

DFG Deutsche
Forschungsgemeinschaft
Priority Programme 1962

ROM-Based Inexact Subdivision Methods for PDE-Constrained Multiobjective Optimization

Stefan Banholzer, Bennet Gebken, Lena Reichle, Stefan Volkwein



Preprint Number SPP1962-166

received on April 7, 2021

Edited by
SPP1962 at Weierstrass Institute for Applied Analysis and Stochastics (WIAS)
Leibniz Institute in the Forschungsverbund Berlin e.V.
Mohrenstraße 39, 10117 Berlin, Germany
E-Mail: spp1962@wias-berlin.de

World Wide Web: <http://spp1962.wias-berlin.de/>

ROM-based inexact subdivision methods for PDE-constrained multiobjective optimization

Stefan Banholzer ^{1,‡}, Bennet Gebken ^{2,‡}, Lena Reichle ^{1,‡}, and Stefan Volkwein ^{1,‡,*}

¹ Department of Mathematics and Statistics, University of Konstanz

² Faculty for Computer Science, Electrical Engineering and Mathematics, Paderborn University

* Correspondence: Stefan.Volkwein@uni-konstanz.de

‡ These authors contributed equally to this work.

Abstract: The goal in multiobjective optimization is to determine the so-called Pareto set. Our optimization problem is governed by a parameter dependent semilinear elliptic partial differential equation (PDE). To solve it, we use a gradient based set-oriented numerical method. The numerical solution of the PDE by standard discretization methods usually leads to high computational effort. To overcome this difficulty, reduced-order modeling (ROM) is developed utilizing the reduced basis method. These model simplifications cause inexactness in the gradients. For that reason, an additional descent condition is proposed. Applying a modified subdivision algorithm, numerical experiments illustrate the efficiency of our solution approach.

Keywords: Multiobjective optimization; PDE-constrained optimization; reduced-order modelling; set-oriented methods; inexact optimization

1. Introduction

Multiobjective optimization plays an important role in many applications, e.g. in industry, medicine or engineering. One of the mentioned examples is the minimization of costs with simultaneous quality optimization in production or the minimization of CO₂ emission in energy generation and simultaneous cost minimization. These problems lead to multiobjective optimization problems (MOPs), where we want to achieve an optimal compromise with respect to all given objectives at the same time. Normally, the different objectives are contradictory such that there exists an infinite number of optimal compromises. The set of these compromises is called the *Pareto set*. The goal is to approximate the Pareto set in an efficient way, which turns out to be more expensive than solving a single objective optimization problem.

Since multiobjective optimization problems are of great importance, there exist several algorithms to solve them. Among the most popular methods are scalarization methods, which transform MOPs into scalar problems. For example, in the weighted sum method [12,22,25,37], convex combinations of the original objectives are optimized. Another popular approach is to use non-deterministic methods like evolutionary algorithms; cf. e.g. [11]. Furthermore, as multiobjective problems are generalizations of scalar problems, some solution methods can be generalized from the scalar to the multiobjective case [14,15,31].

In addition to the classical methods above, there are set-based strategies for the solution of MOPs. Continuation methods [1,20,33] use the fact that the Pareto set is typically (the projection of) a smooth manifold. Subdivision methods [2,9,23,34] use tools from the area of dynamical system to generate a covering of the Pareto set via hypercubes. However, especially when the objective functions and their gradients are expensive to evaluate – e.g., since an underlying PDE has to be solved for every evaluation – the computational time of these methods can quickly become very large. In the presence of PDE constraints, surrogate models offer a promising tool to reduce the computational effort significantly [32]. Examples are dimensional reduction techniques such as the Reduced Basis (RB) Method [19,26]. In an offline phase, a low-dimensional surrogate model of the PDE is constructed by using, e.g., the greedy algorithm, cf. [19]. In the online phase, only the RB model is used to solve the PDE, which saves a lot of computing time.

Citation: Banholzer, S.; Gebken, B.; Reichle, L.; Volkwein, S. ROM-based inexact subdivision methods for PDE-constrained MOP. *Math. Comput. Appl.* **2021**, *1*, 0. <https://doi.org/>

Received:

Accepted:

Published:

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Submitted to *Math. Comput. Appl.* for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

38 In this article, we combine an extension of the set-oriented method presented in [2] based
 39 on inexact gradient evaluations of the objective functions with an RB approach and a discrete
 40 empirical interpolation method (DEIM) [5,6] for semilinear elliptic PDEs. In order to deal with the
 41 inexactness introduced by the surrogate model, we combine the first-order optimality conditions
 42 for multiobjective optimization problems with error estimates for the RB-DEIM method and
 43 derive an additional condition for the descent direction [1] to get a tight superset of the Pareto set.
 44 This approach allows us to better control the quality of the result by controlling the errors for the
 45 objective functions independently. In order to obtain an even tighter superset of the Pareto set, we
 46 update these error estimates in our subdivision algorithm after each iteration step.

47 The article is organized as follows: In Section 2, we recall the basic concepts of multiobjective
 48 optimization problems and review results on descent directions with exact and inexact gradients.
 49 Furthermore, we develop a set oriented method to solve these problems, where only inexact
 50 gradient information is utilized. In Section 3, the PDE-constrained multiobjective optimization
 51 problem and the underlying semilinear PDE are introduced. Subsequently, we show how reduced-
 52 order modelling can be applied efficiently. In Section 4, numerical results concerning both the
 53 subdivision and the modified algorithm are presented. Finally, we give a conclusion and discuss
 54 possible future work in Section 5.

55 2. A set oriented method for multiobjective optimization with inexact objective gradients

56 In this section, we briefly recall the basic concepts of multiobjective optimization. Further-
 57 more, we develop a set oriented method to solve these problems, where only inexact gradient
 58 information is utilized.

59 2.1. Multiobjective optimization

Let $\mu_a, \mu_b \in \mathbb{R}^m$ with $\mu_a \leq \mu_b$ be arbitrary. We define the convex and compact parameter set
 $\mathcal{P}_{\text{ad}} = [\mu_a, \mu_b] \subset \mathbb{R}^m$. Now the goal is to solve the constrained multiobjective optimization
 problem

$$\min \hat{J}(\mu) \quad \text{subject to (s.t.)} \quad \mu \in \mathcal{P}_{\text{ad}} \quad (1)$$

60 with a given objective $\hat{J} = (\hat{J}_1, \dots, \hat{J}_k) : \mathcal{P}_{\text{ad}} \rightarrow \mathbb{R}^k$ and $k > 1$.

