European Journal of Operational Research 316 (2024) 36–45

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/eor

Continuous Optimization

A branch-and-bound algorithm with growing datasets for large-scale parameter estimation

Susanne Sass [a], Alexander Mitsos [b,a,c], Dominik Bongartz [d], Ian H. Bell [e], Nikolay I. Nikolov [f], Angelos Tsoukalas [g,*]

[a] Process Systems Engineering (AVT.SVT), RWTH Aachen University, Aachen, 52074, Germany
[b] JARA-CSD, Aachen, 52056, Germany
[c] Institute of Energy and Climate Research: Energy Systems Engineering (IEK-10), Forschungszentrum Jülich GmbH, Jülich, 52425, Germany
[d] Department of Chemical Engineering, KU Leuven, Leuven, 3001, Belgium
[e] Applied Chemicals and Materials Division, National Institute of Standards and Technology, Boulder, 80305, CO, United States
[f] Institute of Statistics, RWTH Aachen University, Aachen, 52056, Germany
[g] Department of Technology and Operations Management, Rotterdam School of Management (RSM), Erasmus University Rotterdam, Rotterdam, 3062 PA, Netherlands

ARTICLE INFO

Keywords:
Global optimization
Nonlinear programming
Large scale optimization
Regression
Spatial branch and bound algorithm

ABSTRACT

The solution of nonconvex parameter estimation problems with deterministic global optimization methods is desirable but challenging, especially if large measurement datasets are considered. We propose to exploit the structure of this class of optimization problems to enable their solution with the spatial branch-and-bound algorithm. In detail, we start with a reduced dataset in the root node and progressively augment it, converging to the full dataset. We show for nonlinear programs (NLPs) that our algorithm converges to the global solution of the original problem considering the full dataset. The implementation of the algorithm extends our open-source solver MAiNGO. A numerical case study with a mixed-integer nonlinear program (MINLP) from chemical engineering and a dynamic optimization problem from biochemistry both using noise-free measurement data emphasizes the potential for savings of computational effort with our proposed approach.

1. Introduction

Parameter estimation is typically required to obtain accurate models for the simulation and optimization of real-world processes from industry, the prediction of the performance of drugs and therapies in medicine, or various other applications (e.g., Almquist et al., 2014; Fischler & Bolles, 1981; Gau & Stadtherr, 2002). Estimating the parameters of nonlinear models often leads to nonconvex mixed-integer nonlinear programs (MINLPs) which are typically solved with local and heuristic global optimization methods (see, e.g., Abbiw-Jackson et al., 2006; Egea et al., 2010; Liu & Sahinidis, 1997). But only deterministic global optimization (DGO) methods can prove that a model candidate is not suitable for predicting the given measurement data (Mitsos et al., 2009; Singer & Barton, 2006). However, large-scale MINLPs are challenging for existing DGO methods (Boukouvala et al., 2016; Floudas, 2000; Horst & Pardalos, 1995). Thus, we propose to exploit the structure of parameter estimation problems to enable the solution in presence of large datasets. In particular, we focus on the spatial branch-and-bound (B&B) algorithm (Horst & Tuy, 1996; Tawarmalani & Sahinidis, 2002).

In previous case studies (Sass et al., 2023), we observed that reducing the size of the dataset has the potential to save computational effort while retaining the order of magnitude of lower and upper bounds on the optimal solution of the base problem. Based on these results, we propose to start with a reduced dataset in the root node of the B&B tree and extend it subsequently within the B&B procedure converging to the full dataset, i.e., the original problem formulation, after a finite number of iterations. For this, we introduce *augmentation rules* which decide in each node whether to augment the dataset or to branch. With such *growing datasets*, we aim at pruning obviously suboptimal regions based on a reduced and, thus, computationally less costly model, while eventually we still determine a global solution of the full problem.

Our approach has similarities to the widely-known usage of stochastic gradient descent algorithms or batch gradient descent algorithms

* Corresponding author.
E-mail address: tsoukalas@rsm.nl (A. Tsoukalas).

https://doi.org/10.1016/j.ejor.2024.02.020
Received 31 March 2023; Accepted 16 February 2024
Available online 23 February 2024

to make the training of large-scale models in machine learning (e.g., Byrd et al., 2016; Murphy, 2022) or appointment scheduling (Pan et al., 2021) tractable. However, while these approaches are used to speed up local optimization procedures, we obtain a DGO method. In fact, we extend the proof of convergence to the global optimum for a B&B algorithm applied to nonlinear programs (NLPs) given in Theorem 5.26 of Locatelli and Schoen (2013). As another difference, the (mini)batches or single data points in the aforementioned local optimization procedures are regularly resampled, typically without changing the number of data points, in the respective machine learning approaches. Similarly, Agosti et al. (2020) iteratively update their reduced order model without changing its size based on information from the full order model to estimate patient-specific parameters of a tumor growth model. In contrast to that, we increase the size of the reduced dataset gradually as done for infinite sum problems (see, e.g., Beyhaghi et al., 2020) or in stochastic optimization (Rulliere et al., 2013). While the latter two approaches cannot know the true final dataset, our final dataset is given by the fixed full dataset provided with the model.

We investigate the numerical performance of the proposed B&B algorithm with growing datasets in a case study optimizing the equation of state of propane (Lemmon et al., 2009; Sass et al., 2023) as well as a dynamic model coding metabolic pathways (Moles et al., 2003; Villaverde et al., 2019) based on noise-free synthetic measurement data. For this, we implement the extension for handling growing datasets in our open-source solver MAiNGO[1] for the DGO of factorable MINLPs (Bongartz et al., 2018). MAiNGO uses McCormick relaxations (McCormick, 1976; Tsoukalas & Mitsos, 2014) within the lower bounding procedure, whereas widely used DGO solvers like BARON (Tawarmalani & Sahinidis, 2005) apply the auxiliary variables method (AVM) (Smith & Pantelides, 1997; Tawarmalani & Sahinidis, 2002, 2004, 2005) to obtain convex relaxations. While the number of optimization variables treated within the B&B algorithm remains constant by McCormick relaxations, AVM adds more auxiliary optimization variables as more (nonlinear) terms are included in the model functions and, thus, as a greater number of data points are considered. Aside from that, the use of McCormick relaxations makes MAiNGO particularly suitable for reduced-space formulations (Bongartz & Mitsos, 2017; Mitsos et al., 2009), meaning again that the number of constraints and optimization variables does not necessarily grow with the size of the dataset. In other words, the size of the dataset mainly affects the computational costs for function evaluations rather than the size of the optimization problem solved. Consequently, we expect that the extension to growing datasets is even more advantageous for AVM-based solvers like BARON, especially when working with full-space formulations within these. However, a general study on the performance of different DGO solvers as well as implementing the proposed extension in third-party DGO solvers is not in scope of this study since these solvers are either not open-source or have a too complex code structure.

The remainder of this article is structured as follows. In Section 2, we introduce our key ideas on how to incorporate and exploit growing datasets including our so-called augmentation rules in Sections 2.1 and 2.2. An overview of the B&B algorithm with growing datasets as well as its proof of convergence follows in Section 3. In Section 4, we present numerical results for our case study, before we conclude in Section 5. Basic definitions used in the proof of convergence are given in Appendix A.

[1] Available at https://git.rwth-aachen.de/avt-svt/public/maingo.

2. Parameter estimation exploiting growing datasets

We focus on solving the general parameter estimation problem

$$\begin{aligned} \min_{\boldsymbol{p}\in\mathcal{P}} \; & \overbrace{\sum_{(\boldsymbol{x}_d,y_d)\in\mathcal{D}} \left(f(\boldsymbol{x}_d;\boldsymbol{p})-y_d\right)^2}^{=:\, g_{\mathcal{D}}} \\ \text{s.t. } & h(\boldsymbol{x}_d,y_d;\boldsymbol{p})\le 0 \qquad \forall d=1,\dots,|\mathcal{D}| \\ & \widetilde{h}(\boldsymbol{p})\le 0\,, \end{aligned} \tag{PE}$$

where $\boldsymbol{p}\in\mathcal{P}$ with a closed, bounded box $\mathcal{P}\subsetneq\mathbb{R}^n$ are the unknown parameters, $\mathcal{D}=\{(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\dots\}\subsetneq\mathbb{R}^m\times\mathbb{R}$ denotes the full set of measurement data, $f(\cdot;\boldsymbol{p}):\mathbb{R}^m\to\mathbb{R}$ is a general parameterized model function, $h(\boldsymbol{x}_d,y_d;\cdot):\mathcal{P}\to\mathbb{R}$ is the residual of an inequality constraint in dependence of a single data point $(\boldsymbol{x}_d,y_d)\in\mathcal{D}$, and $\widetilde{h}:\mathcal{P}\to\mathbb{R}$ is the residual of a data-independent inequality constraint on the parameters. Note that we restrict ourselves to one constraint $h(\boldsymbol{x}_d,y_d;\cdot)$ and $\widetilde{h}$ only to keep the notation simple. All statements follow analogously for multiple constraints with the same general properties. In the theoretical considerations, we focus on continuous-valued optimization parameters $\boldsymbol{p}\in\mathbb{R}^n$ as done by Locatelli and Schoen (2013). In the numerical case study, we solve an MINLP. For this, we interpret "closed, bounded box" for unknown integer parameters $\boldsymbol{p}\in\mathbb{Z}^{n_d}$ as subsequent discrete values. While the general model formulation (PE) allows for error-prone data $\mathcal{D}$, we use data generated by the model, i.e., data with $f(\boldsymbol{x}_d;\boldsymbol{p}^*)=y_d$ for any $(\boldsymbol{x}_d,y_d)\in\mathcal{D}$ for optimal parameter values $\boldsymbol{p}^*$, for a first case study. Please refer to the supplementary material for more details. Throughout, $|\cdot|$ denotes the cardinality, $\mathbb{E}[\cdot]$ the expected value, and $\mathbb{P}(\cdot)$ the probability of their arguments.

