exactly linearly with the node width.

As we shall see in Sect. 5.1, β-order convergence of a LBS requires that the optimality gap decreases proportionally to $(W^n)^\beta$, with $\beta > 0$, i.e., a higher value of β is associated with a better quality of the LBS. [24] showed that the convergence orders below one of LBSs used in PBDAs are inherent to the projection resulting from running a B&B in the $\mathcal{X}$ space only. In particular, even LBSs based on the ideal relaxation, i.e., on convex envelopes of the scenario value functions $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ may have less than first-order convergence, unless $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ is Lipschitz, which is not guaranteed in general. In contrast, we show in Sect. 5.2 that the scheme $\mathrm{SP}_s^n = \mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$, obtained by simply dropping the NACs, has at least first-order convergence under the much milder assumption that the objective and constraint functions of DE are Lipschitz. If additionally, the used convex relaxations are Lipschitz, subsequent convex and linear relaxations used in MUSE-BB preserve this first-order convergence.

As demonstrated by Examples 1 and 2, the convergence order may still be as low as one, despite branching on second-stage variables. In Sect. 5.3 we show that this limitation is inherent to dropping the NACs, and that dualizing them instead results in a LBS that is as least as strong as the presented one, but additionally guarantees second-order convergence at unconstrained minimizers.

Despite this promising outlook for MUSE-BB, we need to point out that the seemingly superior convergence order of LBSs for MUSE-BB compared to that of PBDAs may be relativized by the fact that the occurrence of clustering is not exclusively determined by convergence order, but also by the local growth order of objective and constraint functions, see [28]. Even if for a given problem, a LBS for MUSE-BB has a higher convergence order than a comparable scheme for a PBDA, the lower order might still be sufficient to mitigate clustering in PBDAs. This is because by operating in the projected space, the relevant growth order for

PBDAs is that of of the scenario value functions $f_s^{\mathcal{X},\mathcal{Y}_s}$, which may also be reduced compared that of the original objective functions f_s. In Example 1, e.g., we have $f_1^{\mathcal{X},\mathcal{Y}_1}(x) = \sqrt{x}$, and thus a growth order of 1/2, matching the convergence order of the scheme $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$, indicating that clustering might still be avoided, despite the reduced convergence order. Conditions for which PBDAs or algorithms like MUSE-BB will show superior performance are thus not immediately clear from the present analysis.

5.1 Preliminaries

To avoid the so-called cluster effect [25–27] where a B&B algorithm visits a large number of nodes near approximate global minimizers, LBSs need to exhibit a sufficiently large convergence order. Early works on clustering [25–27] focused on clustering around unconstrained minimizers, where the convergence order of LBSs is equivalent to the convergence order of the relaxations used for the objective function. Around constrained minimizers, on the other hand, one additionally needs to consider the effect of relaxing the feasible set, leading to an extended notion of convergence order [28, 55], which additionally depends on the convergence orders of the relaxations used for the constraint functions. In B&B for general nonlinear programming problems, relaxations of objective and constraints are typically generated by convex relaxation methods. In [29] we therefore analyzed the convergence order of McCormick [41], α-BB [56], and convex hull relaxations. Convergence orders for (further) relaxation through polyhedral outer approximation were investigated by [57–59]. While [28] consider a classical LBS for general nonlinear programming problems based on convex relaxation, their conclusions are not dependent on this type of LBS. [55] present a more general definition of a LBS, and give conditions under which convex and Lagrangian relaxations with appropriate convergence orders result in first- and second-order convergent LBS.

In preparation for Definition 4, where we use an extended notion of convergence order of a LBS in the sense of [55], we introduce additional nomenclature and definitions. For each B&B node n and the corresponding *subproblem domains* $\mathcal{Z}_s^n := \mathcal{X}^n \times \mathcal{Y}_s^n$, we introduce the *scenario-specific feasible sets*

$$\mathcal{F}_s^n := \{(\boldsymbol{x}, \boldsymbol{y}_s) \in \mathcal{Z}_s^n : \boldsymbol{g}_{\mathrm{I}}(\boldsymbol{x}) \leq \boldsymbol{0}, \boldsymbol{g}_{\mathrm{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) \leq \boldsymbol{0}\}.$$

Similarly, for the *overall domains* $\mathcal{Z}^n := \mathcal{X}^n \times \mathcal{Y}^n$, associated with each node n, we express the *feasible set* of $\mathrm{DE}^n = \mathrm{DE}^{\mathcal{X}^n,\mathcal{Y}^n}$ as

$$\mathcal{F}^n := \{(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{Z}^n : (\boldsymbol{x}, \boldsymbol{y}_s) \in \mathcal{F}_s^n \, \forall s \in \mathcal{S}\}.$$

Furthermore, since we branch on both $\boldsymbol{x}$ and $\boldsymbol{y}$, the distinction between them becomes irrelevant in many parts of the following analysis. For conciseness, we therefore aggregate the first- and second-stage variables, i.e., we introduce the notation $(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_{N_s}) =: \boldsymbol{z} \in \mathcal{Z}^n \subset \mathbb{R}^{N_z}$, and $(\boldsymbol{x}, \boldsymbol{y}_s) =: \boldsymbol{z}_s \in \mathcal{Z}_s^n \subset \mathbb{R}^{N_{z,s}}$, where, $N_z := N_x + N_s N_y$ and $N_{z,s} := N_x + N_y$.

Definition 2 (*distance between two sets*) A measure for the distance of two sets $\mathcal{Z}_1, \mathcal{Z}_2, \subset \mathbb{R}^m$ is given by:

$$\mathrm{d}(\mathcal{Z}_1, \mathcal{Z}_2) := \inf_{\substack{z_1 \in \mathcal{Z}_1 \\ z_2 \in \mathcal{Z}_2}} \|z_1 - z_2\|$$

Throughout this text, $\|\bullet\|$ denotes the Euclidian norm.

Definition 3 (*violation measure*) A measure for the minimum constraint violation of some optimization problem $\mathrm{P}(\mathcal{V})$ with variable domain $\mathcal{V} \subset \mathbb{R}^{N_v}$ and constraints $\boldsymbol{g}_\mathrm{P} : \mathbb{R}^{N_v} \to \mathbb{R}^{N_g^\mathrm{P}}$, on some subdomain $\mathcal{V}^n \subset \mathcal{V}$ is given by:

$$\begin{aligned} \mathrm{vio}_\mathrm{P}^n &:= \mathrm{d}\left(\{\boldsymbol{g}_\mathrm{P}(\boldsymbol{v}) : \boldsymbol{v} \in \mathcal{V}^n\}, \mathbb{R}_-^{N_g^\mathrm{P}}\right) \\ &= \min_{\boldsymbol{v} \in \mathcal{V}^n} \left(\sum_{j=1}^{N_g^\mathrm{P}} \max\{g_{\mathrm{P},j}(\boldsymbol{v}), 0\}^2 \right)^{1/2}, \end{aligned}$$

where $\mathbb{R}_-$ denotes the nonpositive orthant.

Alternative to Definition 3, one may also define the violation in terms of, e.g., the ∞-norm, which would yield $\min_{\boldsymbol{v} \in \mathcal{V}^n} \max_{j \in \{1,\ldots N_g^\mathrm{P}\}} \max\{0; g_{\mathrm{P},j}(\boldsymbol{v})\}$.

We chose Definition 3, following [28, 55], who use it, within their definitions of convergence order of LBSs (Definition 8 and 14, respectively). For clarity, we separate the definition of violation from that of convergence order.

We adapt Definition 14 of [55] to scenario-based LBSs of $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$. All such schemes effectively lift the deterministic equivalent formulation DE^n to the equivalent nonanticipativity formulation $\mathrm{DE}_\mathrm{NAC}^{\mathcal{X},\mathcal{Y}}$, which introduces separate first-stage variables and constraints for each scenario and couples them via the NACs. Following this, scenario-based LBSs obtain relaxations of $\mathrm{TSP}^{\mathcal{X},\mathcal{Y}}$, by dropping or dualizing the NACs from $\mathrm{DE}_\mathrm{NAC}^{\mathcal{X},\mathcal{Y}}$, potentially followed by further relaxations of the objective and constraints.

Definition 4 (*Hausdorff convergence order of scenario-based LBSs*) Denote the optimal objective value of DE^n as f_DE^n, and let R^n be any relaxation of DE^n that decomposes into the N_s scenario relaxations of the form:

$$f_{\mathrm{R},s}^n := \min_{z_s \in \mathcal{F}_{\mathrm{R},s}^n} f_{\mathrm{R},s}(z_s) \qquad \mathrm{R}_s^n$$

where for each s, the feasible set $\mathcal{F}_{\mathrm{R},s}^n$ contains $\mathcal{F}_s^n$, and the objective functions $f_{\mathrm{R},s}$ are such that the weighted sum of the optimal objective values $f_{\mathrm{R},s}^n$ underestimates f_DE^n, i.e.,

$$f_\mathrm{R}^n := \sum_{s \in \mathcal{S}} w_s \, f_{\mathrm{R},s}^n \leq f_\mathrm{DE}^n.$$

We say that (the LBS based on) R_s^n has:

1. β_f-order (Hausdorff) convergence at a feasible point $\boldsymbol{z} \in \mathcal{Z}$ if there exists $C_f > 0$ such that for every $\mathcal{Z}^n \subset \mathcal{Z}$ with $\boldsymbol{z} \in \mathcal{Z}^n$,

$$f_\mathrm{DE}^n - f_\mathrm{R}^n \leq C_f \, \mathrm{W}\left(\mathcal{Z}^n\right)^{\beta_f}$$

2. β_g-order (Hausdorff) convergence at an infeasible point $\boldsymbol{z} \in \mathcal{Z}$ if there exists $C_g > 0$ such that for every $\mathcal{Z}^n \subset \mathcal{Z}$ with $\boldsymbol{z} \in \mathcal{Z}^n$,

$$\mathrm{vio}_\mathrm{DE}^n - \mathrm{vio}_\mathrm{R}^n \leq C_g \, \mathrm{W}\left(\mathcal{Z}^n\right)^{\beta_g}$$

We say that (the LBS based on) R_s^n has (Hausdorff) convergence of order β on $\mathcal{Z}$ if is has β-order (Hausdorff) convergence at each $\boldsymbol{z} \in \mathcal{Z}$.

The generic scenario-based relaxation R_s^n encompasses all LBSs we consider: LSP_s^n; SP_s^n; the additional relaxation of these problems, resulting from replacing all functions by their McCormick relaxations on $\mathcal{Z}^n$ (i.e., MC_s^n in the case of SP_s^n); and the linear outer approximation of MC_s^n through subtangents, LP_s^n. In all cases, the convergence order is with respect to DE^n, i.e., feasibility and infeasibility are always to be understood with respect to the original variables and constraints. As in Definition 14 of [55], the convergence order at feasible [infeasible] points establishes an upper bound on the underestimation of the optimal objective value [minimal constraint violation] in terms of the node width. Thus, the theoretical results of [28] are directly applicable. In particular, assuming sufficiently small prefactors C_f and C_g, and that all minimizers are strict, the previous analyses indicate that second-order convergence at feasible points mitigates clustering around unconstrained minimizers located at points of differentiability [27, 28], while first-order convergence suffices for unconstrained minimizers if they are located at points of nondifferentiability [28, 60]. At constrained minimizers, on the other hand, first-order convergence may mitigate clustering if the objective and active constraints grow linearly around the minimizer [28].

Note that according to Definition 3, the constraint violations $\mathrm{vio}_{\mathrm{DE}}^n$ and $\mathrm{vio}_{\mathrm{R}}^n$ are defined relative to the overall constraints of the respective problems. In contrast to DE^n, all scenario relaxations R_s^n by definition have separate copies of the first-stage variables $\boldsymbol{x}$ and the first-stage constraints $\boldsymbol{g}_{\mathrm{I}}$ (or their relaxations) for each scenario s. Hence, the total number of variables and constraints of the N_s subproblems R_s^n are $N_\xi := N_s(N_x + N_y)$, and $N_g^{\mathrm{R}} := N_s(N_{\mathrm{I}} + N_{\mathrm{II}})$, respectively. Similarly to $\boldsymbol{g}_{\mathrm{DE}}$, we define $\boldsymbol{g}_{\mathrm{R}}$ by aggregating the constraint functions of R_s^n for all s; i.e., $\boldsymbol{g}_{\mathrm{R}}$ is the vector-valued function $\boldsymbol{g}_{\mathrm{R}} : \bigtimes_{s\in\mathcal{S}}(\mathcal{Z}_s) \mapsto \mathbb{R}^{N_g^{\mathrm{R}}}$, such that for $\boldsymbol{\xi} = (\boldsymbol{x}_1, \boldsymbol{y}_1, \ldots, \boldsymbol{x}_{N_s}, \boldsymbol{y}_{N_{s,N_y}}) \in \bigtimes_{s\in\mathcal{S}}(\mathcal{Z}_s) \subset \mathbb{R}^{N_\xi}$ we have:

$$\boldsymbol{g}_{\mathrm{R}}(\boldsymbol{\xi}) := \begin{pmatrix} g_{\mathrm{R},1}(\boldsymbol{\xi}) \\ \vdots \\ g_{\mathrm{R},N_g^{\mathrm{R}}}(\boldsymbol{\xi}) \end{pmatrix},$$

e.g., when using LSP_s^n or SP_s^n for R_s^n, we define these entries as

$$\boldsymbol{g}_{\mathrm{LSP}}(\boldsymbol{\xi}) = \boldsymbol{g}_{\mathrm{SP}}(\boldsymbol{\xi}) = \begin{pmatrix} g_{\mathrm{I},1}(\boldsymbol{x}_1) \\ \vdots \\ g_{\mathrm{I},N_{\mathrm{I}}}(\boldsymbol{x}_{N_s}) \\ g_{\mathrm{II},1,1}(\boldsymbol{x}_1, \boldsymbol{y}_1) \\ \vdots \\ g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\boldsymbol{x}_{N_s}, \boldsymbol{y}_{N_s}) \end{pmatrix}.$$

Since the bounds in Definition 4 are relative to the width of the overall variable domain $\mathcal{Z}^n$, it is only meaningful for B&B algorithms for which this width diminishes to 0. MUSE-BB clearly satisfies this condition, as shown for completeness in the following result.