61 Compared to scalar optimization, we do not have a natural total order of \mathbb{R}^k for $k > 1$.
 62 Therefore, we cannot expect that there is a single point in \mathcal{P}_{ad} that minimizes all objectives \hat{J}_i
 63 simultaneously. For this reason, we make use of the following definition.

64 **Definition 1.** a) A point $\bar{\mu} \in \mathcal{P}_{\text{ad}}$ is called (globally) Pareto optimal, if there is no $\mu \in \mathcal{P}_{\text{ad}}$
 65 satisfying $\hat{J}(\mu) \preceq \hat{J}(\bar{\mu})$. In that case we call $\bar{\mu}$ a Pareto point.

b) The set of all Pareto points in \mathcal{P}_{ad} is called Pareto set and is denoted by

$$\mathbb{P} = \{\mu \in \mathcal{P}_{\text{ad}} \mid \mu \text{ is Pareto optimal}\}.$$

66 c) The image $\hat{J}(\mathbb{P}) \subset \mathbb{R}^k$ of the Pareto set under \hat{J} is called the Pareto front.

If \hat{J} is continuously differentiable on an open set containing \mathcal{P}_{ad} , then there exists a first-order
 necessary condition for Pareto optimality. To formulate this condition, we define the convex,
 closed and bounded set

$$\Delta_k = \left\{ \alpha \in \mathbb{R}^k \mid \alpha_i \geq 0 \text{ and } \sum_{i=1}^k \alpha_i = 1 \right\}.$$

Further, the row vector

$$\nabla \hat{J}_i(\mu) = \left(\frac{\partial \hat{J}_i}{\partial \mu_j}(\mu) \right)_{1 \leq j \leq m} \in \mathbb{R}^{1 \times m}$$

67 stands for the gradient of the i -th objective \hat{J}_i with $i \in \{1, \dots, k\}$.

Definition 2. If for a given $\bar{\mu} \in \mathcal{P}_{\text{ad}}$ there exists an $\bar{\alpha} = (\bar{\alpha}_i)_{1 \leq i \leq k} \in \Delta_k$ with

$$\sum_{i=1}^k \bar{\alpha}_i \nabla \hat{J}_i(\bar{\mu})(\mu - \bar{\mu}) = (\mu - \bar{\mu})^\top D\hat{J}(\bar{\mu})^\top \bar{\alpha} \geq 0 \quad \text{for all } \mu \in \mathcal{P}_{\text{ad}} \quad (2)$$

then we call $\bar{\mu}$ Pareto critical, where

$$D\hat{J}(\mu) = \begin{pmatrix} \nabla \hat{J}_1(\bar{\mu}) \\ \vdots \\ \nabla \hat{J}_k(\mu) \end{pmatrix} \in \mathbb{R}^{k \times m}$$

68 denotes the Jacobi matrix of \hat{J} at μ . The set of all Pareto critical points is called the Pareto critical
69 set, denoted by \mathbb{P}_c .

70 Now we recall the first-order necessary optimality conditions for (1).

71 **Theorem 1.** Let \hat{J} be continuously differentiable and $\bar{\mu} \in \mathcal{P}_{\text{ad}}$ Pareto optimal. Then $\bar{\mu}$ is Pareto-
72 critical, i.e., it holds $\mathbb{P} \subset \mathbb{P}_c$. Condition (2) is called the Karush-Kuhn-Tucker (KKT) condition
73 for multiobjective optimization problems.

74 **Proof.** The claim follows from [12, Theorem 3.25] and the specific choice of \mathcal{P}_{ad} . \square

Remark 1. a) Let $\bar{\mu}$ belong to the interior of \mathcal{P}_{ad} , i.e., $\bar{\mu} \in \text{int}(\mathcal{P}_{\text{ad}}) = (\mu_a, \mu_b)$. Then (2) is
equivalent to

$$\sum_{i=1}^k \bar{\alpha}_i \nabla \hat{J}_i(\bar{\mu}) = \alpha^\top D\hat{J}(\mu) = 0 \quad \text{in } \mathbb{R}^{1 \times m} \quad (3)$$

or

$$D\hat{J}(\mu)^\top \alpha = 0 \quad \text{in } \mathbb{R}^m = \mathbb{R}^{m \times 1};$$

75 see also [16].

76 b) Throughout the paper we only calculate the Pareto critical points in the interior of \mathcal{P}_{ad} and
77 make use of (3). The idea is to choose \mathcal{P}_{ad} sufficiently large so that we get $\mathbb{P}_c \subset \text{int}(\mathcal{P}_{\text{ad}})$.
c) Due to Theorem 1 we have

$$\mathbb{P} \subset \mathbb{P}_c = \left\{ \mu \in \mathcal{P}_{\text{ad}} \mid \exists \alpha = \alpha(\mu) \in \Delta_k : D\hat{J}(\bar{\mu})^\top \alpha = 0 \text{ in } \mathbb{R}^m \right\} \subset \text{int}(\mathcal{P}_{\text{ad}}).$$

78 provided $\mathbb{P}_c \subset \text{int}(\mathcal{P}_{\text{ad}})$ holds true. \diamond

79 2.2. Descent direction with exact gradients

80 Next we introduce the notion of a descent direction for the vector valued objective function \hat{J} at a
81 non-Pareto critical point $\mu \notin \mathbb{P}_c$. From now on we assume that $\hat{J} : \mathcal{P}_{\text{ad}} \rightarrow \mathbb{R}^k$ is continuously
82 differentiable (on an open set containing \mathcal{P}_{ad}).

Definition 3. The vector $v \in \mathbb{R}^m$ is a descent direction for \hat{J} in $\mu \in \mathcal{P}_{\text{ad}}$, if we have

$$\nabla \hat{J}_i(\mu)v \leq 0 \quad \text{for all } i \in \{1, \dots, k\}$$

83 and if there is at least one $i \in \{1, \dots, k\}$ with $\nabla \hat{J}_i(\mu)v < 0$.