Returning to theory, we make the following assumptions.

Assumption 1.
Let $f(\boldsymbol{x}_d;\cdot):\mathcal{P}\to\mathbb{R}$ and $h(\boldsymbol{x}_d,y_d;\cdot):\mathcal{P}\to\mathbb{R}$ for any fixed $(\boldsymbol{x}_d,y_d)\in\mathcal{D}$ as well as $\widetilde{h}:\mathcal{P}\to\mathbb{R}$ be continuous.

For conciseness of notation, we denote the squared deviation between measurements and model predictions by $g(\boldsymbol{p};\boldsymbol{x}_d,y_d):=\left(f(\boldsymbol{x}_d;\boldsymbol{p})-y_d\right)^2$ for any $d\in\{1,\dots,|\mathcal{D}|\}$, $\boldsymbol{p}\in\mathcal{P}$ and the summed squared error (SE) of model predictions and measurements for any set $\mathcal{D}_\mathrm{r}\subseteq\mathcal{D}$ by $g_{\mathcal{D}_\mathrm{r}}(\boldsymbol{p})=\sum_{(\boldsymbol{x}_d,y_d)\in\mathcal{D}_\mathrm{r}} g(\boldsymbol{p};\boldsymbol{x}_d,y_d)$.

As the key idea behind the proposed extension of a B&B algorithm, we split the overall error into the error terms of each single data point. Consequently, we add another assumption.

Assumption 2.

(i) Let $g^{\mathrm{cv}}(\cdot;\boldsymbol{x}_d,y_d)$ be any nonnegative convex underestimator of $g(\cdot;\boldsymbol{x}_d,y_d)$ over $\mathcal{P}$ for any fixed $(\boldsymbol{x}_d,y_d)\in\mathcal{D}$.

(ii) Let $h^{\mathrm{cv}}(\boldsymbol{x}_d,y_d;\cdot)$ and $\widetilde{h}^{\mathrm{cv}}$ be any convex underestimator of $h(\boldsymbol{x}_d,y_d;\cdot)$ for any fixed $(\boldsymbol{x}_d,y_d)\in\mathcal{D}$ and $\widetilde{h}$, respectively, over $\mathcal{P}$.

For any set $\mathcal{D}_\mathrm{r}\subseteq\mathcal{D}$, we define $g^{\mathrm{cv}}_{\mathcal{D}_\mathrm{r}}(\boldsymbol{p}):=\sum_{(\boldsymbol{x}_d,y_d)\in\mathcal{D}_\mathrm{r}} g^{\mathrm{cv}}(\boldsymbol{p};\boldsymbol{x}_d,y_d)$ and obtain the following Lemma.

Lemma 1. *Let Assumptions 1 and 2 hold. Then,*

(i) *$g^{cv}_{\mathcal{D}_r}(\boldsymbol{p})$ is a convex underestimator of both $g_{\mathcal{D}_r}$ and $g_{\mathcal{D}}$ over $\mathcal{P}$ for any $\mathcal{D}_r\subseteq\mathcal{D}$,*

(ii) *a lower bound on the globally optimum solution of (PE) is given for any $\mathcal{D}_r\subseteq\mathcal{D}$ by*

$$\begin{aligned} \min_{\boldsymbol{p}\in\mathcal{P}} \; & g^{cv}_{\mathcal{D}_r} \\ s.t. \; & h^{cv}(\boldsymbol{x}_d,y_d;\boldsymbol{p})\le 0 \qquad \forall(\boldsymbol{x}_d,y_d)\in\mathcal{D}_r \\ & \widetilde{h}^{cv}(\boldsymbol{p})\le 0\,, \end{aligned} \tag{1}$$

(iii) *$g_{\mathcal{D}}(\boldsymbol{p})$ is an upper bound on the globally optimum solution of (PE) for any $\boldsymbol{p}\in\mathcal{P}$ satisfying all inequality constraints.*

Subroutine 1: Augment or branch at iteration k

Input: Set of active nodes $\mathcal{N}_k$, set of fathomed nodes $\mathcal{N}^{\mathrm{f}}_k$, currently processed node $N_k = (\mathcal{P}_k, \mathcal{D}_k)$, augmentation rule $\mathbb{A}$, augmentation size φ, full dataset $\mathcal{D}$

Output: Updated sets $\mathcal{N}_{k+1}$ and $\mathcal{N}^{\mathrm{f}}_{k+1}$

if $\mathbb{A}(N_k) =$ *True* **then**
  Choose $\mathcal{D}_{\mathrm{new}} \subseteq \mathcal{D} \setminus \mathcal{D}_k$ such that $|\mathcal{D}_{\mathrm{new}}| = \min\{\lceil \varphi \cdot |\mathcal{D}| \rceil\ ,\ |\mathcal{D} \setminus \mathcal{D}_k|\}$;
  $N^{\mathrm{new}} := (\mathcal{P}_k, \mathcal{D}_k \cup \mathcal{D}_{\mathrm{new}})$;
  $\mathcal{N}_{k+1} := (\mathcal{N}_k \setminus N_k) \cup N^{\mathrm{new}}$;
else
  Branch N_k by partitioning $\mathcal{P}_k = \mathcal{P}_{k,1} \cup \mathcal{P}_{k,2} \cup \dots$;
  $N^{\mathrm{new},1} := (\mathcal{P}_{k,1}, \mathcal{D}_k)$, $N^{\mathrm{new},2} := (\mathcal{P}_{k,2}, \mathcal{D}_k)$, $\dots$;
  $\mathcal{N}_{k+1} := (\mathcal{N}_k \setminus N_k) \cup N^{\mathrm{new},1} \cup N^{\mathrm{new},2} \cup \dots$;
end
$\mathcal{N}^{\mathrm{f}}_{k+1} := \mathcal{N}^{\mathrm{f}}_k \cup N_k$;

The proof of Lemma 1 follows from our definitions and since the sum of convex underestimators is a convex underestimator of the sum for continuous functions. In particular, the nonnegativity of the squared error terms and of the convex underestimators as stated in Assumption 2i yields the statement in Lemma 1i. Note that applying $h^{\mathrm{cv}}(\boldsymbol{x}_d, y_d; \boldsymbol{p}) \leq 0$ for any $(\boldsymbol{x}_d, y_d) \in \mathcal{D}_{\mathrm{r}}$ just means to skip some constraints from (PE) if $\mathcal{D}_{\mathrm{r}} \subsetneq \mathcal{D}$. If lower bounding problem (1) is feasible, we denote an optimum point of (1) by $\boldsymbol{p}^{\mathrm{lb}}_{\mathcal{D}_{\mathrm{r}}}$.

With Lemma 1, we gain valid lower bounds on the solution of (PE) utilizing reduced datasets. Meaning, we can potentially save computational effort for the solution of (PE) by utilizing a smaller model for the calculations of convex relaxations to obtain convex problem (1), its linearization and the subsequent linear optimization within the lower bounding procedure. Similarly, we can obtain a valid upper based on a smaller model if we plug in a solution point from a local nonlinear optimization of the reduced problem into the original problem and this point turns out to be feasible for the full dataset as well. For the further analysis of lower and upper bounds, we denote with *full lower bound* and *reduced lower bound* the lower bound calculated based on the full dataset and based on a reduced dataset, respectively.

The downside of the reduced lower bound being valid is that it is potentially less tight than the full lower bound. If the gap between reduced and full lower bound prevents pruning, the B&B tree grows larger than necessary which may completely nullify the time savings due to data reduction. In fact, there are two reasons potentially preventing pruning when using a reduced dataset for lower bounding: (*i*) the full lower bound would also be smaller than the current upper bound, or (*ii*) the full lower bound would be larger than the current upper bound but the approximate reduced lower bound is not. In case (*i*), we want to branch the node to narrow down the region containing the optimal solution. In case (*ii*), we want to augment the dataset to decrease the gap between reduced and full lower bound. However, we only have information on the reduced lower bound at the beginning of the proposed B&B algorithm and, thus, do not know which case we are in. For the decision whether to branch or augment, we therefore introduce *augmentation rules*.

We associate all nodes of the B&B tree with a parameter domain and a dataset. An augmentation rule $\mathbb{A} : \mathcal{N} \to \{\mathrm{True}, \mathrm{False}\}$ decides in each B&B iteration k for the processed node $N_k = (\mathcal{P}_k, \mathcal{D}_k) \in \mathcal{N}$ whether to branch parameter domain $\mathcal{P}_k \subseteq \mathcal{P}$, which means adding child nodes whose parameter domains form a partition of domain $\mathcal{P}_k$ (see, e.g., Belotti et al., 2009; Horst & Tuy, 1996; Locatelli & Schoen, 2013; Tawarmalani & Sahinidis, 2002), or whether to augment dataset $\mathcal{D}_k \subseteq \mathcal{D}$, see Subroutine 1. Note that the child node inherits the parameter domain $\mathcal{P}_k$ of its parent node when augmenting. For the time being, we pick the additional data points $\mathcal{D}_{\mathrm{new}}$ randomly from $\mathcal{D} \setminus \mathcal{D}_k$. The number of data points to be added is determined by an a-priori given *augmentation size* $\varphi \in (0, 1]$ which gives the fraction of the full dataset to be added. For example, with $|\mathcal{D}| = 100$ and $\varphi = 0.25$ we would add 25% of the full dataset, i.e., 25 new data points, when augmenting. With an initial dataset containing of 10% of the available data points, the augmentation procedure would determine four different reduced datasets containing 10, 35, 60, and 85 data points. For the proof of convergence, we will use augmentation rules which eventually lead to the full dataset, i.e., which *complete finitely*.