Lemma 1 (Exhaustive Subdivision) *The branching scheme used in MUSE-BB is* exhaustive*, i.e., in the limit all infinite sequences of descendant nodes converge to some accumulation point.*

Proof In Line 1 of Subroutine 1 we eventually select the variable corresponding to the dimension of $\mathcal{Z}^n$ with largest relative domain width (since the effect of different branching priorities is canceled after a finite number of iterations). While the bisection of the selected variable can still be rejected during variable filtering (Lines 16 and 17 in Subroutine 4), this

can only happen a finite number of times, as the strong-branching scores are based on lower bound *improvements* which inherently tend to zero. Thus, the width of all variable domains tends to zero. □

Note that since PBDAs only partition $\mathcal{X}$, $\mathrm{W}(\mathcal{Z}^n)$ would need to be substituted with $\mathrm{W}(\mathcal{X}^n)$ in the bounds of Definition 4 to obtain an appropriate alternative definition for PBDAs, also see the related Definition 14 and Sect. 5 of [55].

5.2 First-order convergence

As we shall see in Lemma 2, branching on second-stage variables $\boldsymbol{y}$ in addition to first-stage variables $\boldsymbol{x}$, resolves the possibility of convergence orders below one, i.e., SP_s^n can be guaranteed to have (at least) first-order convergence under the weak assumption of Lipschitz continuity of the objective and constraint functions. Furthermore, Corollaries 1 and 2 show that the additional relaxations used in MUSE-BB preserve first-order convergence.

Assumption 1 (*Lipschitz, factorable functions*) All constraint and objective functions are Lipschitz, i.e., there exist constants $L_{g,\mathrm{I},i} > 0;\ i = 1, \ldots, N_{\mathrm{I}}$, and for all $s \in \mathcal{S}$ there exist constants $L_{g,\mathrm{II},s,j} > 0,\ j = 1, \ldots, N_{\mathrm{II}};\ L_{f,s} > 0$, such that:

$$\begin{aligned}
\left|g_{\mathrm{I},i}(\boldsymbol{x}) - g_{\mathrm{I},i}(\boldsymbol{x}')\right| &\leq L_{g,\mathrm{I},i} \left\|\boldsymbol{x} - \boldsymbol{x}'\right\| \quad \forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}, i = 1, \ldots, N_{\mathrm{I}}, \\
\left|g_{\mathrm{II},s,j}(z_s) - g_{\mathrm{II},s,j}(z_s')\right| &\leq L_{g,\mathrm{II},s,j} \left\|z_s - z_s'\right\| \quad \forall z_s, z_s' \in \mathcal{Z}_s, j = 1, \ldots, N_{\mathrm{II}}, \\
\left|f_s(z_s) - f_s(z_s')\right| &\leq L_{f,s} \left\|z_s - z_s'\right\| \quad \forall z_s, z_s' \in \mathcal{Z}_s.
\end{aligned}$$

The following Lemma shows first-order convergence of the LBS based on SP_s^n. Its proof relies on the fact that in any given node n, points from the domains $\mathcal{Z}_s^n = \mathcal{X}^n \times \mathcal{Y}_s^n$ of scenario subproblems are at most $\sqrt{N_x + N_y}\,\mathrm{W}\left(\mathcal{Z}_s^n\right)$ apart. Furthermore, the overall node domain is $\mathcal{Z}^n = \mathcal{X}^n \times \mathcal{Y}^n$ and thus $\mathrm{W}\left(\mathcal{Z}_s^n\right) \leq \mathrm{W}(\mathcal{Z}^n)$. We point out that all of the algebraic steps in the following proof would also hold when replacing SP_s^n with the LBS $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$ used in PBDA. Thus in fact, both LBS have first-order convergence in the $\mathcal{Z}$ space. However, while for MUSE-BB $\mathrm{W}(\mathcal{Z}^n)$ tends to zero by Lemma 1, it does not for PBDAs, where $\mathcal{Y}^n = \mathcal{Y}$ for all nodes n, and hence only $\mathrm{W}(\mathcal{X}^n)$ tends to zero. A meaningful convergence order for $\mathrm{SP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$ would therefore require bounds in terms of $\mathrm{W}(\mathcal{X}^n)$ instead of $\mathrm{W}(\mathcal{Z}^n)$, also see the note before Lemma 1 and the related Definition 14 and Sect. 5 of [55].

Lemma 2 (first-order convergence of SP_s^n) *Under Assumption 1, SP_s^n has a convergence order of $\beta \geq 1$.*

Proof Recall that Definition 4 considers convergence orders at feasible and infeasible points with respect to DE^n, leading to a natural proof outline.

Convergence Order at Feasible Points: First consider some nested sequence of nodes converging to a point $\widetilde{\boldsymbol{z}}$ that is *feasible* in DE. Since $\widetilde{\boldsymbol{z}}$ is contained in all nodes n of such sequences, DE^n (and thus SP_s^n) have optimal solutions for each n. Let $\boldsymbol{z}^{\mathrm{DE}^n} = (\boldsymbol{x}^{\mathrm{DE}^n}, \boldsymbol{y}^{\mathrm{DE}^n}) = (\boldsymbol{x}^{\mathrm{DE}^n}, \boldsymbol{y}_1^{\mathrm{DE}^n}, \ldots, \boldsymbol{y}_{N_s}^{\mathrm{DE}^n}) \in \mathcal{Z}^n$ be an optimal solution to DE^n, and define $\boldsymbol{z}_s^{\mathrm{DE}^n} = (\boldsymbol{x}^{\mathrm{DE}^n}, \boldsymbol{y}_s^{\mathrm{DE}^n}) \in \mathcal{Z}_s^n$. Similarly, let $\boldsymbol{z}_s^{\mathrm{SP}^n} = (\boldsymbol{x}_s^{\mathrm{SP}^n}, \boldsymbol{y}_s^{\mathrm{SP}^n}) \in \mathcal{Z}_s^n$ be an optimal solution to SP_s^n. Using the Lipschitz property of f_s, we can immediately express the difference in optimal objective values as:

$$f_{\mathrm{DE}}^n - f_{\mathrm{SP}}^n = \sum_{s \in \mathcal{S}} w_s \left(f_s(\boldsymbol{z}_s^{\mathrm{DE}^n}) - f_s(\boldsymbol{z}_s^{\mathrm{SP}^n}) \right)$$

$$\leq \sum_{s\in\mathcal{S}} w_s C^{\mathrm{SP}}_{f,s}\, \mathrm{W}\left(\mathcal{Z}^n\right)$$

where $C^{\mathrm{SP}}_{f,s} := L_{f,s}\sqrt{N_x + N_y}$. Thus SP^n_s has at least first-order convergence at all feasible points with $C_f = \sum_{s\in\mathcal{S}} w_s C^{\mathrm{SP}}_{f,s}$.

Convergence Order at Infeasible Points: Now consider some nested sequence of nodes converging to a point $\widetilde{\boldsymbol{z}}$ that is *infeasible* in DE, i.e., $\widetilde{\boldsymbol{z}} \notin \mathcal{F}$. By compactness of $\mathcal{F}$, all such sequences eventually reach a node n that does not contain any feasible point, i.e., $\mathrm{vio}^n_{\mathrm{DE}} > 0$. In the following we only consider nodes at or beyond this threshold, since for larger nodes, the existence of a feasible point implies $\mathrm{vio}^n_{\mathrm{DE}} = 0$ by Definition 3, and thus also $\mathrm{vio}^n_{\mathrm{SP}} = 0$ by the fact that SP is a relaxation. Thus for large nodes $\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}} = 0$ and the properties from Definition 4 hold trivially.

Let $\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n} = (\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}, \widetilde{\boldsymbol{y}}^{\mathrm{DE}^n}) = (\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}, \widetilde{\boldsymbol{y}}^{\mathrm{DE}^n}_1, \ldots, \widetilde{\boldsymbol{y}}^{\mathrm{DE}^n}_{N_s}) \in \mathcal{Z}^n$ and $\boldsymbol{\zeta}^{\mathrm{DE}^n} \in \mathbb{R}_-^{N_g^{\mathrm{DE}}}$ be points at which the minimum constraint violation $\mathrm{vio}^n_{\mathrm{DE}}$ is attained, i.e., $\mathrm{vio}^n_{\mathrm{DE}} = \left\| \boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}^{\mathrm{DE}^n} \right\|$. Similarly, let $\widetilde{\boldsymbol{\xi}}^{\mathrm{SP}^n} = (\widetilde{\boldsymbol{x}}^{\mathrm{SP}^n}_1, \widetilde{\boldsymbol{y}}^{\mathrm{SP}^n}_1, \ldots, \widetilde{\boldsymbol{x}}^{\mathrm{SP}^n}_{N_s}, \widetilde{\boldsymbol{y}}^{\mathrm{SP}^n}_{N_s}) \in \times_{s\in\mathcal{S}}(\mathcal{Z}^n_s)$ and $\widetilde{\boldsymbol{\zeta}}^{\mathrm{SP}^n} \in \mathbb{R}_-^{N_g^{\mathrm{SP}}}$ be points at which the minimum constraint violation $\mathrm{vio}^n_{\mathrm{SP}}$ is attained, i.e., $\mathrm{vio}^n_{\mathrm{SP}} = \left\| \boldsymbol{g}_{\mathrm{SP}}(\widetilde{\boldsymbol{\xi}}^{\mathrm{SP}^n}) - \widetilde{\boldsymbol{\zeta}}^{\mathrm{SP}^n} \right\|$. Furthermore, define $\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}_s = (\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}, \widetilde{\boldsymbol{y}}^{\mathrm{DE}^n}_s) \in \mathcal{Z}^n_s$ and $\widetilde{\boldsymbol{z}}^{\mathrm{SP}^n}_s = (\widetilde{\boldsymbol{x}}^{\mathrm{SP}^n}_s, \widetilde{\boldsymbol{y}}^{\mathrm{SP}^n}_s) \in \mathcal{Z}^n_s$.

To derive an upper bound on $\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}}$, we first give a lower bound on the minimum constraint violation $\mathrm{vio}^n_{\mathrm{SP}}$. For this we drop positive terms in the definition of $\mathrm{vio}^n_{\mathrm{SP}}$, corresponding to the first-stage constraints of all but the first scenario:

$$\begin{aligned}
\mathrm{vio}^n_{\mathrm{SP}} &= \left\| \boldsymbol{g}_{\mathrm{SP}}(\widetilde{\boldsymbol{\xi}}^{\mathrm{SP}^n}) - \widetilde{\boldsymbol{\zeta}}^{\mathrm{SP}^n} \right\| \\
&= \left(\sum_{j}^{N_g^{\mathrm{SP}}} \left| g_{\mathrm{SP},j}(\widetilde{\boldsymbol{\xi}}^{\mathrm{SP}^n}) - \widetilde{\zeta}^{\mathrm{SP}^n}_j \right|^2 \right)^{1/2} \\
&= \Big(\left| g_{\mathrm{I},1}(\widetilde{\boldsymbol{x}}^{\mathrm{SP}^n}_1) - \widetilde{\zeta}^{\mathrm{SP}^n}_1 \right|^2 + \cdots + \left| g_{\mathrm{I},N_{\mathrm{I}}}(\widetilde{\boldsymbol{x}}^{\mathrm{SP}^n}_1) - \widetilde{\zeta}^{\mathrm{SP}^n}_{N_{\mathrm{I}}} \right|^2 \\
&\quad + \cdots + \left| g_{\mathrm{I},N_{\mathrm{I}}}(\widetilde{\boldsymbol{x}}^{\mathrm{SP}^n}_{N_s}) - \widetilde{\zeta}^{\mathrm{SP}^n}_{N_s N_{\mathrm{I}}} \right|^2 \\
&\quad + \left| g_{\mathrm{II},1,1}(\widetilde{\boldsymbol{z}}^{\mathrm{SP}^n}_1) - \widetilde{\zeta}^{\mathrm{SP}^n}_{N_s N_{\mathrm{I}}+1} \right|^2 + \cdots + \left| g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\widetilde{\boldsymbol{z}}^{\mathrm{SP}^n}_{N_s}) - \widetilde{\zeta}^{\mathrm{SP}^n}_{N_g^{\mathrm{SP}}} \right|^2 \Big)^{1/2} \\
&\geq \Big(\left| g_{\mathrm{I},1}(\widetilde{\boldsymbol{x}}^{\mathrm{SP}^n}_1) - \widetilde{\zeta}^{\mathrm{SP}^n}_1 \right|^2 + \cdots + \left| g_{\mathrm{I},N_{\mathrm{I}}}(\widetilde{\boldsymbol{x}}^{\mathrm{SP}^n}_1) - \widetilde{\zeta}^{\mathrm{SP}^n}_{N_{\mathrm{I}}} \right|^2 \\
&\quad + \left| g_{\mathrm{II},1,1}(\widetilde{\boldsymbol{z}}^{\mathrm{SP}^n}_1) - \widetilde{\zeta}^{\mathrm{SP}^n}_{N_s N_{\mathrm{I}}+1} \right|^2 + \cdots + \left| g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\widetilde{\boldsymbol{z}}^{\mathrm{SP}^n}_{N_s}) - \widetilde{\zeta}^{\mathrm{SP}^n}_{N_g^{\mathrm{SP}}} \right|^2 \Big)^{1/2} \\
&=: \left\| \widetilde{\boldsymbol{g}}^{\mathrm{SP}^n} - \boldsymbol{\zeta}^{\mathrm{SP}^n} \right\|
\end{aligned} \tag{2}$$

Note that this corresponds to a projection of the associated points from $\mathbb{R}^{N_g^{\mathrm{SP}}}$ onto $\mathbb{R}^{N_g^{\mathrm{DE}}}$, i.e., $\widetilde{\boldsymbol{g}}^{\mathrm{SP}^n}, \boldsymbol{\zeta}^{\mathrm{SP}^n} \in \mathbb{R}^{N_g^{\mathrm{DE}}}$.