84 One way to compute a descent direction is to solve a constrained quadratic optimization
85 problem in \mathbb{R}^k as shown in the following theorem. For a proof we refer to [31]. A similar result
86 was shown in [17].

Theorem 2. For given $\mu \in \text{int}(\mathcal{P}_{\text{ad}})$ let $\bar{\alpha} = \bar{\alpha}(\mu) \in \Delta_k$ be a (global) solution of the convex
constrained quadratic minimization problem

$$\min \frac{1}{2} \|D\hat{J}(\mu)^\top \alpha\|_2^2 \quad \text{s.t.} \quad \alpha \in \Delta_k. \quad (4)$$

87 Then we have either $D\hat{J}(\mu)^\top \bar{\alpha} = 0$ or $v = -D\hat{J}(\mu)^\top \bar{\alpha} \in \mathbb{R}^m$ is a descent direction for \hat{J} in μ .

88 Combined with a backtracking Armijo line search, the descent direction from Theorem 2
89 can be used to construct the steepest descent method in Algorithm 1.

Algorithm 1: Steepest descent method.

Require : $\kappa \in (0, 1)$, $\mu^0 \in \text{int}(\mathcal{P}_{\text{ad}})$ and $l = 0$;
1 Calculate α^0 as a solution of (4) for $\mu = \mu^0$; set $v^0 = -D\hat{J}(\mu^0)^\top \alpha^0$;
2 **while** $v^l \neq 0$ **do**
3 Choose the stepsize $t_l > 0$ as maximum of the set

$$\mathcal{T}_l = \left\{ t = 2^{-j} \mid j \in \mathbb{N}, \mu^l + tv^l \in \text{int}(\mathcal{P}_{\text{ad}}) \right.$$

$$\left. \text{and } \hat{J}(\mu^l + tv^l) \leq \hat{J}(\mu^l) + \kappa t D\hat{J}(\mu^l)v^l \right\};$$

Set $\mu^{l+1} = a(\mu^l)$ with $a(\mu^l) = \mu^l + t_l v^l$ and update $l = l + 1$;
4 Calculate α^l as a solution of (4) for $\mu = \mu^l$;
5 Set $v^l = -D\hat{J}(\mu^l)^\top \alpha^l$;
6 **end**

90 **Remark 2.** a) If Algorithm 1 terminates after a finite number l of iterations, then μ^l is a Pareto
91 critical point.
92 b) Assume that Algorithm 1 does not stop after a finite number of iterations. Then, every
93 accumulation point $\bar{\mu}$ of the sequence $\{\mu^l\}_{l \in \mathbb{N}}$ generated by Algorithm 1 is a Pareto critical
94 point. A proof – based on [15, Theorem 1] – can be found in [28, Theorem 5.2.5]. \diamond

95 **2.3. Descent direction with inexact gradients**

Suppose that we have continuously differentiable approximations $\hat{J}^\ell = (\hat{J}_i^\ell)_{1 \leq i \leq k}$ of the objective function \hat{J} satisfying

$$\max_{\mu \in \mathcal{P}_{\text{ad}}} \|\nabla \hat{J}_i(\mu) - \nabla \hat{J}_i^\ell(\mu)\|_2 \leq \varepsilon_i \quad \text{for } i \in \{1, \dots, k\}, \quad (5)$$

96 for given tolerances $\varepsilon_i \geq 0$. By $\mathbb{P} \subset \mathcal{P}_{\text{ad}}$ we denote the Pareto set for \hat{J} and by $\mathbb{P}^\ell \subset \mathcal{P}_{\text{ad}}$ the
97 Pareto set for \hat{J}^ℓ . If we write \mathbb{P}_c we mean the Pareto critical set for \hat{J} and \mathbb{P}_c^ℓ is the Pareto critical
98 set for \hat{J}^ℓ . Note that in general, we have neither $\mathbb{P}^\ell \subset \mathbb{P}$ nor $\mathbb{P} \subset \mathbb{P}^\ell$.

99 In this section, our goal is to compute an approximation of \mathbb{P} based on the approximation
100 \hat{J}^ℓ of the objective function and the error bounds ε_i . We begin by investigating the relationship
101 between the KKT conditions of the original objective function and its approximation.

Lemma 1. Let (5) be satisfied and $\bar{\mu} \in \text{int}(\mathcal{P}_{\text{ad}})$ be Pareto critical for \hat{J} with the KKT-condition vector $\bar{\alpha} \in \Delta_k$. Then it holds

$$\|D\hat{J}^\ell(\bar{\mu})^\top \bar{\alpha}\|_2 \leq \sum_{i=1}^k \bar{\alpha}_i \varepsilon_i = \langle \bar{\alpha}, \varepsilon \rangle_2 \leq \|\varepsilon\|_\infty \quad (6)$$

102 with $\varepsilon = (\varepsilon_i)_{1 \leq i \leq k}$, where we set $\langle \bar{\alpha}, \varepsilon \rangle_2 = \bar{\alpha}^\top \varepsilon$.

Proof. From $\bar{\mu} \in \text{int}(\mathcal{P}_{\text{ad}})$ and Remark 1-a) we infer that $D\hat{J}(\bar{\mu})^\top \bar{\alpha} = 0$ holds. Hence,

$$\begin{aligned} \|D\hat{J}^\ell(\bar{\mu})^\top \bar{\alpha}\|_2 &= \|D\hat{J}^\ell(\bar{\mu})^\top \bar{\alpha} - D\hat{J}(\bar{\mu})^\top \bar{\alpha}\|_2 = \left\| \sum_{j=1}^k (\nabla \hat{J}_j^\ell(\bar{\mu}) - \nabla \hat{J}_j(\bar{\mu})) \bar{\alpha}_j \right\|_2 \\ &\leq \sum_{j=1}^k \|\nabla \hat{J}_j^\ell(\bar{\mu}) - \nabla \hat{J}_j(\bar{\mu})\|_2 \bar{\alpha}_j \leq \sum_{j=1}^k \varepsilon_j \bar{\alpha}_j = \langle \bar{\alpha}, \varepsilon \rangle_2 \leq \|\varepsilon\|_\infty \end{aligned}$$