Definition 1. Let $\mathcal{D}_k \subseteq \mathcal{D}$ be the dataset used in node N_k, which is processed in iteration k of the B&B algorithm with growing datasets depicted in Fig. 1.

The augmentation rule $\mathbb{A} : \mathcal{N} \to \{\mathrm{True}, \mathrm{False}\}$ *completes finitely*, if for any infinite nested sequence of nodes $\{N_{k_j}\}_{j\to\infty}$ with $N_{k_j} = (\mathcal{P}_{k_j}, \mathcal{D}_{k_j})$ it holds $\exists J < \infty : \mathcal{D}_{k_j} = \mathcal{D}\ \forall j \geq J$.

Note that we have $\mathcal{P}_{k_{j_2}} \subseteq \mathcal{P}_{k_{j_1}}$ and $\mathcal{D}_{k_{j_2}} \supseteq \mathcal{D}_{k_{j_1}}$ for any two nodes $N_{k_{j_1}}, N_{k_{j_2}}$ with $j_2 > j_1$ in an infinite nested sequence of nodes $\{N_{k_j}\}_{j\to\infty}$ since we either branch the parameter domain and keep the dataset or augment the dataset and keep the parameter domain in each iteration of the B&B algorithm.

2.1. Augmentation rule CONST

An intuitive augmentation rule is to augment every cth iteration of the B&B algorithm with growing datasets, where $c \in \mathbb{N}$ is an a-priori given constant. However, depending on the node selection strategy, such a rule would pick nodes at different depths within the B&B tree for augmenting. In the absence of additional information, we therefore augment if the depth of the current node is a multiple of constant c instead. We obtain an augmentation rule which completes finitely.

Proposition 1. *Let an initial dataset $\mathcal{D}_0 \subseteq \mathcal{D}$ be given. If the augmentation step defined in Subroutine 1 is performed with a constant augmentation size $\varphi \in (0, 1]$, then augmentation rule*

$$\mathbb{A}_{\mathrm{CONST}}(N_k) := \begin{cases} \mathit{True}, & \mathit{if}\ \frac{\mathit{depth}(N_k)}{c} \in \mathbb{Z} \\ \mathit{False}, & \mathit{else} \end{cases}$$

with an a-priori known constant $c \in \mathbb{Z}$ completes finitely.

Proof. Let $\{N_{k_j}\}_{j\to\infty}$ with $N_{k_j} = (\mathcal{P}_{k_j}, \mathcal{D}_{k_j})$ be a sequence of nested nodes generated by the B&B algorithm with growing datasets. By definition, we have $\mathrm{depth}(N_{k_j}) = j$ for any sequence of nested nodes starting with the root node.

We need $l := \lceil \frac{|\mathcal{D}| - |\mathcal{D}_0|}{\lceil \varphi \cdot |\mathcal{D}| \rceil} \rceil$ subsequent augmentations to reach the full dataset. According to augmentation rule $\mathbb{A}_{\mathrm{CONST}}$, augmentation is triggered every cth depth. Thus, we obtain the full dataset in nodes with depth $l \cdot c$, i.e., for any $j \geq J$ with $J := l \cdot c$. □

2.2. Augmentation rule SCALING

The more sophisticated augmentation rule SCALING aims to branch if we could have pruned the node when using the full lower bound. As we do not know the full lower bound, we estimate the full lower bound by scaling the reduced lower bound. In particular, we assume in this section that a solution of the lower bounding problem (1) exists, i.e., that there is a feasible point in this node. Otherwise, the node will be pruned without augmenting or branching.

As indicated by our results in Sass et al. (2023), we assume that the mean squared error does not change significantly when changing the dataset

$$\frac{1}{|\mathcal{D}_{\mathrm{r}}|} \sum_{(\boldsymbol{x}_d, y_d) \in \mathcal{D}_{\mathrm{r}}} g^{\mathrm{cv}}(\boldsymbol{p}; \boldsymbol{x}_d, y_d) \approx \frac{1}{|\mathcal{D}|} \sum_{(\boldsymbol{x}_d, y_d) \in \mathcal{D}} g^{\mathrm{cv}}(\boldsymbol{p}; \boldsymbol{x}_d, y_d) \tag{2}$$

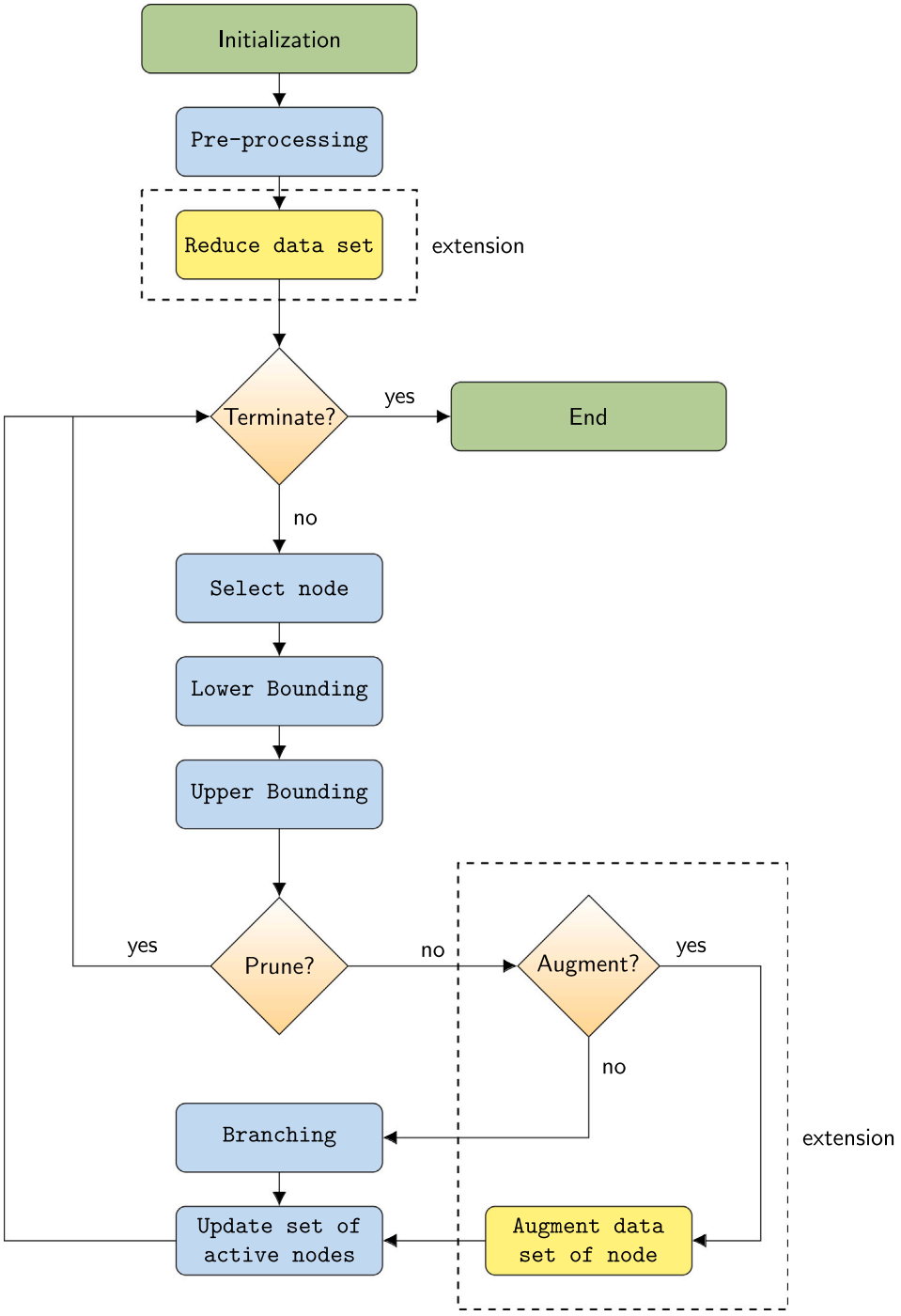


Fig. 1. Flow chart of the proposed algorithm, where the dashed boxes highlight the extensions of a common B&B algorithm to manage growing datasets.

or fulfills at least a sublinear relation

$$\frac{1}{|D_{\mathrm{r}}|}\sum_{(\boldsymbol{x}_d,y_d)\in D_{\mathrm{r}}} g^{\mathrm{cv}}(\boldsymbol{p};\boldsymbol{x}_d,y_d) \leq \frac{1}{|D|}\sum_{(\boldsymbol{x}_d,y_d)\in D} g^{\mathrm{cv}}(\boldsymbol{p};\boldsymbol{x}_d,y_d)\,. \tag{3}$$

We obtain the rule

$$\mathbb{A}^{\star}_{\text{SCALING}}(N_k) := \begin{cases} \text{True}, & \text{if } \frac{|D|}{|D_{\mathrm{r}}|}\cdot g^{\mathrm{cv}}_{D_{\mathrm{r}}}(\boldsymbol{p}^{\mathrm{lb}}_{D_{\mathrm{r}}}) \geq ub_k - \varepsilon \\ \text{False}, & \text{else,} \end{cases} \tag{4}$$

where ub_k is the best upper bound found until iteration k of the B&B algorithm which is also used for pruning in node N_k. Note that both assumptions (2) and (3) may be violated, particularly in models sensitive to single data points. Therefore, the condition used in SCALING* is a heuristic.