We can now derive the desired upper bound on $\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}}$. By Definition 3, we have

$$\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}} = \left\| \boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}^{\mathrm{DE}^n} \right\| - \left\| \boldsymbol{g}_{\mathrm{SP}}(\widetilde{\boldsymbol{\xi}}^{\mathrm{SP}^n}) - \widetilde{\boldsymbol{\zeta}}^{\mathrm{SP}^n} \right\|,$$

and underestimation of the subtracted part by the projection from 2 gives

$$\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}} \leq \left\| \boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}^{\mathrm{DE}^n} \right\| - \left\| \widetilde{\boldsymbol{g}}^{\mathrm{SP}^n} - \boldsymbol{\zeta}^{\mathrm{SP}^n} \right\|,$$

By definition of the infimum in $\mathrm{vio}^n_{\mathrm{DE}}$, we have $\left\| \boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}^{\mathrm{DE}^n} \right\| \leq \left\| \boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta} \right\|$ for all $\boldsymbol{\zeta} \in \mathbb{R}_-^{N_g^{\mathrm{DE}}}$, in particular, choosing $\boldsymbol{\zeta}^{\mathrm{SP}^n}$ results in

$$\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}} \leq \left\| \boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}^{\mathrm{SP}^n} \right\| - \left\| \widetilde{\boldsymbol{g}}^{\mathrm{SP}^n} - \boldsymbol{\zeta}^{\mathrm{SP}^n} \right\|.$$

Applying the reverse triangle inequality gives

$$\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}} \leq \left\| \boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \widetilde{\boldsymbol{g}}^{\mathrm{SP}^n} \right\|,$$

and thus by definition of the Euclidian norm and $\widetilde{\boldsymbol{g}}^{\mathrm{SP}^n}$:

$$\begin{aligned}\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}} \leq \Bigg(& \left| g_{\mathrm{I},1}(\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}) - g_{\mathrm{I},1}(\widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n}) \right|^2 + \cdots + \left| g_{\mathrm{I},N_{\mathrm{I}}}(\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}) - g_{\mathrm{I},N_{\mathrm{I}}}(\widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n}) \right|^2 \\ & + \left| g_{\mathrm{II},1,1}(\widetilde{\boldsymbol{z}}_1^{\mathrm{DE}^n}) - g_{\mathrm{II},1,1}(\widetilde{\boldsymbol{z}}_1^{\mathrm{SP}^n}) \right|^2 \\ & + \cdots + \left| g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{DE}^n}) - g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{SP}^n}) \right|^2 \Bigg)^{1/2}.\end{aligned}$$

By Lipschitz continuity of each individual constraint function, all differences can be bounded by the respective Lipschitz constants

$$\begin{aligned}\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}} \leq \Bigg(& \left(L_{g_{\mathrm{I},1}} \left\| \widetilde{\boldsymbol{x}}^{\mathrm{DE}^n} - \widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n} \right\| \right)^2 + \cdots + \left(L_{g_{\mathrm{I},N_{\mathrm{I}}}} \left\| \widetilde{\boldsymbol{x}}^{\mathrm{DE}^n} - \widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n} \right\| \right)^2 \\ & + \left(L_{g_{\mathrm{II},1,1}} \left\| \widetilde{\boldsymbol{z}}_1^{\mathrm{DE}^n} - \widetilde{\boldsymbol{z}}_1^{\mathrm{SP}^n} \right\| \right)^2 \\ & + \cdots + \left(L_{g_{\mathrm{II},N_s,N_{\mathrm{II}}}} \left\| \widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{DE}^n} - \widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{SP}^n} \right\| \right)^2 \Bigg)^{1/2}.\end{aligned}$$

Finally, since the maximum distances of points in $\mathcal{X}^n$ and $\mathcal{Z}^n$ are $\sqrt{N_x}\,\mathrm{W}(\mathcal{X}^n)$ and $\sqrt{N_x + N_y}\,\mathrm{W}(\mathcal{Z}_s^n)$, respectively, and since both $\mathrm{W}(\mathcal{X}^n)$ and $\mathrm{W}(\mathcal{Z}_s^n)$ can be overestimated by $\mathrm{W}(\mathcal{Z}^n)$ we have:

$$\begin{aligned}\mathrm{vio}^n_{\mathrm{DE}} - \mathrm{vio}^n_{\mathrm{SP}} &\leq \Bigg(\left(L_{g_{\mathrm{I},1}} \sqrt{N_x}\,\mathrm{W}(\mathcal{X}^n) \right)^2 + \cdots + \left(L_{g_{\mathrm{I},N_{\mathrm{I}}}} \sqrt{N_x}\,\mathrm{W}(\mathcal{X}^n) \right)^2 \\ &\quad + \left(L_{g_{\mathrm{II},1,1}} \sqrt{N_x + N_y}\,\mathrm{W}(\mathcal{Z}_s^n) \right)^2 \\ &\quad + \cdots + \left(L_{g_{\mathrm{II},N_s,N_{\mathrm{II}}}} \sqrt{N_x + N_y}\,\mathrm{W}(\mathcal{Z}_s^n) \right)^2 \Bigg)^{1/2} \\ &\leq C_g\,\mathrm{W}(\mathcal{Z}^n)\end{aligned}$$

Thus SP_s^n has at least first-order convergence at all infeasible points with

$$C_g = \sqrt{N_x \sum_{i=1}^{N_{\mathrm{I}}} L_{g,\mathrm{I},i}^2 + (N_x + N_y) \sum_{s \in \mathcal{S}} \sum_{j=1}^{N_{\mathrm{II}}} L_{g_{\mathrm{II},s,j}}^2}.$$

Conclusion: As the LBS based on SP_s^n has convergence orders of $\beta \geq 1$ at both feasible and infeasible points, it has convergence order of $\beta \geq 1$. □

Unsurprisingly, when the Assumption 1 is not satisfied, the convergence order of MUSE-BB can also be below 1. For instance, take Example 1 but use $w_1 f_1 = -\sqrt{y_1}$; this gives a convergence order of 0.5.

Next we show that both the McCormick based LBS, MC_s^n, as well as its linearization via subtangents, LP_s^n, inherit the first-order convergence of SP_s^n under mild additional assumptions. For both of these convergence results, we require the following assumption:

Assumption 2 (*first-order pointwise convergent relaxations*) The objective function f_s and all elements of the constraint functions $\boldsymbol{g}_{\mathrm{I}}$ and $\boldsymbol{g}_{\mathrm{II},s}$ have first-order *pointwise convergent* relaxations, i.e., there exist constants $C_{f,s}^{\mathrm{MC}} > 0, s \in \mathcal{S}$, $C_{g,\mathrm{I},i}^{\mathrm{MC}} > 0, i = 1, \ldots, N_{\mathrm{I}}$, and $C_{g,\mathrm{II},s,j}^{\mathrm{MC}} > 0, s \in \mathcal{S}, j = 1, \ldots, N_{\mathrm{II}}$, such that for all $\mathcal{Z}^n \subset \mathcal{Z}$ and any s, the convex relaxations $f_s^{\mathrm{cv},n}$, $\boldsymbol{g}_{\mathrm{I}}^{\mathrm{cv},n}$ and $\boldsymbol{g}_{\mathrm{II},s}^{\mathrm{cv},n}$ in MC_s^n satisfy

$$
\begin{aligned}
f_s(z_s) - f_s^{\mathrm{cv},n}(z_s) &\leq C_{f,s}^{\mathrm{MC}} \,\mathrm{W}\left(\mathcal{Z}_s^n\right), \quad \forall z_s \in \mathcal{Z}_s^n, \\
g_{\mathrm{I},i}(\boldsymbol{x}) - g_{\mathrm{I},i}^{\mathrm{cv},n}(\boldsymbol{x}) &\leq C_{g,\mathrm{I},i}^{\mathrm{MC}} \,\mathrm{W}\left(\mathcal{X}^n\right), \quad \forall \boldsymbol{x} \in \mathcal{X}^n,\ i = 1, \ldots N_{\mathrm{I}}, \\
g_{\mathrm{II},s,j}(z_s) - g_{\mathrm{II},s,j}^{\mathrm{cv},n}(z_s) &\leq C_{g,II,s,j}^{\mathrm{MC}} \,\mathrm{W}\left(\mathcal{Z}_s^n\right), \quad \forall z_s \in \mathcal{Z}_s^n,\ j = 1, \ldots N_{\mathrm{II}}.
\end{aligned}
$$

In fact, for many functions McCormick relaxations satisfying an even stronger variant of Assumption 2, with second- instead of just first-order pointwise convergence are known [also see 29]. For our purposes, however, Assumption 2 is sufficient.

Corollary 1 (first-order convergence of MC_s^n) *Under Assumptions 1 and 2, MC_s^n has a convergence order of $\beta \geq 1$.*

Proof By Lemma 2, the scheme SP_s^n has first-order convergence with respect to the original problem DE^n. Furthermore, under Assumption 2, the LBS MC_s^n has at least first-order convergence with respect to SP_s^n by Theorem 1 of [55]. Combining these results implies first-order convergence of MC_s^n with respect to DE^n. □

For the first-order convergence of LP_s^n, we additionally require the following assumption:

Assumption 3 (*Lipschitz convex relaxations*) For any node n the convex relaxations $f_s^{\mathrm{cv},n}$, $\boldsymbol{g}_{\mathrm{I}}^{\mathrm{cv},n}$, and $\boldsymbol{g}_{\mathrm{II},s}^{\mathrm{cv},n}$ in MC_s^n are Lipschitz, i.e., there exist constants $L_{f,s}^{\mathrm{MC}} > 0$, $L_{g,\mathrm{I},i}^{\mathrm{MC}} > 0$; $i = 1, \ldots, N_{\mathrm{I}}$, and $L_{g,\mathrm{II},s,j}^{\mathrm{MC}}$, $j = 1, \ldots, N_{\mathrm{II}}$, that constitute upper bounds on the norm of the respective subgradients. In particular, this implies:

$$
\begin{aligned}
\left\| \check{\nabla} f_s^{\mathrm{cv},n}(z_s)^{\intercal}(z_s' - z_s) \right\| &\leq L_{f,s}^{\mathrm{MC}} \sqrt{N_x + N_y} \,\mathrm{W}\left(\mathcal{Z}_s^n\right), \quad \forall z_s, z_s' \in \mathcal{Z}_s^n \\
\left\| \check{\nabla} g_{\mathrm{I},i}^{\mathrm{cv},n}(\boldsymbol{x})^{\intercal}(\boldsymbol{x}' - \boldsymbol{x}) \right\| &\leq L_{g,\mathrm{I},i}^{\mathrm{MC}} \sqrt{N_x} \,\mathrm{W}\left(\mathcal{X}^n\right), \quad \forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}^n,\ i = 1, \ldots N_{\mathrm{I}}, \\
\left\| \check{\nabla} g_{\mathrm{II},s,j}^{\mathrm{cv},n}(z_s)^{\intercal}(z_s' - z_s) \right\| &\leq L_{g,\mathrm{II},s,j}^{\mathrm{MC}} \sqrt{N_x + N_y} \,\mathrm{W}\left(\mathcal{Z}_s^n\right), \quad \forall z_s, z_s' \in \mathcal{Z}_s^n,\ j = 1, \ldots N_{\mathrm{II}}.
\end{aligned}
$$

Assumption 3 is satisfied if the relaxations used for all intrinsic functions are Lipschitz [cf. 61]. This in turn is the case for standard relaxations of a wide class of functions, provided they are Lipschitz themselves.

Corollary 2 (first-order convergence of LP_s^n) *Under Assumptions 1–3, LP_s^n has a convergence order of $\beta \geq 1$.*

Proof We structure the proof as in Lemma 2.

Convergence Order at Feasible Points: First consider some nested sequence of nodes converging to a point $\widetilde{z}$ that is feasible in DE. For all nodes n of such sequences let $z_s^{\mathrm{DE}^n}$ and $z_s^{\mathrm{LP}^n}$ be solutions of DE^n, and LP_s^n, respectively, and note that

$$f_{\mathrm{LP},s}^n = \mathrm{sub}_{f_s}^n(z_s^{\mathrm{LP}^n}) = f_s^{\mathrm{cv},n}(\boldsymbol{m}_{z_s}^n) + \check{\nabla} f_s^{\mathrm{cv},n}(\boldsymbol{m}_{z_s}^n)^\intercal \left(z_s^{\mathrm{LP}^n} - \boldsymbol{m}_{z_s}^n\right),$$

where $\boldsymbol{m}_{z_s}^n$ is the midpoint of $\mathcal{Z}_s^n$, see subtangent. We can bound the difference of optimal values of DE^n, and LP_s^n by subtracting and adding the terms $f_s(\boldsymbol{m}_{z_s}^n)$ and applying Assumptions 1–3:

$$\begin{aligned}
f_{\mathrm{DE}}^n - f_{\mathrm{LP}}^n &= \sum_{s\in\mathcal{S}} w_s \left(f_{\mathrm{DE},s}^n - f_{\mathrm{LP},s}^n\right) = \sum_{s\in\mathcal{S}} w_s \left(f_s(z_s^{\mathrm{DE}^n}) - \mathrm{sub}_{f_s}^n(z_s^{\mathrm{LP}^n})\right) \\
&= \sum_{s\in\mathcal{S}} w_s \Big(f_s(z_s^{\mathrm{DE}^n}) - f_s(\boldsymbol{m}_{z_s}^n) + f_s(\boldsymbol{m}_{z_s}^n) - f_s^{\mathrm{cv},n}(\boldsymbol{m}_{z_s}^n) \\
&\quad - \check{\nabla} f_s^{\mathrm{cv},n}(\boldsymbol{m}_{z_s}^n)^\intercal \left(z_s^{\mathrm{LP}^n} - \boldsymbol{m}_{z_s}^n\right)\Big) \\
&\le \sum_{s\in\mathcal{S}} w_s C_{f,s}^{\mathrm{LP}} \,\mathrm{W}\left(\mathcal{Z}^n\right)
\end{aligned}$$

where $C_{f,s}^{\mathrm{LP}} := \left((L_{f,s} + L_{f,s}^{\mathrm{MC}})\sqrt{N_x + N_y} + C_{f,s}^{\mathrm{MC}}\right)$. Thus LP_s^n has first-order convergence at feasible points with $C_f = \sum_{s\in\mathcal{S}} w_s C_{f,s}^{\mathrm{LP}}$.