103 which gives the desired results. \square

104 Based on estimate (6) we define two approximation sets for the Pareto critical set \mathbb{P}_c of \hat{J} .

Definition 4. Let us introduce the two sets

$$\mathbb{P}_1^\ell = \left\{ \mu \in \text{int}(\mathcal{P}_{\text{ad}}) \mid \min_{\alpha \in \Delta_k} \|D\hat{J}^\ell(\mu)^\top \alpha\|_2^2 \leq \|\varepsilon\|_\infty^2 \right\} \subset \mathcal{P}_{\text{ad}}$$

and

$$\mathbb{P}_2^\ell = \left\{ \mu \in \text{int}(\mathcal{P}_{\text{ad}}) \mid \min_{\alpha \in \Delta_k} \left(\|D\hat{J}^\ell(\mu)^\top \alpha\|_2^2 - \langle \alpha, \varepsilon \rangle_2^2 \right) \leq 0 \right\} \subset \mathcal{P}_{\text{ad}}.$$

Lemma 2. It holds

$$\mathbb{P}_c \subset \mathbb{P}_2^\ell, \quad \mathbb{P}_c^\ell \subset \mathbb{P}_2^\ell \quad \text{and} \quad \mathbb{P}_2^\ell \subset \mathbb{P}_1^\ell.$$

Proof. Let $\bar{\mu} \in \mathbb{P}_c$ be a Pareto critical point of \hat{J} , then there exists $\bar{\alpha} \in \Delta_k$ with $D\hat{J}(\bar{\mu})^\top \bar{\alpha} = 0$. From Lemma 1 it follows that $\mathbb{P}_c \subset \mathbb{P}_2^\ell$.

Next we assume that $\bar{\mu} \in \mathbb{P}_c^\ell$ is a Pareto critical point of \hat{J}^ℓ , then there exists $\bar{\alpha} \in \Delta_k$ with $D\hat{J}^\ell(\bar{\mu})^\top \bar{\alpha} = 0$. This implies

$$\min_{\alpha \in \Delta_k} \left(\|D\hat{J}^\ell(\bar{\mu})^\top \alpha\|_2^2 - \langle \alpha, \varepsilon \rangle_2^2 \right) \leq \|D\hat{J}^\ell(\bar{\mu})^\top \bar{\alpha}\|_2^2 - \langle \bar{\alpha}, \varepsilon \rangle_2^2 = -\langle \bar{\alpha}, \varepsilon \rangle_2^2 \leq 0.$$

Hence, we have $\bar{\mu} \in \mathbb{P}_2^\ell$.

Let $\bar{\mu} \in \mathbb{P}_2^\ell$. Then, there exists $\bar{\alpha} \in \Delta_k$ with

$$\|D\hat{J}^\ell(\bar{\mu})^\top \bar{\alpha}\|_2^2 - \langle \bar{\alpha}, \varepsilon \rangle_2^2 \leq 0.$$

105 Thus, we get $\|D\hat{J}^\ell(\bar{\mu})^\top \bar{\alpha}\|_2^2 \leq \langle \bar{\alpha}, \varepsilon \rangle_2^2 \leq \|\varepsilon\|_\infty^2$ which implies $\mathbb{P}_2^\ell \subset \mathbb{P}_1^\ell$. \square

106 Our goal is to compute the set \mathbb{P}_2^ℓ via a descent method like Algorithm 1. To this end, the
107 following theorem presents a modified version of the descent direction (4), which additionally
108 takes the error bounds ε_i into account.

Theorem 3. Let $\varepsilon = (\varepsilon_j)_{1 \leq j \leq k}$ with $\varepsilon \geq 0$ and $\mu \in \text{int}(\mathcal{P}_{\text{ad}})$ be given. Assume that α_ε is a minimizer of the quadratic problem

$$\min_{\alpha \in \Delta_k} \left(\|D\hat{J}^\ell(\mu)^\top \alpha\|_2^2 - \langle \alpha, \varepsilon \rangle_2^2 \right). \quad (7)$$

109 Then we have $\mu \in \mathbb{P}_2^\ell$ or $v_\varepsilon = -D\hat{J}^\ell(\mu)^\top \alpha_\varepsilon \in \mathbb{R}^m$ is a descent direction for \hat{J}^ℓ in μ .

Proof. The Lagrange functions for (7) is given as

$$L : \mathbb{R}^k \times \mathbb{R} \times \mathbb{R}^k \rightarrow \mathbb{R}, \quad (\alpha, \lambda, \varrho) \mapsto \|D\hat{J}^\ell(\mu)^\top \alpha\|_2^2 - \langle \alpha, \varepsilon \rangle_2^2 + \lambda \left(1 - \sum_{j=1}^k \alpha_j \right) - \sum_{j=1}^k \varrho_j \alpha_j.$$

Since α_ε is a minimizer of (7), we get Lagrangian multipliers $\lambda \in \mathbb{R}$ and $\varrho \in \mathbb{R}_{\geq 0}^k$ with:

$$\begin{aligned} 2D\hat{J}^\ell(\mu)D\hat{J}^\ell(\mu)^\top \alpha_\varepsilon - 2\varepsilon \langle \varepsilon, \alpha_\varepsilon \rangle_2 + \lambda(-1, \dots, -1)^\top - \varrho &= 0, \\ \varrho_i \geq 0 \quad \text{and} \quad (\alpha_\varepsilon)_i \varrho_i &= 0. \end{aligned} \quad (8)$$

If we multiply (8) with α_ε^\top from the left, we get

$$2\alpha_\varepsilon^\top D\hat{J}^\ell(\mu)D\hat{J}^\ell(\mu)^\top \alpha_\varepsilon - 2\langle \varepsilon, \alpha_\varepsilon \rangle_2^2 - \lambda \sum_{i=1}^m ((\alpha_\varepsilon)_i - (\alpha_\varepsilon)_i \varrho_i) = 0$$

which implies

$$\lambda = 2 \|D\hat{J}^\ell(\mu)^\top \alpha_\varepsilon\|_2^2 - \langle \alpha_\varepsilon, \varepsilon \rangle_2^2.$$

First case: $\lambda \leq 0$, then $\mu \in \mathbb{P}_2^\ell$ holds and we are done.