For a fixed dataset D, see Assumption 3, we can show that augmenting via SCALING is most likely triggered only if we could prune based on the full dataset. Note that dataset D may still contain deviations from

Table 1
Datasets, corresponding optimal objective and scaled optimal objective for the estimation problem described in Example 1.

Data points	Opt. obj.	Opt. obj. scaled with $\frac{\lvert D\rvert}{\lvert D_r\rvert}$
$D_0 = \{(0, 0.6)\}$	0.0	$3 \cdot 0 = 0$
$D_1 = \{(0, 0.6), (0, 1)\}$	0.08	$\frac{3}{2} \cdot 0.08 = 0.12$
$D = \{(0, 0), (0, 0.6), (0, 1)\}$	0.5067	0.5067

the model predictions with the true model parameters. However, in this article, we do not account for any random behavior in the data.

Assumption 3.

(i) The full dataset D is non-random, i.e., D is a fixed set and not considered as a random sample.
(ii) Let $D_r \subsetneq D$ be a reduced dataset with an a-priori fixed size, where the data points in D_r are picked randomly from D such that D_r follows a discrete uniform probability distribution over all subsets of D with size $|D_r|$.

Lemma 2. *If $D_r \subsetneq D$ is chosen according to Assumption 3, then*

$$\mathbb{E}\left[g^{cv}_{D_r}(\boldsymbol{p}^{lb}_{D_r})\right] \leq \frac{|D_r|}{|D|} g^{cv}_{D}(\boldsymbol{p}^{lb}_{D}).$$

Proof. The monotonicity of the expected value and the optimality of $\boldsymbol{p}^{\text{lb}}_{D_r}$ yield $\mathbb{E}\left[g^{\text{cv}}_{D_r}(\boldsymbol{p}^{\text{lb}}_{D_r})\right] \leq \mathbb{E}\left[g^{\text{cv}}_{D_r}(\boldsymbol{p}^{\text{lb}}_{D})\right]$. From the discrete uniform distribution assumption, it follows for any single data point $(\boldsymbol{x}_d, y_d) \in D$ that $\mathbb{P}((\boldsymbol{x}_d, y_d) \in D_r) = \frac{|D_r|}{|D|}$. In combination with the linearity of expectation, we obtain $\mathbb{E}\left[g^{\text{cv}}_{D_r}(\boldsymbol{p}^{\text{lb}}_{D})\right] = \frac{|D_r|}{|D|} g^{\text{cv}}_{D}(\boldsymbol{p}^{\text{lb}}_{D})$. □

Theorem 1. *Let Assumptions 1 and 2 hold and $\varepsilon > 0$ be given. Let $D_r \subsetneq D$ be chosen according to Assumptions 3. Assume augmentation rule*

$$\mathbb{A}_{\text{SCALING}}(N) := \begin{cases} \textit{True}, & \textit{if } \rho \cdot \frac{|D|}{|D_r|} \cdot g^{cv}_{D_r}(\boldsymbol{p}^{lb}_{D_r}) \geq ub - \varepsilon \\ \textit{False}, & \textit{else} \end{cases} \tag{5}$$

with upper bound ub computed for currently active node N, and a constant $0 < \rho \leq 1$.

If $\varepsilon < ub$ and $g^{cv}_{D} < ub - \varepsilon$, then the probability for augmenting is less than ρ.

Proof. Due to the nonnegativity of g^{cv}_{r} and $\varepsilon < ub \Leftrightarrow ub - \varepsilon > 0$, we can apply Markov's inequality

$$\begin{aligned}\mathbb{P}\left(\rho \cdot \frac{|D|}{|D_r|} \cdot g^{\text{cv}}_{D_r}(\boldsymbol{p}^{\text{lb}}_{D_r}) \geq ub - \varepsilon\right) &= \mathbb{P}\left(g^{\text{cv}}_{D_r}(\boldsymbol{p}^{\text{lb}}_{D_r}) \geq \frac{|D_r|}{|D|} \cdot \frac{1}{\rho} \cdot (ub - \varepsilon)\right) \\ &\leq \frac{\mathbb{E}\left[g^{\text{cv}}_{D_r}(\boldsymbol{p}^{\text{lb}}_{D_r})\right]}{\frac{|D_r|}{|D|}\frac{1}{\rho}(ub - \varepsilon)}.\end{aligned}$$

With Lemma 2 and $0 \leq g^{\text{cv}}_{D} < ub - \varepsilon$, the statement follows. □

Theorem 1 gives us a theoretical bound on the probability that $\mathbb{A}_{\text{SCALING}}$ triggered augmenting although we could not prune the node based on the full lower bound. Note that we introduced an *augmentation weight* ρ into $\mathbb{A}^{\star}_{\text{SCALING}}$ (4) to obtain $\mathbb{A}_{\text{SCALING}}$ (5) such that we can investigate the numerical behavior based on different bounds on the probability. Note further that condition $\varepsilon < ub$ is a weak assumption, since $\varepsilon \geq ub$ implies that the B&B algorithm has already converged if ε is the optimality tolerance.

We expect augmentation rule $\mathbb{A}_{\text{SCALING}}$ to be true sufficiently often to reach the full dataset. However, there are pathological cases where this augmentation rule does not complete finitely.

Example 1 (*Conflicting Data Prevents Convergence*)**.** We want to solve

$$\min_{a \in [0,25]} (a-1)^2 + (a-0.6)^2 + (a-0)^2 \ ,$$

where we find a linear function $f(x; a) = a \cdot x$ through the origin and three data points $D = \{(1, 0), (1, 0.6), (1, 1)\}$ at the same input $x = 1$, with a naive implementation of the B&B algorithm with growing datasets using augmentation rule $\mathbb{A}^{\star}_{\text{SCALING}}$, i.e., $\mathbb{A}_{\text{SCALING}}$ with $\rho = 1$. Assume that the (reduced) datasets are chosen as in Table 1. Even if the lower bound calculated based on reduced datasets D_0 and D_1 is exact, i.e., equals the respective optimal objective, gaps would remain $g^{\text{cv}}_{D_0} \ll g^{\text{cv}}_{D_1} \ll g^{\text{cv}}_{D} \leq g_D(\boldsymbol{p}^{\text{incumbent}})$. Branching does not help either since we already assume exact lower bounds. With that, neither augmenting nor convergence is possible. □

If the convergence of the B&B algorithm with growing datasets suffers from $\mathbb{A}_{\text{SCALING}}$ not completing finitely, we can use the hybrid augmentation rule $\mathbb{A}_{\text{SCALCST}}(N_k) := \left(\mathbb{A}_{\text{SCALING}}(N_k) \vee \mathbb{A}_{\text{CONST}}(N_k)\right)$. Note that constant c should be sufficiently large to not completely overrule the SCALING condition.

3. A B&B algorithm with growing datasets

Algorithm 1: B&B algorithm with growing datasets

Input: Model formulation including parameter domain $\mathcal{P}$ and full dataset D
Output: Global optimal solution including bounds on optimal objective value

1 Initialize counter $k := 0$, root node $N_0 := (\mathcal{P}, D)$ and set of active nodes $\mathcal{N}_0 := \{N_0\}$;
2 Pre-processing of root node N_0;
3 Pick initial dataset $D_0 \subsetneq D$ and reset dataset of root node $N_0 := (\mathcal{P}, D_0)$;
4 **while** $ub_k - lb_k >$ *optimality tolerance* **do**
5 Select next node $N_k \in \mathcal{N}_k$;
  // Dataset of active node N_k: $D_k \subseteq D$
6 Solve lower bounding problem in N_k;
7 Solve upper bounding problem in N_k and obtain point $\boldsymbol{p}^{ub}_{D_k}$;
8 Evaluate model at $\boldsymbol{p}^{ub}_{D_k}$ based on D and obtain upper bound;
9 Update upper bound over all active nodes ub_k;
10 **if** N_k *cannot be pruned* **then**
11 Call Subroutine 1;
12 **end**
13 Update lower bound over all active nodes lb_k;
14 $k := k + 1$;
15 **end**

Together with the components described in Section 2, we propose to extend the common procedure of a B&B algorithm by growing datasets as shown in Fig. 1. The preprocessing in the root node, e.g., bound tightening and multistart, is still based on the full dataset, while we start with a reduced dataset for lower bounding, compare also Algorithm 1. For the upper bounding, we compute a local optimum based on the reduced dataset and evaluate the corresponding solution point based on the full dataset to get a valid upper bound for (PE). Whether to augment or branch is determined based on the augmentation rules presented in Section 2. Note that both child nodes inherit the dataset from the parent node in branching, cf. Subroutine 1. As a first step, we propose a very simple setup for determining and augmenting the reduced datasets: the size of the initial dataset and the augmentation size φ are user settings, namely 10% and 25% of the full dataset, respectively, by default; the indices of the data points to be chosen are picked randomly for both the initialization and augmentation.

The B&B algorithm with growing datasets is a *deterministic* global optimization algorithm for solving Problem (PE) based on the full dataset, since we obtain valid lower and upper bounds, see Lemma 1. The heuristic augmentation rules are not used for pruning but only to

decide whether to augment. In fact, we can show that extending a convergent B&B algorithm by growing datasets preserves the convergence to the global optimum of the original problem formulation (PE).