Convergence Order at Infeasible Points: Now consider some sequence of nodes converging to an infeasible point. As in the proof of Lemma 2, let $\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n} = (\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}, \widetilde{\boldsymbol{y}}^{\mathrm{DE}^n}) = (\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}, \widetilde{\boldsymbol{y}}_1^{\mathrm{DE}^n}, \ldots, \widetilde{\boldsymbol{y}}_{N_s}^{\mathrm{DE}^n}) \in \mathcal{Z}^n$, and $\boldsymbol{\zeta}^{\mathrm{DE}^n} \in \mathbb{R}_-^{N_g^{\mathrm{DE}}}$ be points at which the minimum constraint violation $\mathrm{vio}_{\mathrm{DE}}^n$ is attained, i.e., $\mathrm{vio}_{\mathrm{DE}}^n = \left\|\boldsymbol{g}(\widetilde{\boldsymbol{z}}^{\mathrm{DE}^n}) - \boldsymbol{\zeta}^{\mathrm{DE}^n}\right\|$, and, let $\widetilde{\boldsymbol{\xi}}^{\mathrm{LP}^n} = (\widetilde{\boldsymbol{x}}_1^{\mathrm{LP}^n}, \widetilde{\boldsymbol{y}}_1^{\mathrm{LP}^n}, \ldots, \widetilde{\boldsymbol{x}}_{N_s}^{\mathrm{LP}^n}, \widetilde{\boldsymbol{y}}_{N_s}^{\mathrm{LP}^n}) \in \times_{s\in\mathcal{S}}(\mathcal{Z}_s^n)$ and $\widetilde{\boldsymbol{\zeta}}^{\mathrm{LP}^n} \in \mathbb{R}_-^{N_g^{\mathrm{LP}}}$ be points at which the minimum constraint violation $\mathrm{vio}_{\mathrm{LP}}^n$ is attained, i.e., $\mathrm{vio}_{\mathrm{LP}}^n = \left\|\boldsymbol{g}_{\mathrm{LP}}(\widetilde{\boldsymbol{\xi}}^{\mathrm{LP}^n}) - \widetilde{\boldsymbol{\zeta}}^{\mathrm{LP}^n}\right\|$, where $\boldsymbol{g}_{\mathrm{LP}}$ is the vector-valued function containing the constraints of all LP_s^n, i.e., the subtangents of the entries in $\boldsymbol{g}_{\mathrm{SP}}$, see subtangent.

Using the same arguments as in the proof of Lemma 2 with $\boldsymbol{g}_{\mathrm{LP}}(\widetilde{\boldsymbol{\xi}}^{\mathrm{LP}^n})$ instead of $\boldsymbol{g}_{\mathrm{SP}}(\widetilde{\boldsymbol{\xi}}^{\mathrm{SP}^n})$ we can bound the difference in violation measures of DE^n and LP_s^n, resulting in:

$$\begin{aligned}
\mathrm{vio}_{\mathrm{DE}}^n - \mathrm{vio}_{\mathrm{LP}}^n \le \Big(&\left|g_{\mathrm{I},1}(\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}) - \mathrm{sub}_{g_{\mathrm{I},1}}^n(\widetilde{\boldsymbol{x}}_1^{\mathrm{LP}^n})\right|^2 \\
&+ \cdots + \left|g_{\mathrm{I},N_{\mathrm{I}}}(\widetilde{\boldsymbol{x}}^{\mathrm{DE}^n}) - \mathrm{sub}_{g_{\mathrm{I},N_{\mathrm{I}}}}^n(\widetilde{\boldsymbol{x}}_1^{\mathrm{SP}^n})\right|^2 \\
&+ \left|g_{\mathrm{II},1,1}(\widetilde{\boldsymbol{z}}_1^{\mathrm{DE}^n}) - \mathrm{sub}_{g_{\mathrm{II},1,1}}^n(\widetilde{\boldsymbol{z}}_1^{\mathrm{SP}^n})\right|^2 \\
&+ \cdots + \left|g_{\mathrm{II},N_s,N_{\mathrm{II}}}(\widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{DE}^n}) - \mathrm{sub}_{g_{\mathrm{II},N_s,N_{\mathrm{II}}}}^n(\widetilde{\boldsymbol{z}}_{N_s}^{\mathrm{SP}^n})\right|^2 \Big)^{1/2}
\end{aligned}$$

as with the objective function, we can bound the differences between each constraint function and the respective subgradient, using Assumptions 1–3, which results in

$$\mathrm{vio}_{\mathrm{DE}}^n - \mathrm{vio}_{\mathrm{LP}}^n \le C_g^{\mathrm{LP}} \,\mathrm{W}\left(\mathcal{Z}^n\right),$$

where

$$C_g^{\mathrm{LP}} := \Bigg(\sum_{i=1}^{N_{\mathrm{I}}} \left((L_{g,\mathrm{I},i} + L_{g,\mathrm{I},i}^{\mathrm{MC}})\sqrt{N_x} + C_{g,\mathrm{I},i}^{\mathrm{MC}} \right)^2 + \sum_{s\in\mathcal{S}} \sum_{j=1}^{N_{\mathrm{II}}} \left((L_{g,\mathrm{II},s,j} + L_{g,\mathrm{II},s,j}^{\mathrm{MC}})\sqrt{N_x + N_y} + C_{g,\mathrm{II},s,j}^{\mathrm{MC}} \right)^2 \Bigg)^{1/2}.$$

Thus LP_s^n has first-order convergence at any infeasible point with $C_g = C_g^{\mathrm{LP}}$.

Conclusion: As the LBS based on LP_s^n has convergence orders of $\beta \geq 1$ at both feasible and infeasible points, it has convergence order of $\beta \geq 1$. □

We are now in the position to prove finite ε_f-convergence of MUSE-BB.

Corollary 3 (finite termination of MUSE-BB) *Under Assumptions 1–3, MUSE-BB terminates finitely for any optimality tolerance $\varepsilon_f > 0$, either providing an ε_f-optimal solution or a certificate that the problem is infeasibile.*

Proof By Lemma 1, each sequence of descendant nodes converges to some accumulation point $\widetilde{z}$. We show that the use of any LBS R_s^n with convergence order of $\beta > 0$ implies that all such sequences finitely reach a node that can be fathomed by value dominance or infeasibility.

Convergence at Feasible Points: First consider sequences for which $\widetilde{z}$ is feasible. After a finite number of iterations, any such sequence will produce a node n, for which $\mathrm{W}(Z^n) \leq \left(\frac{\varepsilon_f}{C_f}\right)^{1/\beta}$, which implies that $f_{\mathrm{DE}}^n - f_{\mathrm{R}}^n \leq \varepsilon_f$, i.e., that n is fathomed by value dominance.

Convergence at Infeasible Points: Next consider sequences for which $\widetilde{z}$ is infeasible, and which are not terminated finitely because some descendant node can be fathomed by value dominance. By compactness of the feasible set, any such sequence will eventually produce a node $\widetilde{n}$ that contains no feasible point, and thus has a positive violation measure $\mathrm{vio}_{\mathrm{DE}}^n[\widetilde{n}]$. Since the violation measure increases monotonically for descendants of node $\widetilde{n}$, the sequence is terminated when or before the descendant node n is produced, for which $\mathrm{W}(Z^n) \leq \left(\frac{\mathrm{vio}_{\mathrm{DE}}^n[\widetilde{n}]}{C_g}\right)^{1/\beta}$, as this implies $0 \leq \mathrm{vio}_{\mathrm{DE}}^n - \mathrm{vio}_{\mathrm{DE}}^n[\widetilde{n}] \leq \mathrm{vio}_{\mathrm{R}}^n$, i.e., infeasibility is detected by the scheme R_s^n, and node n is fathomed by infeasibility.

Conclusion: In summary, each node sequence terminates finitely and since the original domain is compact, the total number of sequences must be finite. By Corollary 2, the assumptions imply that the LBS $\mathrm{R}_s^n = \mathrm{LP}_s^n$, used in MUSE-BB has a convergence order of $\beta > 1$, thus MUSE-BB terminates finitely, once all sequences of descendant nodes are terminated. □

After demonstrating first-order convergence of the LBS employed by MUSE-BB and the resulting ε_f-convergence, we now consider in which cases these convergence properties may be sufficient to mitigate clustering. As indicated by [28], clustering may be mitigated around individual minimizers of DE, if the convergence order of the LBS is larger or equal to the order at which objective and constraint functions grow around this minimizer. While Example 2 demonstrates that SP_s^n (and by extension, also LP_s^n) may have a convergence order as low as one at *constrained minimizers*, objective and constraint functions often grow at a linear rate around such points [28]. Therefore LP_s^n may mitigate clustering around certain constrained minimizers, provided the respective coefficients C_f and C_g are sufficiently small [28]. On the other hand, at *partially* or *unconstrained minimizers*, where f is differentiable, f grows

quadratically or faster in some of the feasible directions. As a result, a LBS needs to have at least second-order convergence at unconstrained minimizers to to mitigate clustering [26–28]. Unfortunately, the convergence order of SP_s^n may also be as low as one at unconstrained minimizers, as shown by the following example.

Example 3 Consider an instance of DE with $N_x = 1, N_y = 0, N_s = 2$ and an original domain $\mathcal{X} = [-1, 1]$. Take

$$w_1 f_1(x_1) = 0.5(x_1 - 1)^2; \qquad w_2 f_2(x_2) = 0.5(x_2 + 1)^2$$

such that $f(x) = x^2 + 1$, and thus the optimal solution and objective value are $x^{\text{DE}} = 0$, and $f(x^{\text{DE}}) = 1$, respectively. For any nested sequence of nodes converging to this optimum, the solutions x^{DE^n} of the node problem DE^n lie in $\mathcal{X}^n = [\underline{x}^n, \overline{x}^n]$, and thus $\underline{x}^n \leq 0, \overline{x}^n \geq 0$. For such nodes, the solutions of SP_s^n are $x_1^{\text{SP}^n} = \overline{x}^n$, and $x_2^{\text{SP}^n} = \underline{x}^n$, respectively. Hence the difference in objective values is:

$$\begin{aligned} f_{\text{DE}}^n - f_{\text{SP}}^n &= 1 - 0.5\left((\overline{x}^n - 1)^2 + (\underline{x}^n + 1)^2\right) \\ &= -0.5\,(\overline{x}^n)^2 + \overline{x}^n - \underline{x}^n - 0.5\,(\underline{x}^n)^2 \end{aligned}$$

Now consider a sequence for which $\overline{x}^n = W^n, \underline{x}^n = 0$; for this sequence the above expression simplifies to

$$f_{\text{DE}}^n - f_{\text{SP}}^n = W^n - 0.5\left(W^n\right)^2.$$

Now for any $C_f > 0$ this expression becomes larger than $C_f\,(W^n)^2$ for the node n_0, for which

$$W^{n_0} < \frac{1}{C_f + 0.5},$$

i.e., SP_s^n is at best first-order convergent at the unconstrained minimizer x^{DE^n}.

In summary, the present implementation of MUSE-BB may suffer from clustering around unconstrained minimizers. To address this, an alternative LBS with at least quadratic convergence order is required. In the following section we analyze an extension of MUSE-BB whose LBS has this property.

5.3 Second-order convergence

In this section we show that using LSP_s^n instead of SP_s^n, i.e., dualizing the NACs instead of dropping them, enables at least second-order convergence at unconstrained minimizers. Additionally, we consider the resulting effect on the implementation, i.e., how the LBS LP_s^n needs to be adapted when using LSP_s^n.

A necessary condition for a LBS to have β-order convergence is that the relaxations used for its construction have β-order convergence, also see [55]. While this condition is generally not sufficient for β-order convergence of the resulting LBS, it is sufficient for β-order convergence around *Slater points, i.e., unconstrained feasible points* [Corollaries 2, 3 of 55].

[Corollary 6 of 24]shows that the optimal objective value $f_{\text{LR}}^{\mathcal{X}^n,\mathcal{Y}}(\boldsymbol{\lambda}^*)$, obtained from the subproblems $\text{LSP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$, where the NACs are dualized instead of dropped, is equivalent to minimizing the w_s-weighted sum of convex envelopes of $f_s^{\mathcal{X}^n,\mathcal{Y}}$. As a result, $f_{\text{LR}}^{\mathcal{X}^n,\mathcal{Y}}(\boldsymbol{\lambda}^*)$ constitutes a (constant valued) relaxation of the objective function f on the domain $\mathcal{X}^n \times \mathcal{Y}$.