Second case: $\lambda > 0$, then $\mu \notin \mathbb{P}_2^\ell$ holds. In this case we show that $v = -D\hat{J}^\ell(\mu)^\top \alpha_\varepsilon$ is a descent direction in μ for every objective function \hat{J}_j^ℓ with $j = 1, \dots, k$:

Define

$$K(\mu) = \left\{ D\hat{J}^\ell(\mu)^\top \alpha \mid \alpha \in \mathbb{R}^k, \alpha_i \geq 0 \text{ and } \sum_{i=1}^k \alpha_i = 1 \right\}.$$

If we can show that $w^\top v < 0$ holds for every $w \in K$, we know that v is a descent direction for every objective function J_i^ℓ in μ .

Choose $w \in K(\mu)$. Then there exists an $\alpha_w \in \Delta_k$ with $w = D\hat{J}^\ell(\mu)^\top \alpha_w$ and using (8) we obtain

$$\begin{aligned} w^\top v &= (D\hat{J}^\ell(\mu)^\top \alpha_w)^\top v = -\alpha_w^\top D\hat{J}^\ell(\mu) D\hat{J}^\ell(\mu)^\top \alpha_\varepsilon \\ &= -\alpha_w^\top \left(\varepsilon \langle \varepsilon, \alpha_\varepsilon \rangle_2 + \frac{1}{2} \lambda (1, \dots, 1) + \frac{1}{2} \varrho \right) = -\left(\langle \alpha_w, \varepsilon \rangle_2 \langle \alpha_\varepsilon, \varepsilon \rangle_2 + \frac{1}{2} \lambda + \frac{1}{2} \langle \alpha_w, \varrho \rangle_2 \right) \\ &\leq -\frac{1}{2} (\lambda + \langle \alpha_w, \varrho \rangle_2) \leq -\frac{\lambda}{2} < 0. \end{aligned}$$

110 Hence, v is a descent direction in μ for every objective function \hat{J}_j^ℓ , $j = 1, \dots, k$. □

111 The descent direction v_ε from the previous theorem will be referred to as the modified
112 descent direction. Based on this direction, we can now construct a descent method for the
113 computation of \mathbb{P}_2^ℓ which is shown in Algorithm 2.

Algorithm 2: Descent method with inexact gradients.

Require : $\varepsilon = (\varepsilon_j)_{1 \leq j \leq k}$, $\kappa \in (0, 1)$, $\mu^0 \in \text{int}(\mathcal{P}_{\text{ad}})$ and $l = 0$;

1 **while** $\mu_l \notin \mathbb{P}_2^\ell$ **do**

2 Calculate α_ε^l as solution of (7) for $\mu = \mu^l$;

3 Set $v_\varepsilon^l = -D\hat{J}^\ell(\mu^l)^\top \alpha_\varepsilon^l$;

4 Choose the stepsize $t_l > 0$ as maximum of the set

$$\begin{aligned} \mathcal{T}_l &= \left\{ t = 2^{-j} \mid j \in \mathbb{N}, \mu^l + tv_\varepsilon^l \in \text{int}(\mathcal{P}_{\text{ad}}) \right. \\ &\quad \left. \text{and } \hat{J}^\ell(\mu^l + tv_\varepsilon^l) \leq \hat{J}^\ell(\mu^l) + \kappa t D\hat{J}^\ell(\mu^l) v_\varepsilon^l \right\}; \end{aligned}$$

5 Set $\mu^{l+1} = a_\varepsilon(\mu^l)$ with $a_\varepsilon(\mu^l) = \mu^l + t_l v_\varepsilon^l$ and update $l = l + 1$;

6 **end**

- 114 **Remark 3.** a) If Algorithm 2 terminates after l iteration steps, then μ^l is contained in \mathbb{P}_2^ℓ .
115 b) Assume that Algorithm 2 does not terminate after a finite number of iteration steps. Then,
116 every accumulation point $\bar{\mu}$ of the sequence $\{\mu^l\}_{l \in \mathbb{N}}$ generated by Algorithm 2 is in the set
117 \mathbb{P}_2^ℓ . A proof, which is based on [15, Theorem 1], can be found in [28, Theorem 5.3.5].
118 c) Note that the tolerance ε is constant for all l throughout Algorithm 2. In Section 2.5 we will
119 adapt ε in each iteration. ◇

120 2.4. Subdivision algorithm

121 As mentioned in the introduction, there are set-based solution methods for MOPs which globally
122 approximate the Pareto set via sets (instead of a finite number of points). Here, we will consider
123 the *subdivision algorithm* [2,9,34], which computes an approximation of the Pareto set as a
124 covering of hypercubes (or *boxes*). The idea is to start with a large box containing the Pareto

125 set which is then iteratively subdivided into smaller boxes, while eliminating boxes that do not
126 contain part of the Pareto set.

127 There are essentially two versions of the subdivision scheme: one is gradient free and thus,
128 is particularly useful in the case when the evaluation of gradients is computationally expensive.
129 We refer to [2], where this variant is utilized to numerically realize a reduced-order approach
130 for a PDE-constrained multiobjective optimization problem. The other one is directly based on
131 a dynamical systems approach and utilizes gradient information in a similar way to memetic
132 algorithms; see [31]. Here, we will generalize the latter to the case of inexact gradients.

For a stepsize $t_l > 0$ let us formulate a descent step of the optimization procedure by

$$\mu^{l+1} = a(\mu^l) = \mu^l + t_l v^l \quad \text{or} \quad \mu^{l+1} = a_\varepsilon(\mu^l) = \mu^l + t_l v_\varepsilon^l,$$

133 where v^l and v_ε^l are the descent directions given by Theorems 2 and 3, respectively, with the
134 choice $\mu = \mu^l$. Depending on the descent step that we use, we either want to compute the Pareto
135 critical set \mathbb{P}_c or the superset \mathbb{P}_2^ℓ or \mathbb{P}_1^ℓ of \mathbb{P}_c . Since these sets are the sets of fixed points for
136 their respective descent step, we want to find the subset $A_{\mathbb{P}} \subset \text{int}(\mathcal{P}_{\text{ad}})$ satisfying $a(A_{\mathbb{P}}) = A_{\mathbb{P}}$ or
137 $a_\varepsilon(A_{\mathbb{P}}) = A_{\mathbb{P}}$.