We adapt the proof of convergence for a B&B algorithm subject to optimality and feasibility tolerances given in Theorem 5.26 of Locatelli and Schoen (2013). In fact, the proposed B&B algorithm with growing datasets introduces a dependency on the dataset for the lower and upper bounds. At first, we verify the crucial assumptions of Theorem 5.26 (Locatelli & Schoen, 2013).

Lemma 3. *Let Assumptions 1 and 2 hold. We apply a spatial B&B algorithm with optimality tolerance $\varepsilon > 0$ to (PE). Let the convex underestimator g_D^{cv} be exact in the limit (Locatelli & Schoen, 2013, Def. 5.4), cf. Appendix A. Let $g^{cv}(\cdot; \boldsymbol{x}_d, y_d)$ satisfy the so called isotonic property for any $(\boldsymbol{x}_d, y_d) \in D$ (Locatelli & Schoen, 2013, Eq. (5.24)), cf. Appendix A. Let the subdivision process of the B&B algorithm be exhaustive (Locatelli & Schoen, 2013, Def. 5.5), cf. Appendix A.*

If we use the B&B algorithm with growing datasets depicted in Fig. 1 with iteration index k, optimality tolerance $\varepsilon > 0$, and an augmentation rule $\mathbb{A}$ which completes finitely, then

(i) *each element of the generated sequence $(g_{D_k}^{cv})_k$ satisfies the isotonic property*
(ii) *each element of $(g_{D_k}^{cv})_k$ is a convex underestimator of g_D*
(iii) *each element of $(g_{D_k}^{cv})_k$ is exact in the limit regarding the full dataset*
(iv) *the subdivision process remains exhaustive.*

Proof.

(i) With increasing iteration index k, the dataset D_k can be augmented or remain constant. The statement follows with any $g^{cv}(\cdot; \boldsymbol{x}_d, y_d)$ being nonnegative and satisfying the isotonic property.
(ii) The statement follows from Lemma 1.
(iii) Without loss of generality, we focus on a sequence of nested nodes $\{N_{k_j}\}_{j\to\infty}$ generated by the B&B algorithm with growing datasets. Since the augmentation rule completes finitely, there exits $J < \infty : D_{k_j} = D \; \forall j \geq J$. Thus, $g_{D_{k_j}}^{cv} \equiv g_D^{cv}$ for any $j \geq J$. The statement follows with g_D^{cv} being exact in the limit.
(iv) If data augmentation is triggered, a child node is added which coincides with the currently active node except for the dataset. In particular, the parameter domain of the child node equals the parameter domain of its parent node.
If data augmentation is not triggered, the branching process of the original B&B algorithm is executed.
Thus, the subdivision process remains exhaustive. □

Note that the inequality constraints $h(\boldsymbol{x}_d, y_d; \cdot)$ and $\widetilde{h}$ are not affected by a change of the dataset. We just add more data-dependent constraints $h(\boldsymbol{x}_d, y_d; \cdot)$ when augmenting the dataset. Based on Lemma 3, we can extend the proof of Theorem 5.26 (Locatelli & Schoen, 2013) to growing datasets.

Theorem 2. *Let Assumptions 1 and 2 hold. We apply a spatial B&B algorithm with optimality tolerance $\varepsilon > 0$ to (PE). Let the convex underestimators g_D^{cv}, $h^{cv}(\boldsymbol{x}_d, y_d; \cdot) \; \forall (\boldsymbol{x}_d, y_d) \in D$, and $\widetilde{h}^{cv}$ be exact in the limit. Let $g^{cv}(\cdot; \boldsymbol{x}_d, y_d)$ and $h^{cv}(\boldsymbol{x}_d, y_d; \cdot)$ satisfy the isotonic property $\forall (\boldsymbol{x}_d, y_d) \in D$. Let the subdivision process of the B&B algorithm be exhaustive.*

Then the B&B algorithm with growing datasets depicted in Fig. 1 terminates after a finite number of iterations and

- *either establishes that the problem is infeasible if the final upper bound equals infinity*
- *or returns an ε-optimal solution if the final lower bound is finite.*

Proof. If the algorithm terminates:

- Case I: infinite final upper bound
  Since reducing the dataset results in a relaxation of (PE), the infeasibility of the model follows directly from the proof of Theorem 5.26 given in Locatelli and Schoen (2013).
- Case II: finite final upper bound
  Since we obtain valid lower and upper bounds even when using a reduced dataset, see Lemma 1, the ε-optimality follows directly from the proof of Theorem 5.26 given in Locatelli and Schoen (2013).

The finite convergence follows from the proof of Theorem 5.26 given in Locatelli and Schoen (2013) since

- the modified B&B algorithm generates valid upper and lower bounds for the model based on the full dataset, see Lemma 1,
- Lemma 3 gives exactness in the limit and exhaustiveness of the lower bounding scheme. □

Note that the B&B algorithm with growing datasets is only guaranteed to converge if the augmentation rule completes finitely. This is the case, e.g., for augmentation rule $\mathbb{A}_{\text{CONST}}$, where we reach the full dataset with a specific size of the B&B tree, cf. proof of Proposition 1. Otherwise overfitting or conflicting data may prevent augmenting and closing the optimality gap as in the pathological case for augmentation rule $\mathbb{A}_{\text{SCALING}}$ illustrated in Example 1.

In MAiNGO, only nodes with a parameter domain P_k that has a diameter of at least a user-given tolerance are added to prevent the cluster effect (see, e.g., Du & Kearfott, 1994). In cases of overfitting or conflicting data, MAiNGO may therefore terminate since no active nodes are left, although the optimality tolerance is not reached. Thus, we implemented the following heuristic: we add the node containing the best incumbent found so far with the dataset augmented to the full dataset D. If the solution point of the original problem, i.e., with considering the full dataset, is sufficiently close to the best incumbent calculated based on a reduced dataset, the global solution is contained in the added node. In that case, the B&B algorithm with growing datasets will converge successfully. Otherwise the B&B algorithm with growing datasets at least updates the lower bound based on the full dataset and reports therewith tighter bounds on the optimality gap when hitting another termination criterion like a CPU time limit.

4. Numerical results

In this section, we want to test the actual performance of our proposed algorithm. Many real-world parameter estimation problems minimize a summed squared error over 1000, 10 000 or more measurements (e.g., Alsmeyer et al., 2004; Kapfer et al., 2019; Lemmon et al., 2009). Contrarily, general benchmark libraries for mathematical programming like MINLPlib (Bussieck et al., 2003) and Princeton-Lib (Vanderbei & colleagues, 2004) offer only limited coverage of optimization problems in this format or only small problems. For example, the COCONUT benchmark (Shcherbina et al., 2003) contains a few instances with up to 35 data points. The largest instance named "gulf" (Cox, 1969; Moré et al., 1981) containing 99 data points can be solved within a second even without our extension. At the same time, the second largest instance with 65 data points, see "osborneb" or "Osborne 2" (Moré et al., 1981; Osborne, 1972), containing 11 optimization variables and redundant terms in the objective poses a huge challenge for the local upper bounding solver which is not tackled by our proposed algorithm. Benchmarks from medicine and biology (e.g., Villaverde et al., 2019) typically include large datasets but pose a huge challenge for the B&B algorithm itself due to their large number of unknown parameters.

To test the performance of our proposed algorithm nevertheless, we revisit the optimization problem for fitting the equation of state (EOS) of propane (Lemmon et al., 2009; Sass et al., 2023) and a model describing a kinetic metabolic pathway (Moles et al., 2003). When

Table 2
Total CPU time for convergence of different estimation problems in MAiNGO with a fixed dataset as well as with growing datasets and different augmentation rules.

		EOS2262	EOS262	TSP
Full dataset		22.34 h	111.1 min	217.7 min
CONST	$c = 5$	18.58 h	126.8 min	38.2 min
CONST	$c = 10$	13.61 h	62.1 min	2.3 min
SCALING	$\rho = 1$	4.53 h	23.1 min	39.2 min
SCALING	$\rho = 0.75$	4.48 h	24.3 min	111.9 min
SCALING	$\rho = 0.5$	4.70 h	16.7 min	4.0 min
SCALING	$\rho = 0.25$	4.46 h	22.3 min	2.0 min

Table 3
Number of nodes processed for convergence of different estimation problems by MAiNGO with a fixed dataset as well as with growing datasets and different augmentation rules.

		EOS2262	EOS262	TSP
Full dataset		101 148	110 901	636 794
CONST	$c = 5$	92 789	113 897	101 957
CONST	$c = 10$	101 113	129 732	7247
SCALING	$\rho = 1$	149 002	195 803	239 922
SCALING	$\rho = 0.75$	148 808	208 268	824 246
SCALING	$\rho = 0.5$	153 080	167 751	30 168
SCALING	$\rho = 0.25$	143 001	200 624	15 955

fitted accurately, the former allows for the optimal design and operation of processes using propane, e.g., as an ecological alternative for common fuels and refrigerants, and the latter for deeper insights into biochemical processes. In fact, we solve the EOS model with respect to 9 unknown parameters based on 262 and 2262 data points, calling it EOS262 and EOS2262, respectively. For the biochemical model, we adopt the name "TSP model" from Villaverde et al. (2019) and use one experiment of Moles et al. (2003) with 20 measurements for each of the 8 differential states to optimize 12 unknown parameters. Note that we use noise-free measurement data generated with the respective model. In particular, we do not account for the selection of data points regarding their individual experimental uncertainties and impact on the resulting model in this case study.