Furthermore, they show that this relaxation is at least second-order convergent with respect to $\mathrm{W}(\mathcal{X}^n)$, i.e.,

$$\min_{\boldsymbol{x}\in\mathcal{X}^n} \sum_{s\in\mathcal{S}} f_s^{\mathcal{X}^n,\mathcal{Y}_s}(\boldsymbol{x}) - f_{\mathrm{LR}}^{\mathcal{X}^n,\mathcal{Y}}\left(\boldsymbol{\lambda}^*\right) \leq \tau\, \mathrm{W}\left(\mathcal{X}^n\right)^{\beta}$$

with $\beta \geq 2$, provided the scenario value functions $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ are $\mathcal{C}^2$, i.e., twice continuously differentiable. We point out that in fact, the slightly weaker assumption that f merely has bounded second-order directional derivatives, i.e., that it is $\mathcal{C}^{1,1}$, is also sufficient for second-order convergence of $f_{\mathrm{LR}}^{\mathcal{X}^n,\mathcal{Y}}\left(\boldsymbol{\lambda}^*\right)$, also see [62]. In the special case where the $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ are convex, β above may take any positive value, i.e., the convergence is arbitrarily high. Note that β-order convergence of $f_{\mathrm{LR}}^{\mathcal{X}^n,\mathcal{Y}}\left(\boldsymbol{\lambda}^*\right)$ immediately implies β-order convergence of the LBS $\mathrm{LSP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$ at unconstrained feasible points (and in particular at unconstrained minimizers), because around such points f_{DE}^n, i.e., the optimal value of DE^n, is equivalent to $\min_{\boldsymbol{x}\in\mathcal{X}^n} \sum_{s\in\mathcal{S}} f_s^{\mathcal{X}^n,\mathcal{Y}_s}(\boldsymbol{x})$, also see Corollaries 2 and 3 of [55]. Furthermore, β-order convergence with respect to $\mathrm{W}(\mathcal{X}^n)$ implies β-order convergence with respect to $\mathrm{W}(\mathcal{Z}^n)$, since $\mathrm{W}(\mathcal{X}^n) \leq \mathrm{W}(\mathcal{Z}^n)$. As a result, the same line of argument naturally also holds for the scheme $\mathrm{LSP}_s^n := \mathrm{LSP}_s^{\mathcal{X}^n,\mathcal{Y}_s^n}$, which for any $\mathcal{X}^n$ produces stronger bounds than $\mathrm{LSP}_s^{\mathcal{X}^n,\mathcal{Y}_s}$. Hence, the relaxations $f_{\mathrm{LR}}^{\mathcal{X}^n,\mathcal{Y}^n}\left(\boldsymbol{\lambda}^*\right)$ are at least second-order convergent, and Corollaries 2 and 3 of [55] ensure second-order convergence of LSP_s^n at unconstrained feasible points. The following example demonstrates the improvement of convergence order of LSP_s^n over SP_s^n.

Example 4 Take the problem from Example 3. The optimal dual values for this problem are $\boldsymbol{\lambda}_s^* = (2, -2)$, such that the objectives of LSP_s^n are $f_s(x_s) + \lambda_s x_s = (x_s \mp 1)^2 \pm 2x_s = x_s^2 + 1$. Hence, both subproblems are solved at $x_1^{\mathrm{LSP}^n} = x_2^{\mathrm{LSP}^n} = x^{\mathrm{DE}^n} = 0$, and the difference in objective values is:

$$f_{\mathrm{DE}}^n - f_{\mathrm{LSP}}^n = 1 - 0.5\left((0+1)^2 + (0+1)^2\right) = 0,$$

i.e., LSP_s^n is exact and as such has arbitrarily high convergence order at the unconstrained minimizer x^{DE^n}.

Note that the arbitrarily high convergence order in Example 4 results from the fact that the scenario value functions $f_s^{\mathcal{X},\mathcal{Y}_s}$ are convex. If $f_s^{\mathcal{X},\mathcal{Y}_s}$ are not convex, at least second-order convergence is guaranteed by the previous arguments.

Several results from nonlinear parametric programming provide different regularity conditions under which $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ are $\mathcal{C}^2$. In particular, if we assume f is $\mathcal{C}^2$, and that the second-order sufficient condition (SOSC):

$$\begin{aligned} \nabla f(\boldsymbol{z}^{\mathrm{DE}}) &= \boldsymbol{0} \\ \nabla^2 f(\boldsymbol{z}^{\mathrm{DE}}) &\succ 0 \end{aligned} \qquad (\mathrm{SOSC}(\boldsymbol{z}^{\mathrm{DE}}))$$

holds at an unconstrained minimizer $\boldsymbol{z}^{\mathrm{DE}} = (\boldsymbol{x}^{\mathrm{DE}}, \boldsymbol{y}^{\mathrm{DE}})$ of DE, the fact that $f_s^{\mathcal{X}^n,\mathcal{Y}_s}$ are $\mathcal{C}^2$ follows from the Implicit Function Theorem [63, cf., e.g., Corollary 3.2.3].

Other variants of the Implicit Function Theorem provide similar results for unconstrained minimizers that do not satisfy $\mathrm{SOSC}(\boldsymbol{z}^{\mathrm{DE}})$, e.g., [Theorem 3.3 of 64], or even for constrained minimizers, satisfying certain regularity conditions, related to the growth of the Lagrangian of f, e.g., [63] and [65].

We next show how the stronger convergence properties of the LBS LSP_s^n can be incorporated into MUSE-BB via an adaption of the lower bounding problems subproblems LP_s^n.

Recall that LP_s^n result from three subsequent levels of relaxation: after dropping the NACs from $\mathrm{DE}_{\mathrm{NAC}}^{\mathcal{X},\mathcal{Y}}$ (i), the resulting subproblems SP_s^n are further relaxed via McCormick's method (ii) and outer approximation (iii), resulting in the linear lower bounding problems LP_s^n. In this context, dualizing the NACs, corresponds to replacing the subproblems SP_s^n with LSP_s^n, and performing the subsequent relaxations. Note that the only difference between SP_s^n and LSP_s^n are the additional terms $\boldsymbol{\lambda}_s{}^{\mathsf{T}}\boldsymbol{x}_s$. The McCormick relaxation of the sum of the original, nonlinear objective $f_s(\boldsymbol{x}_s, \boldsymbol{y}_s)$, and the linear term $\boldsymbol{\lambda}_s{}^{\mathsf{T}}\boldsymbol{x}_s$ is simply $f_s^{\mathrm{cv},n}(\boldsymbol{x}_s, \check{\boldsymbol{y}}_s) + \boldsymbol{\lambda}_s{}^{\mathsf{T}}\boldsymbol{x}_s$ [cf. Proposition 2 of 29]. Next we consider the subtangents of these terms: if $\check{\nabla} f_s^{\mathrm{cv},n}$ is the subgradient of $f_s^{\mathrm{cv},n}$, used in the original instance of LP_s^n, then $\check{\nabla} f_s^{\mathrm{cv},n} + \boldsymbol{\lambda}_s$ is a valid subgradient of $f_s^{\mathrm{cv},n}(\boldsymbol{x}_s, \boldsymbol{y}_s) + \boldsymbol{\lambda}_s{}^{\mathsf{T}}\boldsymbol{x}_s$ [cf. Proposition 2.3.3 of 66]. As a result, replacing SP_s^n with LSP_s^n in MUSE-BB is equivalent to adding $\boldsymbol{\lambda}_s$ to the coefficients of $\boldsymbol{x}_s$ in the first set of constraints, of the subproblems LP_s^n.

While conceptually, the multipliers can be updated by performing dual iterations with these modified linear lower bounding subproblems, such updates will generally not converge to the optimal multipliers of the original problem LSP_s^n. Even though convergence over a sequence of nodes can be expected, as the node size diminishes and the McCormick relaxations, and linear relaxations converge towards the original functions, such a conversion in the limit may not be sufficient to yield second-order convergence of the resulting lower bounding scheme.

In summary, similar to PBDAs, the LBS used in MUSE-BB may be made second-order convergent at certain minimizers by dualizing the NACs instead of dropping them. However, the use of optimal dual multipliers $\boldsymbol{\lambda}^*$ appears to be a requirement for second-order convergence, and, as already pointed out in Sect. 4.1, obtaining such multipliers is generally very challenging. The fact that SP_s^n can be interpreted as an instance of LSP_s^n with the suboptimal multipliers $\boldsymbol{\lambda} = \mathbf{0}$, indicates that suboptimal multipliers may result in a first-order convergent LBS, also see the related result on a Lagrangian dual-based LBS for general nonlinear programming problems in [Theorem 6 of 55]. While it may be sufficient to limit multiplier updates to small nodes suspected to contain the neighborhoods of critical minimizers, we leave the investigation of such approaches for future work.

6 Computational results

We now present computational results obtained with the parallelized decomposition algorithm MUSE-BB, and outline how it compares against solving the deterministic equivalent formulation $\mathrm{DE}^{\mathcal{X},\mathcal{Y}}$ with the standard version of MAiNGO. MUSE-BB performs upper bounding based on the subproblems SP_s^n, and OBBT, lower bounding, and DBBT, based on the separable subproblems LP_s^n. All scenario subproblems are solved simultaneously, using one thread per scenario. MAiNGO performs upper bounding based on DE^n and all other routines based on a linearization of DE^n, using a single thread.

We do not compare with other deterministic global solvers as these generally employ different routines for management of the B&B tree, generating relaxations of individual functions, and solving individual lower and upper bounding problems, distorting the effect of the decomposition. Further, we focus our computational experiments on the effects of individual algorithm parameters, rather than conducting larger-scale computational studies, as the latter would require access to a library of two-stage test problems, which is currently unavailable. While previous works do consider some larger-scale problems, the implementations are either unpublished [e.g., the test library "GOSSIP" from 20] or a generic formulation is given while the concrete problem data is not [see, e.g., Sect. 7.1 in 22].

We consider variants of a simple test problem with $N_x = N_y = 1$ and $N_s = 4$, 8, and 16, i.e., with different size based on the number of scenarios. The test problem is a simplified design and operation problem for a combined heat and power (CHP) system, based on stochastic heat and power demands. Scenarios for demand data are generated from a seeded pseudorandom sampling, ensuring identical instances upon repetition for a given N_s value. The problem involves nonlinearities related to economies of scale, thermal and electrical efficiencies, and the implementation of a minimal part-load constraint. A detailed description of the problem is given in "Appendix A".

We focus on the performance difference of the lower bounding routines, hence all experiments are performed with initial points based on dense uniform sampling of 1000 values in each of the x and y_s domains, which always results in ε_f-optimal initial points, that are never improved during the course of the algorithm. We use the default settings of MAiNGO, including a relative optimality tolerance of 1%. All computational experiments are performed on the RWTH Compute Cluster "CLAIX-2018". Each compute node has 2 Intel Xeon Platinum 8160 Processors with 2.1 GHz, 24 cores each, i.e., there is a total of 48 cores per compute node, and 4 GB of main memory per core. In initial tests we observed significant variation of run times, both for MAiNGO and MUSE-BB. We attribute this variation to execution on particular – likely overloaded – compute nodes which consistently require longer solution times compared to other compute nodes. To reduce the effect of this variation, we repeat the solution of each considered instance 20 times and report median values of the resulting solution times and optimality gaps.

6.1 Importance of branching priority

Initially we will focus on the case $k_{\max} = 1$, i.e., we branch only on second-stage variable instances that either produce infeasible subproblems or produce the highest strong-branching score. This means each multisection of second stage variables results in at most 2 child nodes being created, i.e., as in a standard B&B algorithm like MAiNGO.

In problems like DE, exhibiting two-stage structure, the first-stage variables appear in all of the scenario subproblems, while the second-stage variable instances only appear in one, each. This suggests a higher importance of branching on first-stage vs. second-stage variables, especially with increasing N_s. In B&B algorithms, the priority with which variables are branched is typically controlled via branching priorities for individual variables, which are multiplied with the relative interval width before selecting a variable to branch on, also cf. the description of Subroutine 1. As a result, it seems intuitive that B&B algorithms solving DE may generally benefit from relatively high branching priorities for the first-stage variables compared to the second-stage variables, independent of whether decomposition is used or not. For this reason, we compare how MAiNGO and MUSE-BB perform with different branching priority ratios

$$\rho = \frac{\text{first-stage branching priority}}{\text{second-stage branching priority}},$$

which in the present case ($N_x = N_y = 1$) correspond to the branching priority of x (the priority for y_s being 1).