To generate the set $A_{\mathbb{P}}$, we will use a subdivision method. This method produces an outer
approximation of the set $A_{\mathbb{P}}$ in the form of a nested sequence of sets $\mathcal{B}_0, \mathcal{B}_1, \dots \subset \mathfrak{P}(\mathcal{P}_{\text{ad}})$, where
 $\mathfrak{P}(\mathcal{P}_{\text{ad}})$ denotes the power set of \mathcal{P}_{ad} and each \mathcal{B}_l is a subset of \mathcal{B}_{l-1} in the sense that

$$\bigcup_{B \in \mathcal{B}_l} B \subset \bigcup_{B \in \mathcal{B}_{l-1}} B$$

138 holds and \mathcal{B}_l consists of finitely many subsets B covering $A_{\mathbb{P}}$ for all $l \in \mathbb{N}$. For each set \mathcal{B}_l
139 we define a diameter through $\text{diam}(\mathcal{B}_l) = \max_{B \in \mathcal{B}_l} (\text{diam}(B))$. Algorithm 3 shows the classical
140 subdivision method (based on Theorem 2) and our modified descent direction (based on Theorem
141 3).

Algorithm 3: Subdivision algorithm.

Require : $\mathcal{B}_0 \subset \mathfrak{P}(\mathcal{P}_{\text{ad}})$ finite collection of subsets of \mathcal{P}_{ad} with $\bigcup_{B \in \mathcal{B}_0} B = \mathcal{P}_{\text{ad}}$,
 $\theta \in (0, 1)$, $l = 0$; in the setting of inexact gradients $\varepsilon_1, \dots, \varepsilon_k$;

1 **while** $a_{(\varepsilon)}(\bigcup_{B \in \mathcal{B}_l} B) \neq \bigcup_{B \in \mathcal{B}_l} B$ **do**

2 **Subdivision:**

3 Construct from \mathcal{B}_l a set $\hat{\mathcal{B}}_{l+1} \subset \mathfrak{P}(\mathcal{P}_{\text{ad}})$ with

$$\bigcup_{B \in \hat{\mathcal{B}}_{l+1}} B = \bigcup_{B \in \mathcal{B}_l} B \quad \text{and} \quad \text{diam}(\hat{\mathcal{B}}_{l+1}) = \theta \text{diam}(\mathcal{B}_l);$$

4 **Selection:**

5 Define the new set \mathcal{B}_{l+1} by

$$\mathcal{B}_{l+1} = \{B \in \hat{\mathcal{B}}_{l+1} \mid \exists \hat{B} \in \hat{\mathcal{B}}_{l+1} \text{ with } a_{(\varepsilon)}^{-1}(B) \cap \hat{B} \neq \emptyset\};$$

6 Set $l = l + 1$;

7 **end**

142 **Remark 4.** In order to realize the subdivision algorithm numerically we choose a similar way as
143 described in [9, Remark 2.4]. Instead of working explicitly with the centers and radii of the boxes,
144 these are stored within a binary tree in the subdivision step, whereby the memory requirement is
145 noticeably reduced. The selection step is implemented using a certain number of sample points in
146 each box. These sample points are chosen either on an a priori defined grid or randomly within
147 the boxes. Afterwards $a_{(\varepsilon)}$ is evaluated in these points. For more details we refer the reader to [8,
148 Section 5]. \diamond

149 **2.5. Modified subdivision algorithm for inexact gradients**

In Algorithm 2 we utilize the same error bounds $\varepsilon = (\varepsilon_1, \dots, \varepsilon_k)$ with $\varepsilon_i \geq 0, i = 1, \dots, k$, in each iteration step l . Note that the larger the ε , the greater the difference between \mathbb{P}_c and \mathbb{P}_2^ℓ or \mathbb{P}_1^ℓ . In the algorithm, we produce an outer approximation of the set $A_{\mathbb{P}}$ with a nested sequence of sets $\{\mathcal{B}_l\}_{l \in \mathbb{N}}$ by

$$\tilde{\mathcal{B}}_l = \bigcup_{B \in \mathcal{B}_l} B \subseteq \mathcal{P}_{\text{ad}}.$$

Since it holds $\tilde{\mathcal{B}}_l \subset \mathcal{P}_{\text{ad}}$, we have

$$\max \{ \|\nabla \hat{J}_i(\mu) - \nabla \hat{J}_i^\ell(\mu)\|_2 \mid \mu \in \tilde{\mathcal{B}}_l \} \leq \max \{ \|\nabla \hat{J}_i(\mu) - \nabla \hat{J}_i^\ell(\mu)\|_2 \mid \mu \in \mathcal{P}_{\text{ad}} \} \quad \text{for } 1 \leq i \leq k.$$

Now we modify Algorithm 3 by utilizing the descent directions introduced in Theorem 3 and update ε after every iteration step l to generate a better approximation of the set $A_{\mathbb{P}}$. For updating ε we use the formula

$$\varepsilon_i^l = \sup \{ \|\nabla \hat{J}_i(\mu) - \nabla \hat{J}_i^\ell(\mu)\|_2 \mid \mu \in \tilde{\mathcal{B}}_l \} \quad \text{for } 1 \leq i \leq k \quad (9)$$

and set $\varepsilon^l = (\varepsilon_i^l)_{1 \leq i \leq k}$. Due to the nested choice of the box coverings, we have $\varepsilon_i^{l+1} \leq \varepsilon_i^l$ for $i = 1, \dots, k$ and $l \in \mathbb{N}$. In iteration step l we generate the descent direction by computing

$$\alpha_\varepsilon^l \in \arg \min \left\{ \|D\hat{J}^\ell(\mu)^\top \alpha\|_2^2 - \langle \alpha, \varepsilon^l \rangle_2^2 \mid \alpha \in \Delta_k \right\}. \quad (10)$$

Then we set

$$v_\varepsilon^l = -D\hat{J}^\ell(\mu)^\top \alpha_\varepsilon^l \quad \text{and} \quad \mu^{l+1} = a_\varepsilon^l(\mu^l) = \mu^l + t_l v_\varepsilon^l.$$

150 Typically, the set \mathbb{P}_2^ℓ is far smaller than the admissible set \mathcal{P}_{ad} . Hence, we expect that the error
151 bounds in (9) become significantly smaller than the ones from (5). Therefore we expect that we
152 get better results with the modified function a_ε^l instead of a_ε .