The B&B algorithm with growing datasets is available in our open-source DGO solver MAiNGO version 0.7.2. In the root node, we run 3 local searches with Ipopt version 3.12.12 (Wächter & Biegler, 2006) as a preprocessing. For the lower bounding, MAiNGO obtains relaxations from MC++ (Chachuat et al., 2015), applies an adaptation of Kelley's algorithm (Kelley, 1960; Najman et al., 2021) for linearization and invokes the linear solver CLP version 1.17.0 (Forrest et al., 2004) to solve the resulting linear program. For the upper bounding, the local solver LBFGS (Liu & Nocedal, 1989; Nocedal, 1980) implemented in the NLOPT toolbox v2.5.0 (Johnson, 2007) was used. We investigate the augmentation rules CONST with default setting $c = 10$ and SCALING with different values for augmentation weight ρ independently rather than using the default rule SCALCST. All calculations are performed on an Intel(R) Core(TM) i5-3570 processor with 3.4 GHz. Since the quality of the local solutions is very sensitive to the initial solution chosen and the number of steps performed in the local optimization of the upper bounding problem, we limit the local optimization procedure in both preprocessing and upper bounding by a maximum number of iterations rather than a CPU time limit to get deterministic and comparable results. Refer to the supplementary material for the exact settings and data points used as well as the problem formulation in the syntax of MAiNGO and GAMS (Bussieck et al., 2003).

Table 2 summarizes the total CPU time for solving models EOS262, EOS2262, and TSP with the full dataset, i.e., the common B&B algorithm, and with growing datasets to global optimality. These results follow our expectations based on the provided theory: As indicated by the discussion around Theorem 1, augmentation rule SCALING is the most appropriate choice for quick convergence in most of the cases. In detail, we save about 80% of the CPU time independent of the augmentation weight ρ for models EOS2262 and EOS262. For the TSP model, we can even decrease the CPU time by a factor of 100 when choosing SCALING with $\rho = 1$. The use of augmentation rule CONST is more ambiguous. While we can decrease the runtime for the TSP model by a factor of 100 when using CONST with default setting $c = 10$, the runtime for EOS262 increases with $c = 5$. For EOS262 with $c = 5$, the full dataset is reached before MAiNGO could prune parts of the B&B tree based on the reduced dataset. In that way, growing datasets just increase the size of the B&B tree. Note that Proposition 1 and Theorem 2 give a theoretical guarantee for convergence when using augmentation rule CONST, but do not make statements about the speed of convergence.

The time needed for convergence depends on (*i*) the processing time for a node and (*ii*) the number of nodes needed for convergence. The strength of the B&B algorithm with growing datasets lies in the significant reduction of the CPU time per node. When comparing Table 2 and 3, we can see, e.g., for EOS2262 that the B&B algorithm processes about 150 000 nodes in less than 5 h when using reduced datasets with augmentation rules SCALING while it processes only about 100 000 in 22 h with the full dataset. For (*ii*), the tightness of the lower and upper bounds plays a key role. The upper bound depends on the actual local solution found. We calculate the solution point of the upper bounding problem based on the reduced dataset, see Algorithm 1, and, in MAiNGO, the solution point of the lower bounding problem is used for initializing the upper bounding solver. By chance, a reduced dataset may therefore give a better solution and lead to much faster convergence as can be seen for the TSP model with CONST $c = 10$ and SCALING $\rho \in \{0.25, 0.5\}$. Note that the lower bound remains at its natural lower bound of 0, when using noise-free measurement data as in this case study. When using noisy data, the tightness of the lower bound may suffer from the data reduction, which is the downside of Lemma 1.

For all scenarios, the optimal objective value, i.e., the final upper bound, is between the natural lower bound of 0 and the optimality tolerance of $\epsilon = 0.01$. In fact, MAiNGO computes for the EOS model solutions close to those of Lemmon et al. (2009) which were used to generate the data points. Only for parameters γ_{10} and ϵ_{10} there are significant differences. Note that these two parameters have a marginal impact on the model prediction, especially compared to the other parameters n_{10}, t_{10}, d_{10}, c_{10}, l_{10}, η_{10}, and β_{10}. For the TSP model, all optimal parameter values determined by MAiNGO differ significantly from the nominal values of Moles et al. (2003). However, these differences are not tracked by the objective function since they occur between the initial date and the first measurement value at $t = 6$, see the different solution trajectories of state G_2 in Fig. 2. In other words, the parameter estimation problem is still solved correctly based on the given dataset. The exact parameter values for both models are given in the supplementary material.

MAiNGO can handle the reduced-space formulation well which we used for both the EOS and the TSP model, refer to supplementary material for more details. In particular, we solve an optimization problem with 9 and 12 optimization variables, respectively, independent of the size of the dataset. In contrast to that, AVM-based solvers like BARON will typically introduce one or more optimization variables for each data point. To get a first impression on potential CPU time savings when extending an AVM-based B&B algorithm by growing datasets, we use the feature for adding auxiliary variables in MAiNGO (Najman et al., 2021) for the EOS model: for each mathematical expression that occurs at least two times in the model formulation, an auxiliary optimization variable is added to the lower bounding problem. In case of EOS2262 49 auxiliary variables are added, while 17 auxiliary variables are added for EOS262.

For different datasets, we measure the CPU time required for the different steps performed in each node comprising a preprocessing step for bound tightening, solving the lower bounding problem, solving the upper bounding problem, and a postprocessing step for bound-tightening. The CPU time spend for a node is mainly determined by the preprocessing step, which includes bound tightening via constraint

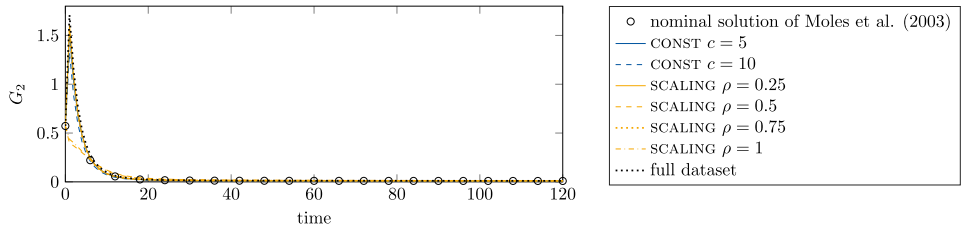
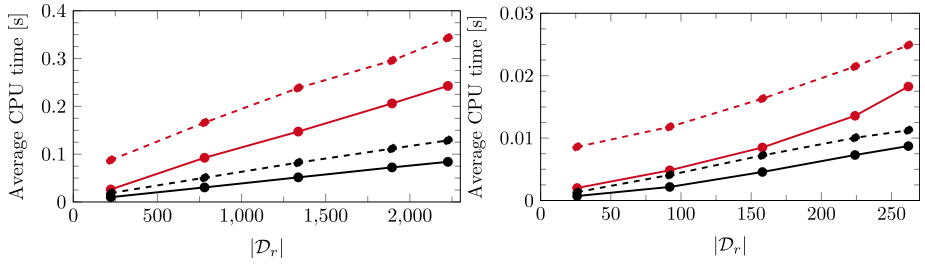


Fig. 2. Integration results for state G_2 of the TSP model in dependence of the different optimal values for the unknown parameters.



(a) EOS2262 (b) EOS262

Fig. 3. Average CPU time spent by MAiNGO with growing datasets in a node for preprocessing (•) and solving the lower bounding problem (•) without (solid lines) and with auxiliary variables (dashed lines) when clustering the nodes by the size of their associated dataset.

propagation and optimality-based bound tightening (Gleixner et al., 2017), and the lower bounding step. As can be seen in Fig. 3 , the average CPU time per node for both preprocessing and solving the lower bounding problem increases with increasing size of the dataset. Even without auxiliary variables, an increasing size of the dataset implies an increasing effort for evaluating data-dependent functions and their subgradients. The addition of auxiliary variables shifts the CPU times to larger values. In MAiNGO, the number of optimization variables within the B&B algorithm remains the same for each dataset even when using the auxiliary variable feature. Thus, we expect both a shift and a larger slope of the curves when adding one or more auxiliary variables per data point. In other words, we expect AVM-based implementations of the B&B algorithm to profit even more from including growing datasets. Note that the seemingly small CPU time differences add up to the significant time savings observed in Table 2 since the B&B algorithm often requires the processing of multiple thousands of nodes for convergence, e.g., more than 100 000 nodes in case of EOS262 and EOS2262 when using the fixed full dataset.

5. Conclusions

We propose a B&B algorithm with growing datasets for solving parameter estimation problems subject to large measurement datasets. In fact, we start with a significantly reduced dataset which is augmented automatically within the B&B procedure until the full dataset is reached. We have proven for NLPs that this extension preserves the convergence of the underlying B&B algorithm to a global optimum of the original problem based on the full dataset.

For augmenting, we presented the intuitive decision rule CONST as well as the more sophisticated rule SCALING. We have shown that the latter augments most likely if we could have pruned the currently processed node based on the full dataset. This is accompanied by a numerical case study based on the equation of state of propane (Lemmon et al., 2009) and a model for metabolic pathways (Moles et al., 2003; Villaverde et al., 2019) with noise-free synthetic measurement data. The total CPU time for DGO can be reduced by about 80% when using the augmentation rule SCALING for the former, and by a factor of 100 with specific settings for augmentation rules CONST and SCALING for the latter.