Figure 3 shows the wall times spent in B&B when using MAiNGO and MUSE-BB on problem instances with $N_s \in \{4, 8, 16\}$. Both individual times (colored dots), as well as the median times (horizontal lines) are depicted. Table 1 lists the median wall times and relative gaps for instances which do not terminate within the time-limit of one hour. In general, the ρ

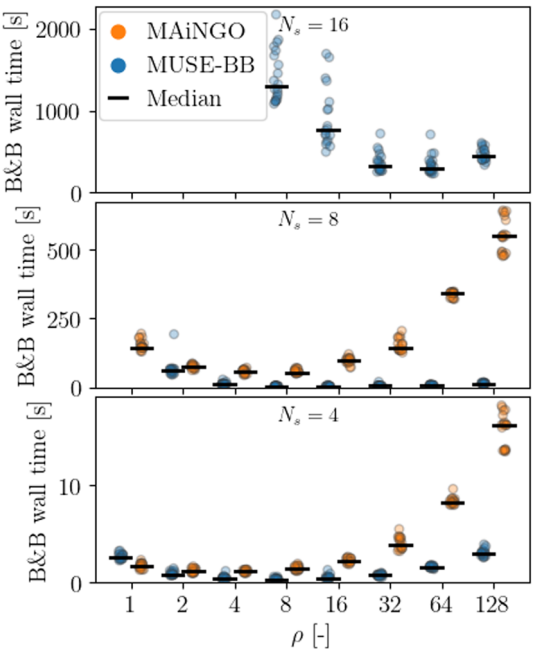


Fig. 3 Variation of solution time for deterministic equivalent (MAiNGO) and parallel decomposition (MUSE-BB) with ρ for $N_s \in \{4, 8, 16\}$ over 20 runs each. Parameter combinations without data points did not terminate within 3600s, also see Table 1

Table 1 Median B&B wall times in seconds over 20 runs, or remaining relative optimality gaps in % (computed as 1 - ratio of lower to upper bound) after 3600 s for solving the CHP sizing model with different number of scenarios and branching priorities, using MAiNGO and MUSE-BB

Algor	MAiNGO			MUSE-BB		
ρN_s	4	8	16	4	8	16
1	1.7	145	17%	2.6	2.4%	20%
2	1.1	77	8.8%	0.82	62	7.6%
4	**1.1**	59	4.6%	0.42	12	2.8%
8	1.4	**55**	**3.7%**	**0.33**	4.3	1292
16	2.2	98	3.8%	0.43	**3.5**	765
32	3.9	142	4.3%	0.75	5.2	329
64	8.2	342	5.0%	1.6	7.8	**295**
128	16	550	5.7%	3.0	14	436

Two out of the 20 runs for the instance $N_s = \rho = 16$, solved with MUSE-BB, timed out. The median is computed with respect to the remaining 18 runs. Minima for each column (highlighted in bold) indicate that the performance of MUSE-BB relatively to MAiNGO improves with an increase of scenarios, and thus problem size

values minimizing average wall time for each scenario are much lower for MAiNGO than for MUSE-BB. However, low ρ values lead to significantly worse performance for MUSE-BB than for MAiNGO, e.g., all runs for $(N_s, \rho) = (8, 1)$ time out after one hour with a median remaining gap of 2.4%. For $N_s = 16$, all instances solved with MAiNGO time out, while for MUSE-BB almost all instances with ρ values above 4 terminate (with the exception of two outliers for $\rho = 16$). This indicates the importance of appropriate branching priorities when solving stochastic problems in general, and when using MUSE-BB in particular. When comparing the best ρ values for each scenario (bold in Table 1), MUSE-BB outperforms

MAiNGO in terms of wall time by a factor of 3.5 and 15 for $N_s = 4$ and $N_s = 8$, respectively. For $N_s = 16$ the value is expected to be significantly larger than $3600/295.3 \approx 12$.

Note that already for these relatively small problem sizes outperformance is close to or even larger than the number of scenarios and thus the number of used threads. This implies that MUSE-BB can be more favorable than more general parallelization approaches such as, e.g., the MPI parallelization of MAiNGO, where open nodes are processed by different CPUs (not used in this work). While such general parallelization approaches are more widely applicable, they do not exploit the special problem structure of DE. Consequently they may be used in conjunction with the parallel processing of individual B&B nodes presented in this work to optimally use computational infrastructure.

The results indicate that optimal branching priority ratios (i.e., ρ values resulting in minimal median wall time) may increase with the number of scenarios considered. Nevertheless, a projection-based approach, where only the first-stage variables are branched (corresponding to $\rho \to \infty$) appears unfavorable, as wall times increase significantly for large ρ-values.

6.2 Effect of multisection

We next consider the effect of the multisection parameters $k_{\max}$, and η. Recall that every time a second-stage variable is selected for branching, we solve the $2N_s$ independent subproblems, resulting from the multisection involving the corresponding N_s variable instances for different scenarios. We then use the results to compute strong-branching scores σ_s for each scenario, and create up to $2^{k_{\max}}$ child nodes, with the actual number being controlled by the value of the strong-branching threshold $\eta \in (0, 1]$, i.e., we reject bisections with a strong-branching score below $\eta\sigma_s$, see Sect. 4.4.

For each N_s value, we take the three ρ values for which MUSE-BB performed best at $k_{\max} = 1$, and perform further experiments for $k_{\max} \in \{2, 4, 8\}$, and $\eta \in \{0.1, 0.2, 0.5, 0.8, 1\}$. Increasing values of $k_{\max}$, and decreasing values of η allow a larger number of child nodes to be created from each multisection, i.e., the maximum is $2^8 = 256$ for $(k_{\max}, \eta) = (8, 1)$. We point out that multiple variables may achieve the maximum strong-branching score. Hence, even for $\eta = 1$, the settings $k_{\max} = 1$, and $k_{\max} > 1$, may produce different B&B trees (and thus wall times) for a given problem instance, as the latter setting allows creating more than 2 child nodes, while the former does not.

As before, we repeat the solution for each parameter combination 20 times. Since combinations with $N_s = 4$, and $N_s = 8$, show no clear trend for the effect of $k_{\max}$, or η, we only focus on combinations with $N_s = 16$, the results of which are depicted in Fig. 4. The B&B wall times of all investigated combinations are visualized in Fig. 6 in "Appendix B".

Only a small set of parameter combinations results in improvements over the best median wall time for $k_{\max} = 1$, (i.e., 295 s for $\rho = 64$). However, these improvements are mostly insignificant, with the best median wall time of 272 s (achieved for $(k_{\max}, \rho, \eta) = (4, 32, 1)$) corresponding to an improvement of less than 8%. For the remaining parameter combinations median wall times remain the same or increase. While combinations with $(N_s, k_{\max}) = (16, 2)$ show no clear trend for the effect of η, for $(N_s, k_{\max}) = (16, 4)$, and $(16, 8)$, an increase of η results in reductions of wall time.

We point out that setting the strong-branching threshold η to a value of 1 produces very similar results as setting $k_{\max}$ to 1, since only bisections that produce the highest strong-branching score may be selected. In fact, for the considered parameter combinations, the total number of iterations for $k_{\max} > 1$ only depends on ρ, with the corresponding values being around 0.06% lower than those for $k_{\max} = 1$.

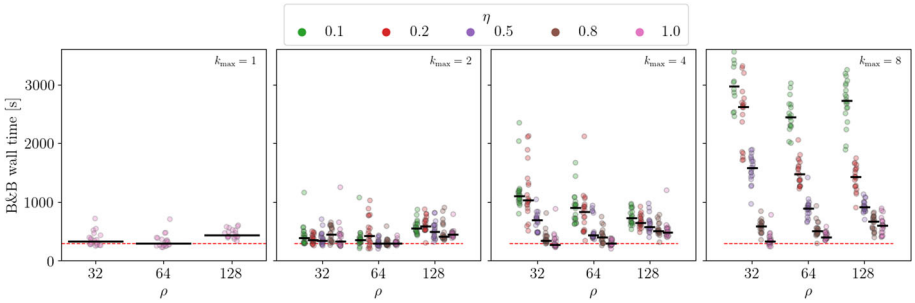


Fig. 4 Variation of solution times for solving the CHP sizing problem using MUSE-BB with $k_{\max} \in \{1, 2, 4, 8\}$, and $\eta \in \{0.1, 0.2, 0.5, 0.8, 1\}$ for the three ρ values resulting in the lowest median wall times for $(k_{\max}, N_s) = (1, 16)$. For each parameter combination, a set of 20 runs is performed. For $(k_{\max}, \rho, \eta) = (8, 32, 0.1)$, $(8, 32, 0.2)$, and $(8, 64, 0.1)$, 8, 2, and 3 runs timed out after 3600 s, respectively. Medians with respect to the remaining runs are depicted as horizontal black lines and the lowest median time for $k_{\max} = 1$ is depicted as a dashed red line for reference. Increasing $k_{\max}$ generally results in larger wall times, and for $k_{\max} = 4$ and 8, increasing η results in smaller wall times. (Color figure online)

For the considered problem, increasing $k_{\max}$ and reducing η tends to result in increases of median wall times, however, the generality of this finding needs to be investigated with a larger group of problems. In particular, it is conceivable that for problems in which multiple variable instances have a comparatively strong effect on the objective or feasible set, values of $k_{\max} > 1$ and $\eta < 1$ may be preferable.

For $k_{\max} = 1$ or $\eta = 1$, the behavior of MUSE-BB is very similar to that of a standard B&B algorithm solving the deterministic equivalent with a strong-branching heuristic. While this is not commonly done, range reduction similar to that of MUSE-BB, i.e., using the intersections of variable domains from rejected bisections for the selected one, may also be done on the basis of full-space bounding problems within classical strong-branching. Whereas MUSE-BB solves smaller, independent subproblems, the bounds obtained from such an adapted strong-branching routine in a standard B&B are naturally stronger. This trade-off appears to be worth further study in future work.

6.3 Scaling with N_s

As we pointed out in the introduction, the fact that MUSE-BB employs a B&B search in the full variable space implies that the number of nodes visited, and thus computational effort, scales exponentially with N_s in the worst-case. This is despite the fact that the proposed multisection branching allows processing an exponential number of nodes with an effort that is linear in N_s, since each of the resulting nodes may need to be further branched and processed.

The computational results from previous sections confirm this expected superlinear scaling with N_s, but also highlight the superiority of MUSE-BB over the solution of the deterministic equivalent via MAiNGO. For MAiNGO the computational time of the best parameter combinations increases by a factor of 50 (55 s / 1.1 s) when going from $N_s = 4$ to $N_s = 8$.(cf. bold times in Table 1). In contrast, for MUSE-BB, the corresponding factor is only 10.6 (3.5 s / 0.33 s).

The optimal value of ρ appears to scale approximately linearly with N_s. Thus it may appear that for large numbers of scenarios, MUSE-BB will behave somewhat like a PBDA,

in the sense that branching is done primarily on $\boldsymbol{x}$. However, recall that each time a particular second-stage variable instance is selected for branching, the current implementation chooses all other instances of that variable for multisection. Since the number of second-stage variable instances increases linearly with N_s, a fixed value of ρ would thus result in more frequent branching on a given second-stage variable instance. Therefore the observed increase of ρ does not necessarily imply a more frequent branching of $\boldsymbol{x}$, but can rather be seen as a compensation for the above-mentioned behavior. Furthermore, unlike PBDAs, MUSE-BB avoids the global solution of subproblems. This difference in computational effort complicates direct comparisons based on individual iterations of MUSE-BB and PBDAs. Again, a more comprehensive comparison with a larger set of test problems will be needed to determine which class of algorithms is best suited to different types of problems.

7 Summary and outlook

We present MUSE-BB, a multisection B&B-based decomposition algorithm for the deterministic global optimization of general nonconvex nonlinear two-stage problems. We prove finite ε_f-convergence, show favorable convergence order of our lower bounding scheme, compared to existing algorithms, and provide initial computational results indicating good scalability of MUSE-BB with the number of scenarios.

Existing decomposition algorithms for two-stage nonconvex MINLP problems [20–22] have been classified as PBDAs [23, 24], since they all employ spatial B&B in the first-stage variables. PBDAs achieve this by solving decomposable subproblems of both first- and second-stage variables in each node. To obtain good lower bounds, these subproblems are solved globally via a nested spatial B&B. Instead, we propose to branch on both first- and second-stage variables within a single B&B tree, and to further relax subproblems, avoiding duplicate branching on first-stage variables, and the nesting of spatial B&B procedures. We either branch normally on a single first-stage variable, or we simultaneously branch on multiple second-stage variables from different scenarios. While such multisection produces an exponential number of child nodes, the total number of distinct subproblems is linear in the number of bisected variables, by virtue of the decomposition. Thus, we only need to process the distinct subproblems and can generate child nodes by appropriately combining the subproblem results. To avoid an excessive number of child nodes with poor lower bounds, we only use a subset of bisections (reverting the remaining ones). We select the bisections based on their associated strong-branching scores, which are readily available after processing. This allows to only generate child nodes corresponding to the most promising bisections with highest strong-branching scores.

Our theoretical results show that by branching on all variables, the lower bounding scheme of MUSE-BB generally has a convergence order of one, if all functions are Lipschitz. This is in contrast to lower bounding schemes of existing decomposition algorithms, which may have convergence orders below one, in general [24]. Whether or not this improved convergence order actually translates into an advantage with respect to the occurrence of clustering is however not clear at this point, and requires further investigation.

We perform initial computational experiments with a small test problem, which despite its size still incorporates relevant nonlinearities found in applications. Our results highlight the importance of choosing appropriate branching priorities for both general B&B and decomposition algorithms. Moreover, the results show that even for this small problem and small numbers of scenarios, MUSE-BB can significantly outperform the standard version of our

open-source deterministic global solver MAiNGO, applied to the deterministic equivalent formulation. For the considered problem instances, the best wall times of MUSE-BB are achieved when essentially limiting the number of child nodes resulting from multisection to two. In this case, MUSE-BB behaves very similar to a B&B algorithm solving the deterministic equivalent using a strong-branching heuristic that employs certain range reduction with the rejected bisections.

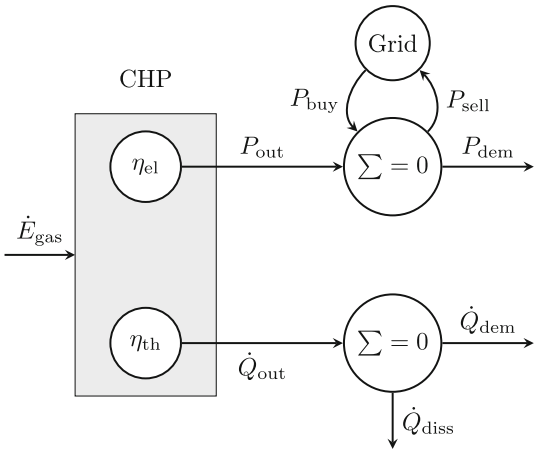
Further theoretical and computational comparison of MUSE-BB with other decomposition algorithms, and with specialized strong-branching in full-space B&B is necessary, in particular, determining conditions in which each method is preferable, e.g., depending on the numbers of first- and second-stage variables, and scenarios. However, as meaningful computational studies require adequate implementations of the alternative algorithms and a sufficiently representative test set, we leave this as future work.

As with existing decomposition algorithms, the lower bounding scheme of MUSE-BB may be improved by dualizing the coupling (nonanticipativity) constraints instead of dropping them. The resulting lower bounding scheme can be shown to have second-order convergence at minimizers satisfying certain regularity conditions. However, this extension requires optimal dual multipliers, which are expensive to compute in general. The details of how such an extension can be implemented need to be clarified, in particular, when using further relaxations.