153 **3. Multiobjective optimization of a semilinear elliptic PDE**

154 In this section we introduce a multiobjective parameter optimization problem governed by a
155 semilinear elliptic PDE. Further, we show how reduced-order modelling can be applied efficiently.

156 **3.1. Problem formulation**

Let $\Omega \subset \mathbb{R}^{\mathfrak{d}}, \mathfrak{d} \in \{1, 2, 3\}$, be a bounded domain with Lipschitz-continuous boundary $\Gamma = \partial\Omega$. Then we consider the problem

$$\min_{(y, \mu)} J(y, \mu) = \begin{pmatrix} J_1(y, \mu) \\ \vdots \\ J_{k-1}(y, \mu) \\ J_k(y, \mu) \end{pmatrix} = \frac{1}{2} \begin{pmatrix} \int_{\Omega} |y - y_1^{\text{d}}|^2 \, \text{d}\mathbf{x} \\ \vdots \\ \int_{\Omega} |y - y_{k-1}^{\text{d}}|^2 \, \text{d}\mathbf{x} \\ \sum_{j=1}^m |\mu_j - \mu_j^{\text{d}}|^2 \end{pmatrix} \quad (11a)$$

subject to the elliptic boundary value problem

$$-c\Delta y + by + dy^3 = f + \sum_{i=1}^m \mu_i \xi_i \text{ in } \Omega, \quad c \frac{\partial y}{\partial \mathbf{n}} + y = g \text{ on } \Gamma, \quad (11b)$$

157 where $y \in V = H^1(\Omega)$ is the state variable and $\mu \in \mathcal{P}_{\text{ad}} = [\mu_a, \mu_b]$ the parameter. We suppose
158 that $g \in L^r(\Gamma)$ with $r > \mathfrak{d} - 1$, $\xi_1, \dots, \xi_m, f \in H = L^2(\Omega)$, $\mu^{\text{d}} = (\mu_j^{\text{d}}) \in \mathbb{R}^m$ and $y_1^{\text{d}}, \dots, y_{k-1}^{\text{d}} \in H$.
159 Moreover, b, c and d are non-negative constants with $c > 0$.

Since Ω is a bounded connected open set with smooth boundary, it is known that V is a Hilbert space endowed with the inner product

$$\langle \varphi, \psi \rangle_V = \int_{\Omega} \nabla \varphi \cdot \nabla \psi \, dx + \int_{\Gamma} \varphi \psi \, ds \quad \text{for } \varphi, \psi \in V$$

160 and the induced norm $\|\varphi\|_V = \langle \varphi, \varphi \rangle_V^{1/2}$ for $\varphi \in V$; see [7, p. 133], for instance.

161 3.2. The parameter dependent semilinear elliptic PDE

In this subsection we study the state equation (11b). First, we define the non-linear operator $\mathcal{A} : V \rightarrow V'$ by

$$\langle \mathcal{A}(y), \varphi \rangle_{V',V} = \int_{\Omega} c \nabla y \cdot \nabla \varphi + (by + dy^3) \varphi \, dx + \int_{\Gamma} y \varphi \, ds \quad \text{for } y, \varphi \in V.$$

Recall that $d \in \{1, 2, 3\}$ implies $V \hookrightarrow L^6(\Omega)$; cf. [36, Section 7]. Therefore, the operator \mathcal{A} is well-defined. Moreover, for $\mu \in \mathcal{P}_{\text{ad}}$ the functional $b_{\mu} \in V'$ is given by

$$\langle b_{\mu}, \varphi \rangle_{V',V} = \int_{\Omega} \left(f + \sum_{i=1}^m \mu_i \xi_i \right) \varphi \, dx + \int_{\Gamma} g \varphi \, ds \quad \text{for } \varphi \in V.$$

162 Now we define a weak solution to the state equation (11b).

Definition 5. A weak solution of (11b) is a function $y \in V$ satisfying

$$\langle \mathcal{A}(y), \varphi \rangle_{V',V} = \langle b_{\mu}, \varphi \rangle_{V',V} \quad \text{for all } \varphi \in V. \quad (12)$$

163 The following result is proved, e.g., in [36, Section 4.2.3].

Proposition 1. For a fixed parameter $\mu \in \mathbb{R}^m$ there exists a unique solution $y \in V$ to (11b). This solution is even continuous on $\overline{\Omega}$ and for a constant c_{∞} it holds

$$\|y\|_V + \|y\|_{C(\overline{\Omega})} \leq c_{\infty} \left(\|g\|_{L^r(\Gamma)} + \left\| f + \sum_{i=1}^m \mu_i \xi_i \right\|_H \right).$$

Remark 5. We define the state space $\mathcal{Y} = V \cap C(\overline{\Omega})$ which is a Banach space endowed with the natural norm

$$\|\varphi\|_{\mathcal{Y}} = \|\varphi\|_V + \|\varphi\|_{C(\overline{\Omega})} \quad \text{for } \varphi \in \mathcal{Y}.$$

164 Motivated by Proposition 1, we define the parameter-to-state mapping $\mathcal{S} : \mathcal{P}_{\text{ad}} \rightarrow \mathcal{Y}$ as follows:

165 For a given parameter $\mu \in \mathcal{P}_{\text{ad}}$ the function $y = \mathcal{S}(\mu) \in \mathcal{Y}$ is the solution to (11b). It follows by
166 standard arguments that \mathcal{S} is continuously Fréchet-differentiable; see [28, Sections 2 and 4]. \diamond