In our case study, we consider simulated noise-free data allowing for a perfect fit. In particular, the lower bound equals 0 except for numerical tolerances independent of the dataset used. For any datasets that cannot be fitted perfectly, e.g., due to measurement errors, uncertainties or a wrong model structure, there may be a significant gap between the objective of the estimation problem based on the reduced dataset and the one based on the full dataset. This may prevent pruning based on the reduced dataset and even augmenting according to the rule SCALING. Although we can enforce augmenting by using the augmentation rules CONST or SCALCST, we cannot enforce pruning. In the worst case, the B&B tree will only grow as long as reduced datasets are used. In this case, we cannot expect any CPU time savings by using growing datasets. Hence, we will study a heuristic B&B algorithm with growing datasets in future, which will use an approximation of the full lower bound based on the reduced lower bound for pruning rather than only for augmenting.

In future, the choice of the datasets needs to be revisited as well. On the one hand, heuristic knowledge for choosing the size and data points of training sets can be transferred from machine learning to the choice of the initial datasets in our approach. On the other hand, the number of data points to be added in an augmentation set, what we called augmentation size, may be dynamically adapted within the B&B algorithm with growing datasets: e.g., we could start with small sets and decide based on the lower bounds before and after augmenting whether to increase the augmentation size. Additionally, heuristic knowledge about the sensitivity of the solution to specific (subsets of) data points from stochastic optimization or machine learning may improve the efficiency of the proposed augmentation step. Similarly, model-specific knowledge about the uncertainty of specific (subsets of) data points or intrinsic clusters of the data points, e.g., data points representing liquid and gas phase in thermodynamic models, may be used.

Finally, we want to emphasize the need to extend open benchmark libraries by parameter estimation problems. While general benchmark libraries for mathematical programming have limited coverage of this class of optimization problems or do only account for small datasets, benchmark libraries with medical and biological parameter estimation problems often contain only large dynamic optimization problems with about 30, 100, or more unknown parameters which cannot be handled by DGO solvers, yet. Larger benchmark models gain importance in

future when the combination of tailored algorithms, e.g., for large-scale parameter estimation as the approach proposed herein or for DGO of dynamic optimization problems (e.g., Esposito & Floudas, 2000; Papamichail & Adjiman, 2002; Singer & Barton, 2006), together with the emerging parallelization of optimization software may significantly extend the capabilities of DGO solvers.

Credit authorship contribution statement

Susanne Sass: Conceptualization, Methodology, Software, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization, Project administration, Funding acquisition. Alexander Mitsos: Conceptualization, Methodology, Resources, Writing - Review & Editing, Supervision, Project administration, Funding acquisition. Dominik Bongartz: Methodology, Software, Writing - Review & Editing. Ian H. Bell: Methodology, Resources, Writing - Review & Editing. Nikolay I. Nikolov: Methodology, Writing - Review & Editing. Angelos Tsoukalas: Conceptualization, Methodology, Writing - Review & Editing, Funding acquisition.

Acknowledgments

Funding: This work was partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under grant MI 1851/10-1 “Parameter estimation with (almost) deterministic global optimization” as well as by the National Institute of Standards and Technology.

Susanne Sass is grateful for her association to the International Research Training Group (DFG) IRTG-2379 “Hierarchical and Hybrid Approaches in Modern Inverse Problems” under grant 333849990/GRK2379. We thank Tanvir Rahat for his support for setting up the “TSP model”.

Commercial equipment, instruments, or materials are identified only in order to adequately specify certain procedures. In no case does such identification imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the products identified are necessarily the best available for the purpose.

Appendix A. Basic definitions used in the proof of convergence

In this section, we repeat the definitions of Locatelli and Schoen (2013) with our notation, see Section 3. For this, we denote the feasible set of the node $N_k = (\mathcal{P}_k, \mathcal{D}_k)$ with associated parameter domain $\mathcal{P}_k \subsetneq \mathbb{R}^n$ and dataset $\mathcal{D}_k \subsetneq \mathbb{R}^m \times \mathbb{R}$ processed in B&B iteration k by $\mathcal{F}_k := \mathcal{P}_k \cap \{\boldsymbol{p} \in \mathcal{P} : \ h(\boldsymbol{x}_d, y_d; \boldsymbol{p}) \leq 0 \ \forall (\boldsymbol{x}_d, y_d) \in \mathcal{D} \ \wedge \ \widetilde{h}(\boldsymbol{p}) \leq 0\}$, where $h(\boldsymbol{x}_d, y_d;)$ and $\widetilde{h}$ are the residuals of inequality constraints.

Definition 2 (*Equation (5.24) of Locatelli & Schoen, 2013*)**.**
An underestimator $F^{\mathrm{cv}} : \mathcal{P} \to \mathbb{R}$ satisfies the *isotonic property* if $\mathcal{F}_l \subseteq \mathcal{F}_k \ \Rightarrow \ F^{\mathrm{cv}}(\boldsymbol{p})|_{\boldsymbol{p}\in\mathcal{F}_l} \geq F^{\mathrm{cv}}(\boldsymbol{p})|_{\boldsymbol{p}\in\mathcal{F}_k} \ \ \forall \boldsymbol{p} \in \mathcal{F}_l$.

Definition 3 (*Definition 5.4 of Locatelli & Schoen, 2013*)**.** A convex underestimator $F^{\mathrm{cv}} : \mathcal{P} \to \mathbb{R}$ of a function F over some region $\mathcal{F}_k$ is *exact in the limit* if $\max_{\boldsymbol{p}\in\mathcal{F}_k}\{F(\boldsymbol{p}) - F^{\mathrm{cv}}(\boldsymbol{p})\} \leq \eta(\mathrm{diam}(\mathcal{F}_k))$, where $\mathrm{diam}(\mathcal{F}_k) = \max_{\boldsymbol{p}_1,\boldsymbol{p}_2\in\mathcal{F}_k} \|\boldsymbol{p}_1 - \boldsymbol{p}_2\|_2$ is the diameter of $\mathcal{F}_k$, and $\eta(r)$ is a continuous nondecreasing function such that $\lim_{r\to 0}\eta(r) = 0$.

Definition 4 (*Definition 5.5 of Locatelli & Schoen, 2013*)**.** A B&B algorithm equipped with a geometric branching rule possesses the *exhaustiveness* property if each infinite nested sequence generated by the algorithm $(\mathcal{F}_{k_j})_j \ : \ \mathcal{F}_{k_0} = \mathcal{F}_0, \ k_j < k_{j+1}, \ \mathcal{F}_{k_{j+1}} \subseteq \mathcal{F}_{k_j} \ \forall j = 0, 1, \ldots$ converges to a singleton $\mathrm{diam}(\mathcal{F}_{k_j}) \to 0$ as $j \to \infty$.

Appendix B. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.ejor.2024.02.020.

References

Abbiw-Jackson, R., Golden, B., Raghavan, S., & Wasil, E. (2006). A divide-and-conquer local search heuristic for data visualization. *Computers & Operations Research*, *33*(11), 3070–3087. http://dx.doi.org/10.1016/j.cor.2005.01.020.

Agosti, A., Ciarletta, P., Garcke, H., & Hinze, M. (2020). Learning patient–specific parameters for a diffuse interface glioblastoma model from neuroimaging data. *Mathematical Methods in the Applied Sciences*, *43*(15), 8945–8979. http://dx.doi.org/10.1002/mma.6588.

Almquist, J., Cvijovic, M., Hatzimanikatis, V., Nielsen, J., & Jirstrand, M. (2014). Kinetic models in industrial biotechnology – improving cell factory performance. *Metabolic Engineering*, *24*, 38–60. http://dx.doi.org/10.1016/j.ymben.2014.03.007.

Alsmeyer, F., Koss, H.-J., & Marquardt, W. (2004). Indirect spectral hard modeling for the analysis of reactive and interacting mixtures. *Applied Spectroscopy*, *58*(8), 975–985. http://dx.doi.org/10.1366/0003702041655368.

Belotti, P., Lee, J., Liberti, L., Margot, F., & Waechter, A. (2009). Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods and Software*, *24*(4–5), 597–634. http://dx.doi.org/10.1080/10556780903087124.

Beyhaghi, P., Alimo, R., & Bewley, T. (2020). A derivative-free optimization algorithm for the efficient minimization of functions obtained via statistical averaging. *Computational Optimization and Applications*, *76*(1), 1–31. http://dx.doi.org/10.1007/s10589-020-00172-4.

Bongartz, D., & Mitsos, A. (2017). Deterministic global optimization of process flowsheets in a reduced space using McCormick relaxations. *Journal of Global Optimization*, *69*(4), 761–796. http://dx.doi.org/10.1007/s10898-017-0547-4.

Bongartz, D., Najman, J., Sass, S., & Mitsos, A. (2018). *MAiNGO – McCormick-based Algorithm for mixed-integer Nonlinear Global Optimization*. Process Systems Engineering (AVT.SVT), RWTH Aachen University, URL http://permalink.avt.rwth-aachen.de/?id=729717 (Accessed 12 Jan 2023).

Boukouvala, F., Misener, R., & Floudas, C. A. (2016). Global optimization advances in mixed-integer nonlinear programming, MINLP, and constrained derivative-free optimization, CDFO. *European Journal of Operational Research*, *252*(3), 701–727. http://dx.doi.org/10.1016/j.ejor.2015.12.018.

Bussieck, M. R., Drud, A. S., & Meeraus, A. (2003). MINLPLib—A collection of test models for mixed-integer nonlinear programming. *INFORMS Journal on Computing*, *15*(1), 114–119. http://dx.doi.org/10.1287/ijoc.15.1.114.15159.