We aim to investigate the effect of combining the parallel bounding routines of MUSE-BB with more general B&B parallelization as implemented in MAiNGO, as this is expected to make MUSE-BB applicable to much larger, more realistic case studies. Furthermore, it may be interesting to generalize the presented implementation to problems with different numbers of second-stage variables and constraints. A related extension would allow branching on arbitrary combinations of second-stage variables from different scenarios, instead of limiting multisection to scenario instances of a particular second-stage variable. Finally, the decomposable bounding routines of MUSE-BB may enable efficient strong-branching in problems that do not fall into the category of two-stage programming problems, but still exhibit block structures, coupled by complicating constraints.

Since MUSE-BB operates a B&B search in the full variable space, its worst-case computational effort remains exponential in N_s. While our computational results indicate that it can perform significantly better than a B&B algorithm applied to the deterministic equivalent, an open question is whether the exponential scaling can be avoided without resorting to existing projection-based algorithms. To this end, we also outlined how an alternative handling of multisection may allow nesting B&B trees which exclusively address second-stage variables from individual scenarios within a tree that addresses first-stage variables. This approach may be seen as a hybrid between existing PBDAs (which duplicate the first stage variables in the nested B&B trees) and MUSE-BB.

Fig. 5 Conceptual CHP operation



A Test Problem

We consider the design of a combined heat and power (CHP) unit, i.e., an equipment sizing problem whose aim is to satisfy given heat and power demands at minimum cost, see Fig. 5.

The size of the CHP is expressed as a nominal heat output $\dot{Q}_{\text{nom}}$, which corresponds to the maximum thermal output $\dot{Q}_{\text{out}}$. The actual output at any given point is determined by a relative heat output $\dot{Q}_{\text{rel}}$:

$$\dot{Q}_{\text{out}} := \dot{Q}_{\text{nom}}\, \dot{Q}_{\text{rel}} \tag{3}$$

The energy input to the CHP in terms of lower heating value of natural gas, $\dot{E}_{\text{gas}}$, can be calculated via the thermal efficiency η_{th}, which is a function of $\dot{Q}_{\text{nom}}$ and $\dot{Q}_{\text{rel}}$:

$$\dot{E}_{\text{gas}} := \frac{\dot{Q}_{\text{out}}}{\eta_{\text{th}}(\dot{Q}_{\text{nom}}, \dot{Q}_{\text{rel}})} \tag{4}$$

Following this the power output P_{out} can be computed via the electrical efficiency η_{th}, which is also a function of $\dot{Q}_{\text{nom}}$ and $\dot{Q}_{\text{rel}}$:

$$P_{\text{out}} := \dot{E}_{\text{gas}}\, \eta_{\text{el}}(\dot{Q}_{\text{nom}}, \dot{Q}_{\text{rel}}) \tag{5}$$

The functional form of the efficiencies η_{th} and η_{el} is given by:

$$\eta_{\text{th}}(\dot{Q}_{\text{nom}}, \dot{Q}_{\text{rel}}) := \eta_{\text{th,nom}}(\dot{Q}_{\text{nom}})\, \eta_{\text{th,rel}}(\dot{Q}_{\text{rel}}) \tag{6}$$

$$\eta_{\text{th,nom}}(\dot{Q}_{\text{nom}}) := 0.498 - \frac{\dot{Q}_{\text{nom}}}{21.17\text{MW}} \tag{7}$$

$$\eta_{\text{th,rel}}(\dot{Q}_{\text{rel}}) := 1.10 - 0.0768\,(\dot{Q}_{\text{rel}} + 0.130)^2 \tag{8}$$

$$\eta_{\text{el}}(\dot{Q}_{\text{nom}}\, \dot{Q}_{\text{rel}}) := \eta_{\text{el,nom}}(\dot{Q}_{\text{nom}})\, \eta_{\text{el,rel}}(\dot{Q}_{\text{rel}}) \tag{9}$$

$$\eta_{\text{el,nom}}(\dot{Q}_{\text{nom}}) := 0.372 + \frac{\dot{Q}_{\text{nom}}}{21.17\text{MW}} \tag{10}$$

$$\eta_{\text{el,rel}}(\dot{Q}_{\text{rel}}) := 1.02 - 0.435\,(0.774\,\dot{Q}_{\text{rel}} - 1)^2 \tag{11}$$

A heat shortage, defined as

$$\dot{Q}_{\text{short}} := \dot{Q}_{\text{dem}} - \dot{Q}_{\text{out}} \tag{12}$$

must be avoided (i.e., $\dot{Q}_{\text{short}}$ must be negative). Correspondingly, the power shortage can be defined as:

$$P_{\text{short}} := P_{\text{dem}} - P_{\text{out}} \tag{13}$$

A power shortage can be adressed by purchasing power from the grid, i.e.:

$$P_{\text{buy}} := \max(0, P_{\text{short}}) \tag{14}$$

If P_{short} or $\dot{Q}_{\text{short}}$ are negative, the excess power can be sold to the grid at a reduced price, while the excess heat can be dissipated into the environment:

$$\dot{P}_{\text{sell}} := \max(0, -P_{\text{short}}) \tag{15}$$

$$\dot{Q}_{\text{diss}} := \max(0, -\dot{Q}_{\text{short}}). \tag{16}$$

With these definitions we can formulate a reduced-space problem that contains the nominal heat output as the only first-stage variable, i.e: $\boldsymbol{x} = (\dot{Q}_{\text{nom}}) \in [1.4\,\text{MW}, 2.3\,\text{MW}]$, and the part-load in each scenario as the only second-stage variable, i.e: $\boldsymbol{y}_s = (\dot{Q}_{\text{rel},s}) \in [0, 1]$.

We choose total annualized costs (TAC) in million € as the objective function. The first-stage objective function describes the annualized investment costs (according to an economy of scales approach) and the second-stage objectives correspond to the annual operating costs in each scenario:

$$f_{\text{I}}(\boldsymbol{x}) = 149567\text{€/a} \left(\frac{\dot{Q}_{\text{nom}}}{1\,\text{MW}} \right)^{0.9} \times 10^{-6} \tag{17}$$

$$\begin{aligned} f_{\text{II},s}(\boldsymbol{x}, \boldsymbol{y}_s) = T_{\text{op}} (& p_{\text{gas}} \dot{E}_{\text{gas},s} \\ & + p_{\text{el,buy}} P_{\text{buy},s} \\ & - p_{\text{el,sell}} P_{\text{sell},s}) \times 10^{-6} \end{aligned} \tag{18}$$

Where $T_{\text{op}} = 6000\,\text{h/a}$, $p_{\text{gas}} = 80\text{€/(MW h)}$, $p_{\text{el,buy}} = 250\text{€/(MW h)}$, $p_{\text{el,sell}} = 100\text{€/(MW h)}$

We approximate the requirement that the CHP unit must either be inactive or operate above a minimal part-load threshold of 50% with quadratic second-stage constraints of the form

$$0.0619263 - (\dot{Q}_{\text{rel},s} - 0.25115)^2 \leq 0 \tag{19}$$

which restrict the relative outputs $\dot{Q}_{\text{rel},s}$ to less than 0.1%, or more than 50% part-load. An additional constraint is that

$$\dot{Q}_{\text{short},s} \leq 0, \tag{20}$$

also see Eq. (12). Note that Eq. (19) implies that heat demands corresponding to part-loads between 0.1% and 50% cannot be satisfied. To ensure the considered instances have a feasible solution, the randomly generated heat demands are set to 0 if they fall into this range. Similarly, the generated power and heat demands are capped to the highest possible production.

B Parameter Study for $k_{\max}$ and η

See Fig. 6.

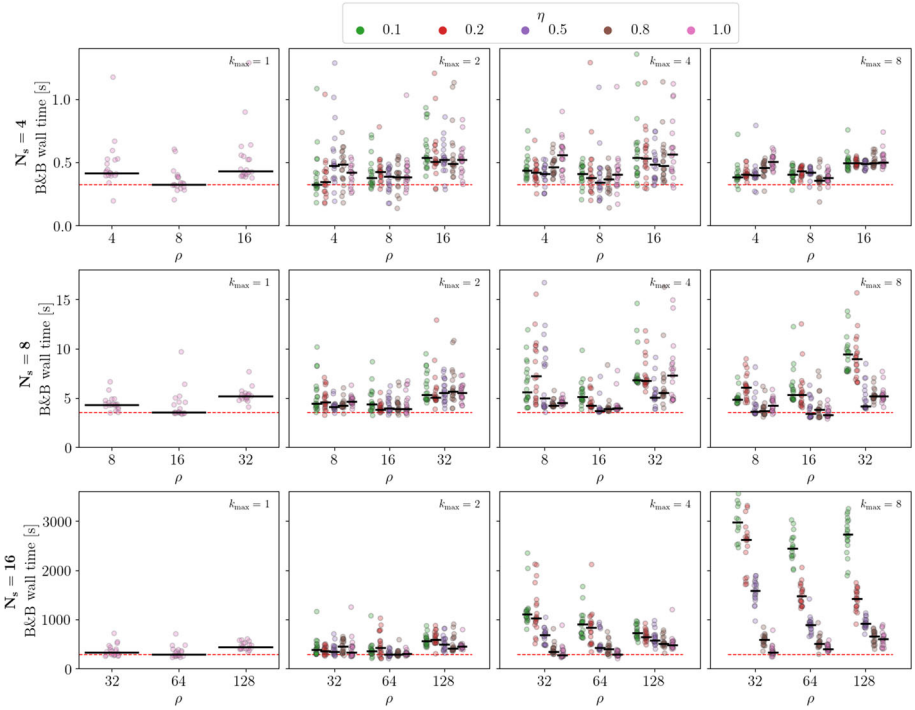


Fig. 6 Variation of solution times for solving the CHP sizing problem using MUSE-BB with $k_{\max} \in \{1, 2, 4, 8\}$, and $\eta \in \{0.1, 0.2, 0.5, 0.8, 1\}$ for the three ρ values resulting in the lowest median wall times for $N_s = 4$, 8, and 16, with $k_{\max} = 1$. For each parameter combination, a set of 20 runs is performed. For $(k_{\max}, \rho, \eta) = (8, 32, 0.1)$, $(8, 32, 0.2)$, and $(8, 64, 0.1)$, 8, 2, and 3 runs timed out after 3600 s, respectively. Medians with respect to the remaining runs are depicted as horizontal black lines and the lowest median time for $k_{\max} = 1$ is depicted as a dashed red line for reference. Whereas for $N_s = 4$, and 8, no clear trend is discernible, for $N_s = 16$, increasing $k_{\max}$ generally results in larger wall times, and for $k_{\max} = 4$, and 8, increasing η results in smaller wall times. (Color figure online)

C Overview of Functions and Optimization Problems

Function	explanation
f	overall objective function
f_{I}	first-stage objective function
$f_{\mathrm{II},s}$	second-stage objective function
$f_{\mathrm{II},s}^{\mathcal{Y}_s}$	second-stage optimal value function
f_s	scenario objective function
$f_s^{\mathcal{X},\mathcal{Y}_s}$	scenario optimal value function

Problem	explanation
TSP	two-stage (stochastic programming) problem
DE (DE^n)	deterministic equivalent form of TSP (restricted to the domain of node n)
$\mathrm{DE}_{\mathrm{NAC}}$	NAC formulation, equivalent to DE and TSP
RP_s^n	recourse problems for a given value of $\boldsymbol{x}$ for node n (providing upper bounds)
LSP_s^n	relaxations of $\mathrm{DE}_{\mathrm{NAC}}$ for node n by dualizing NACs with multipliers $\boldsymbol{\lambda}_s$
SP_s^n	relaxations of $\mathrm{DE}_{\mathrm{NAC}}$ for node n by dropping NACs
MC_s^n	McCormick relaxation of SP_s^n for node n
LP_s^n	linear relaxation of MC_s^n for node n (providing lower bounds)
$\mathrm{OBBT}_{s,v}^n$	OBBT problems for variable v and node n
R_s^n	generic scenario relaxation for node n

Acknowledgements Funding by the Werner Siemens Foundation within the WSS project of the century "catalaix" is acknowledged. Simulations were performed with computing resources granted by RWTH Aachen University under project rwth1468. M. L., M. D., and A. M. acknowledge funding from the Helmholtz Association of German Research Centers. The authors are grateful to J. K. Scott for his instructive talk [23], providing a recording of this talk, and fruitful discussions. Furthermore, they sincerely thank the reviewers for their insightful feedback and constructive suggestions, which have significantly improved the quality and clarity of the manuscript.

Author Contributions A. M., D. B., M. L. contributed to the conceptual development of the algorithm. A. M., and M. L. developed the theoretical results (proofs and examples in 5). M. L. carried out the software implementation with guidance from D. B. M. L. designed and conducted the computational experiments and analyzed the results. Conceptualization: A. M., D. B., M. L. Formal Analysis: A. M., M. L. Funding Acquisition: A. M., M. D. Methodology: A. M., M. L. Software: M. L. Supervision: A. M., M. D. Validation: M. L. Visualization: M. L. Writing—original draft: A. M., M. L. Writing—review and editing: All authors.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

1. Birge, J.R., Louveaux, F.V.: Introduction to Stochastic Programming. Springer, New York (2011). https://doi.org/10.1007/978-1-4614-0237-4
2. Yunt, M., Chachuat, B., Mitsos, A., Barton, P.I.: Designing man-portable power generation systems for varying power demand. AIChE J. **54**(5), 1254–1269 (2008). https://doi.org/10.1002/aic.11442
3. Langiu, M., Dahmen, M., Mitsos, A.: Simultaneous optimization of design and operation of an air-cooled geothermal ORC under consideration of multiple operating points. Comput. Chem. Eng. **161**, 107745 (2022). https://doi.org/10.1016/j.compchemeng.2022.107745
4. Androulakis, I.P., Maranas, C.D., Floudas, C.A.: alphaBB: a global optimization method for general constrained nonconvex problems. J. Glob. Optim. **7**(4), 337–363 (1995). https://doi.org/10.1007/bf01099647
5. Vigerske, S., Gleixner, A.M.: SCIP: global optimization of mixed-integer nonlinear programs in a branch-and-cut framework. Optim. Methods Softw. **33**(3), 1–31 (2017). https://doi.org/10.1080/10556788.2017.1335312
6. Bongartz, D., Najman, J., Sass, S., Mitsos, A.: MAiNGO: McCormick based Algorithm for mixed integer Nonlinear Global Optimization. Technical report, Process Systems Engineering (AVT.SVT), RWTH Aachen University (2018). http://permalink.avt.rwth-aachen.de/?id=729717. The open-source version is available at https://git.rwth-aachen.de/avt-svt/public/maingo. The corresponding Python package `maingopy` is available at https://pypi.org/project/maingopy
7. Belotti, P.: Couenne: a User's Manual (2019). https://www.coin-or.org/Couenne/couenne-user-manual.pdf. Accessed 13 May 2024
8. Sahinidis, N.V.: BARON user manual: v. 24.05.08 (2024). http://www.minlp.com/downloads/docs/baron%20manual.pdf. Accessed 08 May 2024
9. Dantzig, G.B., Wolfe, P.: Decomposition principle for linear programs. Oper. Res. **8**(1), 101–111 (1960). https://doi.org/10.1287/opre.8.1.101
10. Benders, J.F.: Partitioning procedures for solving mixed-variables programming problems. Numer. Math. **4**(1), 238–252 (1962). https://doi.org/10.1007/BF01386316
11. Laporte, G., Louveaux, F.V.: The integer L-shaped method for stochastic integer programs with complete recourse. Oper. Res. Lett. **13**(3), 133–142 (1993). https://doi.org/10.1016/0167-6377(93)90002-X
12. Carøe, C.C., Schultz, R.: Dual decomposition in stochastic integer programming. Oper. Res. Lett. **24**(1–2), 37–45 (1999). https://doi.org/10.1016/S0167-6377(98)00050-9
13. Geoffrion, A.M.: Generalized benders decomposition. J. Optim. Theory Appl. **10**(4), 237–260 (1972). https://doi.org/10.1007/BF00934810
14. Li, X., Tomasgard, A., Barton, P.I.: Nonconvex generalized benders decomposition for stochastic separable mixed-integer nonlinear programs. J. Optim. Theory Appl. **151**(3), 425–454 (2011). https://doi.org/10.1007/s10957-011-9888-1
15. Li, X., Sundaramoorthy, A., Barton, P.I.: Nonconvex generalized benders decomposition. In: Optimization in Science and Engineering, pp. 307–331. Springer, New York (2014). https://doi.org/10.1007/978-1-4939-0808-0_16
16. Karuppiah, R., Grossmann, I.E.: A Lagrangean based branch-and-cut algorithm for global optimization of nonconvex mixed-integer nonlinear programs with decomposable structures. J. Glob. Optim. **41**(2), 163–186 (2007). https://doi.org/10.1007/s10898-007-9203-8
17. Khajavirad, A., Michalek, J.J.: A deterministic Lagrangian-based global optimization approach for quasiseparable nonconvex mixed-integer nonlinear programs. J. Mech. Design (2009). https://doi.org/10.1115/1.3087559
18. Li, H., Cui, Y.: A decomposition algorithm for two-stage stochastic programs with nonconvex recourse functions. SIAM J. Optim. **34**(1), 306–335 (2024). https://doi.org/10.1137/22M1488533
19. Ogbe, E., Li, X.: A joint decomposition method for global optimization of multiscenario nonconvex mixed-integer nonlinear programs. J. Glob. Optim. **75**(3), 595–629 (2019). https://doi.org/10.1007/s10898-019-00786-x
20. Kannan, R.: Algorithms, analysis and software for the global optimization of two-stage stochastic programs. Ph.D. Thesis, Massachusetts Institute of Technology (2018). https://dspace.mit.edu/handle/1721.1/117326. Accessed 13 May 2024
21. Cao, Y., Zavala, V.M.: A scalable global optimization algorithm for stochastic nonlinear programs. J. Glob. Optim. **75**(2), 393–416 (2019). https://doi.org/10.1007/s10898-019-00769-y
22. Li, C., Grossmann, I.E.: A generalized Benders decomposition-based branch and cut algorithm for two-stage stochastic programs with nonconvex constraints and mixed-binary first and second stage variables. J. Glob. Optim. **75**(2), 247–272 (2019). https://doi.org/10.1007/s10898-019-00816-8

23. Robertson, D.L., Cheng, P., Scott, J.K.: Convergence rate analysis for schemes of relaxations in decomposition methods for global nonconvex stochastic optimization. In: CAST Plenary Session of AIChE Annual Meeting 2020 (2020)
24. Robertson, D., Cheng, P., Scott, J.K.: On the convergence order of value function relaxations used in decomposition-based global optimization of nonconvex stochastic programs. J. Glob. Optim. (2025). https://doi.org/10.1007/s10898-024-01458-1
25. Kearfott, B., Du, K.: The cluster problem in global optimization: the univariate case. In: Computing Supplementum, pp. 117–127. Springer, Vienna (1993). https://doi.org/10.1007/978-3-7091-6918-6_10
26. Du, K., Kearfott, R.B.: The cluster problem in multivariate global optimization. J. Glob. Optim. **5**(3), 253–265 (1994). https://doi.org/10.1007/bf01096455
27. Wechsung, A., Schaber, S.D., Barton, P.I.: The cluster problem revisited. J. Glob. Optim. **58**(3), 429–438 (2014). https://doi.org/10.1007/s10898-013-0059-9
28. Kannan, R., Barton, P.I.: The cluster problem in constrained global optimization. J. Glob. Optim. **69**(3), 629–676 (2017). https://doi.org/10.1007/s10898-017-0531-z
29. Bompadre, A., Mitsos, A.: Convergence rate of McCormick relaxations. J. Glob. Optim. **52**(1), 1–28 (2011). https://doi.org/10.1007/s10898-011-9685-2
30. Najman, J., Mitsos, A.: Convergence analysis of multivariate McCormick relaxations. J. Glob. Optim. **66**(4), 597–628 (2016). https://doi.org/10.1007/s10898-016-0408-6
31. 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
32. Smith, E.M.B., Pantelides, C.C.: Global optimisation of nonconvex MINLPs. Comput. Chem. Eng. **21**, 791–796 (1997). https://doi.org/10.1016/s0098-1354(97)87599-0
33. Karmakar, S., Mahato, S.K., Bhunia, A.K.: Interval oriented multi-section techniques for global optimization. J. Comput. Appl. Math. **224**(2), 476–491 (2009). https://doi.org/10.1016/j.cam.2008.05.025
34. Dür, M., Horst, R.: Lagrange duality and partitioning techniques in nonconvex global optimization. J. Optim. Theory Appl. **95**(2), 347–369 (1997). https://doi.org/10.1023/a:1022687222060
35. Oliveira, F., Gupta, V., Hamacher, S., Grossmann, I.E.: A Lagrangean decomposition approach for oil supply chain investment planning under uncertainty with risk considerations. Comput. Chem. Eng. **50**, 184–195 (2013). https://doi.org/10.1016/j.compchemeng.2012.10.012
36. Kirst, P., Stein, O., Steuermann, P.: Deterministic upper bounds for spatial branch-and-bound methods in global minimization with nonconvex constraints. TOP **23**(2), 591–616 (2015). https://doi.org/10.1007/s11750-015-0387-7
37. Füllner, C., Kirst, P., Stein, O.: Convergent upper bounds in global minimization with nonlinear equality constraints. Math. Programm. **187**(1–2), 617–651 (2020). https://doi.org/10.1007/s10107-020-01493-2
38. Csallner, A.E., Csendes, T., Markót, M.C.: Multisection in interval branch and bound methods for global optimization—I. Theor. Results. J. Glob. Optim. **16**(4), 371–392 (2000). https://doi.org/10.1023/a:1008354711345
39. Markót, M.C., Csendes, T., Csallner, A.E.: Multisection in interval branch and bound methods for global optimization—II. Numer. Tests. J. Glob. Optim. **16**(3), 219–228 (2000). https://doi.org/10.1023/a:1008359223042
40. Kazazakis, N.: Parallel Computing, Interval Derivative Methods, Heuristic Algorithms, and Their Implementation in a Numerical Solver, for Deterministic Global Optimization. Ph.D. Thesis, Imperial College London (2017)
41. McCormick, G.P.: Computability of global solutions to factorable nonconvex programs: part I—convex underestimating problems. Math. Programm. **10**(1), 147–175 (1976). https://doi.org/10.1007/bf01580665
42. Tsoukalas, A., Mitsos, A.: Multivariate McCormick relaxations. J. Glob. Optim. **59**(2–3), 633–662 (2014). https://doi.org/10.1007/s10898-014-0176-0
43. Villanueva, M.E.: Set-Theoretic Methods for Analysis Estimation and Control of Nonlinear Systems. Ph.D. Thesis, Imperial College London (2015)
44. Chachuat, B., Houska, B., Paulen, R., Peri'c, N., Rajyaguru, J., Villanueva, M.E.: Set-theoretic approaches in analysis, estimation and control of nonlinear systems. IFAC-PapersOnLine **48**(8), 981–995 (2015). https://doi.org/10.1016/j.ifacol.2015.09.097
45. Najman, J., Bongartz, D., Mitsos, A.: Linearization of McCormick relaxations and hybridization with the auxiliary variable method. J. Glob. Optim. (2021). https://doi.org/10.1007/s10898-020-00977-x
46. Najman, J., Mitsos, A.: Tighter McCormick relaxations through subgradient propagation. J. Glob. Optim. **75**(3), 565–593 (2019). https://doi.org/10.1007/s10898-019-00791-0
47. Li, D., Li, X.: Global optimization of an industrial natural gas production network. IFAC-PapersOnLine **48**(8), 337–342 (2015). https://doi.org/10.1016/j.ifacol.2015.08.204
48. Schichl, H., Neumaier, A.: Interval analysis on directed acyclic graphs for global optimization. J. Glob. Optim. **33**(4), 541–562 (2005). https://doi.org/10.1007/s10898-005-0937-x

49. Gleixner, A.M., Berthold, T., Müller, B., Weltge, S.: Three enhancements for optimization-based bound tightening. J. Glob. Optim. **67**(4), 731–757 (2016). https://doi.org/10.1007/s10898-016-0450-4
50. Ryoo, H.S., Sahinidis, N.V.: Global optimization of nonconvex NLPs and MINLPs with applications in process design. Comput. Chem. Eng. **19**(5), 551–566 (1995). https://doi.org/10.1016/0098-1354(94)00097-2
51. Li, D., Li, X.: Domain reduction for Benders decomposition based global optimization. Comput. Chem. Eng. **93**, 248–265 (2016). https://doi.org/10.1016/j.compchemeng.2016.06.009
52. Applegate, D., Bixby, R., Chvatal, V., Cook, B.: Finding Cuts in the TSP (A Preliminary Report). Technical report, AT&T Bell Labs (1995)
53. Achterberg, T., Koch, T., Martin, A.: Branching rules revisited. Oper. Res. Lett. **33**(1), 42–54 (2005). https://doi.org/10.1016/j.orl.2004.04.002
54. Achterberg, T.: Constraint Integer Programming. Ph.D. Thesis, TU Berlin (2007)
55. Kannan, R., Barton, P.I.: Convergence-order analysis of branch-and-bound algorithms for constrained problems. J. Glob. Optim. **71**(4), 753–813 (2017). https://doi.org/10.1007/s10898-017-0532-y
56. Adjiman, C.S., Floudas, C.A.: alphaBB algorithm. In: Encyclopedia of Optimization, pp. 61–73. Springer, New York (2008). https://doi.org/10.1007/978-0-387-74759-0_11
57. Rote, G.: The convergence rate of the sandwich algorithm for approximating convex functions. Computing **48**(3–4), 337–361 (1992). https://doi.org/10.1007/bf02238642
58. Tawarmalani, M., Sahinidis, N.V.: Global optimization of mixed-integer nonlinear programs: a theoretical and computational study. Math Program **99**(3), 563–591 (2004). https://doi.org/10.1007/s10107-003-0467-6
59. Khan, K.A.: Subtangent-based approaches for dynamic set propagation. In: 2018 IEEE Conference on Decision and Control (CDC), pp. 3050–3055 (2018). https://doi.org/10.1109/cdc.2018.8618872
60. Wechsung, A.: Global Optimization in Reduced Space. Ph.D. Thesis, Massachusetts Institute of Technology
61. Scott, J.K., Stuber, M.D., Barton, P.I.: Generalized McCormick relaxations. J. Glob. Optim. **51**(4), 569–606 (2011). https://doi.org/10.1007/s10898-011-9664-7
62. Zlobec, S.: On the Liu–Floudas convexification of smooth programs. J. Glob. Optim. **32**(3), 401–407 (2005). https://doi.org/10.1007/s10898-004-3134-4
63. Fiacco, A.V. (ed.): Introduction to Sensitivity and Stability Analysis in Nonlinear Programming. Elsevier, Amsterdam (1983). https://doi.org/10.1016/s0076-5392(08)x6041-2
64. Ginchev, I., Torre, D.L., Rocca, M.: $\mathcal{C}^{k,1}$ functions, characterization, Taylor's formula and optimization: a survey. Real Anal. Exch. **35**(2), 311–342 (2009)
65. Stechlinski, P., Khan, K.A., Barton, P.I.: Generalized sensitivity analysis of nonlinear programs. SIAM J. Optim. **28**(1), 272–301 (2018). https://doi.org/10.1137/17m1120385
66. Clarke, F.H.: Optimization and Nonsmooth Analysis. SIAM, Philadelphia (1990). https://doi.org/10.1137/1.9781611971309