Byrd, R. H., Hansen, S. L., Nocedal, J., & Singer, Y. (2016). A stochastic quasi-Newton method for large-scale optimization. *SIAM Journal on Optimization*, *26*(2), 1008–1031. http://dx.doi.org/10.1137/140954362.

Chachuat, B., Houska, B., Paulen, R., Perić, N., Rajyaguru, J., & Villanueva, M. E. (2015). Set-theoretic approaches in analysis, estimation and control of nonlinear systems. *IFAC-PapersOnLine*, *48*(8), 981–995, URL https://github.com/omega-icl/mcpp, Retrieved Nov 11, 2019.

Cox, R. A. (1969). Comparison of the performance of seven optimization algorithms on twelve unconstrained optimization problems. ref. 1335cn04. Gulf Research and Development Company, Pittsburg.

Du, K., & Kearfott, R. B. (1994). The cluster problem in multivariate global optimization. *Journal of Global Optimization*, *5*(3), 253–265. http://dx.doi.org/10.1007/BF01096455.

Egea, J. A., Martí, R., & Banga, J. R. (2010). An evolutionary method for complex-process optimization. *Computers & Operations Research*, *37*(2), 315–324. http://dx.doi.org/10.1016/j.cor.2009.05.003.

Esposito, W. R., & Floudas, C. A. (2000). Global optimization for the parameter estimation of differential-algebraic systems. *Industrial & Engineering Chemistry Research*, *39*(5), 1291–1310. http://dx.doi.org/10.1021/ie990486w.

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, *24*(6), 381–395. http://dx.doi.org/10.1145/358669.358692.

Floudas, C. A. (2000). *Deterministic global optimization: theory, methods and applications, vol. 37*. Springer US, http://dx.doi.org/10.1007/978-1-4757-4949-6.

Forrest, J., de La Nuez, D., & Lougee-Heimer, R. (2004). CLP user guide. IBM Corporation.

Gau, C. Y., & Stadtherr, M. A. (2002). Deterministic global optimization for error-in-variables parameter estimation. *AIChE Journal*, *48*(6), 1192–1197. http://dx.doi.org/10.1002/aic.690480607.

Gleixner, A. M., Berthold, T., Müller, B., & Weltge, S. (2017). Three enhancements for optimization-based bound tightening. *Journal of Global Optimization*, *67*(4), 731–757. http://dx.doi.org/10.1007/s10898-016-0450-4.

Horst, R., & Pardalos, P. M. (1995). *Handbook of global optimization, vol. 2*. Springer US, http://dx.doi.org/10.1007/978-1-4615-2025-2.

Horst, R., & Tuy, H. (1996). *Global optimization: deterministic approaches* (third revised and enlarged edn). Berlin, Heidelberg and s.l.: Springer Berlin Heidelberg, http://dx.doi.org/10.1007/978-3-662-03199-5.

Johnson, S. G. (2007). The NLopt nonlinear-optimization package. URL http://github.com/stevengj/nlopt, Retrieved Nov 06, 2019.

Kapfer, E.-M., Stapor, P., & Hasenauer, J. (2019). Challenges in the calibration of large-scale ordinary differential equation models. *IFAC-PapersOnLine*, *52*(26), 58–64. http://dx.doi.org/10.1016/j.ifacol.2019.12.236.

Kelley, J. E. (1960). The cutting-plane method for solving convex programs. *Journal of the society for Industrial and Applied Mathematics*, *8*(4), 703–712. http://dx.doi.org/10.1137/0108053.

Lemmon, E. W., McLinden, M. O., & Wagner, W. (2009). Thermodynamic properties of propane. III. A reference equation of state for temperatures from the melting line to 650 k and pressures up to 1000 MPa. *Journal of Chemical & Engineering Data*, *54*(12), 3141–3180. http://dx.doi.org/10.1021/je900217v.

Liu, D. C., & Nocedal, J. (1989). On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, *45*(1–3), 503–528. http://dx.doi.org/10.1007/BF01589116.

Liu, M. L., & Sahinidis, N. V. (1997). Process planning in a fuzzy environment. *European Journal of Operational Research*, *100*(1), 142–169. http://dx.doi.org/10.1016/S0377-2217(96)00025-2.

Locatelli, M., & Schoen, F. (2013). *MOS-SIAM series on optimization*, *Global optimization: theory, algorithms, and applications, vol. 15*. http://dx.doi.org/10.1137/1.9781611972672.

McCormick, G. P. (1976). Computability of global solutions to factorable nonconvex programs: Part I – convex underestimating problems. *Mathematical Programming*, *10*(1), 147–175. http://dx.doi.org/10.1007/BF01580665.

Mitsos, A., Chachuat, B., & Barton, P. I. (2009). Mccormick-based relaxations of algorithms. *SIAM Journal on Optimization*, *20*(2), 573–601. http://dx.doi.org/10.1137/080717341.

Moles, C. G., Mendes, P., & Banga, J. R. (2003). Parameter estimation in biochemical pathways: a comparison of global optimization methods. *Genome research*, *13*(11), 2467–2474. http://dx.doi.org/10.1101/gr.1262503.

Moré, J. J., Garbow, B. S., & Hillstrom, K. E. (1981). Testing unconstrained optimization software. *ACM Transactions on Mathematical Software*, *7*(1), 17–41. http://dx.doi.org/10.1145/355934.355936.

Murphy, K. P. (2022). *Adaptive computation and machine learning*, *Probabilistic machine learning: an introduction*. Cambridge, Massachusetts: The MIT Press.

Najman, J., Bongartz, D., & Mitsos, A. (2021). Linearization of McCormick relaxations and hybridization with the auxiliary variable method. *Journal of Global Optimization*, *80*(4), 731–756. http://dx.doi.org/10.1007/s10898-020-00977-x.

Nocedal, J. (1980). Updating quasi-Newton matrices with limited storage. *Mathematics of Computation*, *35*(151), 773–782. http://dx.doi.org/10.2307/2006193.

Osborne, M. R. (1972). Some aspects of nonlinear least squares calculations. In F. A. Lootsma (Ed.), *Numerical methods for nonlinear optimization* (pp. 171–189). New York: Academic Press.

Pan, X., Geng, N., & Xie, X. (2021). Appointment scheduling and real-time sequencing strategies for patient unpunctuality. *European Journal of Operational Research*, *295*(1), 246–260. http://dx.doi.org/10.1016/j.ejor.2021.02.055.

Papamichail, I., & Adjiman, C. (2002). A rigorous global optimization algorithm for problems with ordinary differential equations. *Journal of Global Optimization*, *24*, 1–33. http://dx.doi.org/10.1023/A:1016259507911.

Rulliere, D., Faleh, A., Planchet, F., & Youssef, W. (2013). Exploring or reducing noise? A global optimization algorithm in the presence of noise. *Structural and Multidisciplinary Optimization*, *47*(6), 921–936. http://dx.doi.org/10.1007/s00158-012-0874-5.

Sass, S., Tsoukalas, A., Bell, I. H., Bongartz, D., Najman, J., & Mitsos, A. (2023). Towards global parameter estimation exploiting reduced data sets. *Optimization Methods and Software*, 1–13. http://dx.doi.org/10.1080/10556788.2023.2205645.

Shcherbina, O., Neumaier, A., Sam-Haroud, D., Vu, X.-H., & Nguyen, T.-V. (2003). Benchmarking global optimization and constraint satisfaction codes. In C. Bliek, C. Jermann, & A. Neumaier (Eds.), *Global optimization and constraint satisfaction* (pp. 211–222). Berlin, Heidelberg: Springer Berlin Heidelberg, http://dx.doi.org/10.1007/978-3-540-39901-8_16.

Singer, A. B., & Barton, P. I. (2006). Global optimization with nonlinear ordinary differential equations. *Journal of Global Optimization*, *34*(2), 159–190. http://dx.doi.org/10.1007/s10898-005-7074-4.

Smith, E. M., & Pantelides, C. C. (1997). Global optimisation of nonconvex MINLPs. *Computers & Chemical Engineering*, *21*, 791–796. http://dx.doi.org/10.1016/S0098-1354(97)87599-0.

Tawarmalani, M., & Sahinidis, N. V. (2002). *Convexification and global optimization in continuous and mixed-integer nonlinear programming: theory, algorithms, software, and applications, vol. 65*. Springer US, http://dx.doi.org/10.1007/978-1-4757-3532-1.

Tawarmalani, M., & Sahinidis, N. V. (2004). Global optimization of mixed-integer nonlinear programs: A theoretical and computational study. *Mathematical Programming*, *99*(3), 563–591. http://dx.doi.org/10.1007/s10107-003-0467-6.

Tawarmalani, M., & Sahinidis, N. V. (2005). A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, *103*(2), 225–249. http://dx.doi.org/10.1007/s10107-005-0581-8.

Tsoukalas, A., & Mitsos, A. (2014). Multivariate McCormick relaxations. *Journal of Global Optimization*, *59*(2), 633–662. http://dx.doi.org/10.1007/s10898-014-0176-0.

Vanderbei, R., & colleagues (2004). Nonlinear Optimization Models. URL https://vanderbei.princeton.edu/ampl/nlmodels/, (Accessed 12 Jan 2023).

Villaverde, A. F., Fröhlich, F., Weindl, D., Hasenauer, J., & Banga, J. R. (2019). Benchmarking optimization methods for parameter estimation in large kinetic models. *Bioinformatics (Oxford, England)*, *35*(5), 830–838. http://dx.doi.org/10.1093/bioinformatics/bty736.

Wächter, A., & Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, *106*(1), 25–57. http://dx.doi.org/10.1007/s10107-004-0559-y.