167 3.3. Reduced formulation and adjoint approach

Utilizing the parameter-to-state mapping \mathcal{S} we define the reduced cost functional

$$\hat{J}(\mu) = J(\mathcal{S}(\mu), \mu) = \begin{pmatrix} \hat{J}_1(\mu) \\ \vdots \\ \hat{J}_k(\mu) \end{pmatrix} \quad \text{for } \mu \in \mathcal{P}_{\text{ad}}$$

with

$$\hat{J}_i(\mu) = \frac{1}{2} \int_{\Omega} |(\mathcal{S}(\mu))(x) - y_i^d(x)|^2 \, dx \quad \text{for } 1 \leq i \leq k-1$$

and $\hat{J}_k(\mu) = \sum_{j=1}^m |\mu_j - \mu_j^d|^2 / 2$. Now, the *reduced problem* is given as

$$\min \hat{J}(\mu) \quad \text{s.t.} \quad \mu \in \mathcal{P}_{\text{ad}}. \quad (13)$$

168 If $\bar{\mu} \in \mathcal{P}_{\text{ad}}$ is a locally optimal solution to (13), then the pair $(\mathcal{S}(\bar{\mu}), \bar{\mu})$ is a locally optimal solution
169 to (11). Conversely, if $(\mathcal{S}(\bar{\mu}), \bar{\mu}) \in \mathcal{Y} \times \mathcal{P}_{\text{ad}}$ solves (11) locally, the parameter $\bar{\mu}$ is a locally optimal
170 solution to (13).

To apply the subdivision algorithm, the reduced objective function \hat{J} has to be Fréchet-differentiable that follows immediately from the fact that the parameter-to-state operator \mathcal{S} is Fréchet-differentiable; cf. Remark 5. The gradient of \hat{J} can be expressed by introducing adjoint variables. For that purpose we define the operators $\mathcal{B}_i : V \times V \rightarrow V'$, $1 \leq i \leq k-1$, as

$$\langle \mathcal{B}_i(p, y), \varphi \rangle_{V', V} = \int_{\Omega} c \nabla p \cdot \nabla \varphi + (b + 3dy^2)p\varphi \, dx + \int_{\Gamma} p\varphi \, dx - \int_G (y - y_i^d)\varphi \, dx$$

171 for $p, y, \varphi \in V$. Next we define adjoint variables.

Definition 6. Let a parameter $\mu \in \mathcal{P}_{\text{ad}}$ be given and $y(\mu) = \mathcal{S}(\mu) \in \mathcal{Y}$. For every $i \in \{1, \dots, k-1\}$ we call the solution $p_i(\mu) \in V$ to

$$\langle \mathcal{B}_i(p_i(\mu), y(\mu)), \varphi \rangle_{V', V} = 0 \quad \text{for all } \varphi \in V \quad (14)$$

172 the adjoint variable associated with the objective \hat{J}_i .

Remark 6. a) Notice that (14) is the weak formulation of the elliptic problem

$$\begin{aligned} -c\Delta p_i(\mu) + (b + 3dy(\mu)^2)p_i(\mu) &= y(\mu) - y_i^d \quad \text{in } \Omega, \\ c \frac{\partial p_i}{\partial \mathbf{n}}(\mu) + p_i(\mu) &= 0 \quad \text{on } \Gamma \end{aligned} \quad (15)$$

173 for $i = 1, \dots, k-1$.

174 b) Applying the Lax-Milgram lemma (see, e.g., [36, Section 2]) one can show that (14) has a
175 unique solution $p_i(\mu) \in V$ for all $\mu \in \mathcal{P}_{\text{ad}}$ and $i = 1, \dots, k-1$. \diamond

Now we can express the gradient of the reduced cost functional as follows:

$$\nabla \hat{J}_i(\mu) = \begin{pmatrix} \int_{\Omega} \xi_1 p_i(\mu) \, dx \\ \vdots \\ \int_{\Omega} \xi_m p_i(\mu) \, dx \end{pmatrix} \quad \text{for } 1 \leq i \leq k-1, \quad \nabla \hat{J}_k(\mu) = \mu - \mu^d,$$

176 where $y(\mu) \in \mathcal{Y}$ and $p_i(\mu) \in V$, $i \in \{1, \dots, k-1\}$, solve (12) and (14), respectively.

177 3.4. Finite element (FE) Galerkin discretization

178 Let us briefly introduce a standard finite element (FE) method based on a triangulation of the
179 spatial domain Ω . Here we utilize piecewise linear FE ansatz functions $\varphi_1, \dots, \varphi_N \in V$ which
180 are linearly independent. We define the finite-dimensional subspace $V^N = \text{span} \{\varphi_1, \dots, \varphi_N\} \subset V$
181 supplied with the same topology as in V .

Next we replace (12) by a FE Galerkin scheme: For each $\mu \in \mathcal{P}_{\text{ad}}$ the FE solution $y^N(\mu) \in V^N$ solves

$$\langle \mathcal{A}(y^N(\mu)), \varphi_i \rangle_{V', V} = \langle b_\mu, \varphi_i \rangle_{V', V} \quad \text{for } i = 1, \dots, N. \quad (16)$$

It follows by the same arguments as for (12) that the FE problem (16) has a unique solution $y^N = y^N(\mu)$ for every $\mu \in \mathcal{P}_{\text{ad}}$. Hence, the parameter-to-state FE mapping $\mathcal{S}^N : \mathcal{P}_{\text{ad}} \rightarrow V^N$,

$\mu \mapsto \mathcal{S}^N(\mu) = y^N(\mu)$ is well-defined. For $y^N \in V^N$ there exist coefficients $y_i^N, i = 1, \dots, N$, satisfying

$$y^N = \sum_{i=1}^N y_i^N \varphi_i \quad \text{for } \mathbf{x} \in \Omega. \quad (17)$$

Inserting (17) into (16) we can express (16) as a non-linear algebraic system. For that purpose we introduce the $N \times N$ -matrices

$$\mathbf{K} = \left(\left(\int_{\Omega} \nabla \varphi_j \cdot \nabla \varphi_i \, d\mathbf{x} \right) \right), \quad \mathbf{M} = \left(\left(\int_{\Omega} \varphi_j \varphi_i \, d\mathbf{x} \right) \right), \quad \mathbf{Q} = \left(\left(\int_{\Gamma} \varphi_j \varphi_i \, d\mathbf{s} \right) \right),$$

the N -vectors

$$y^N = (y_i^N), \quad \mathbf{F}_{\mu} = \left(\int_{\Omega} \left(f + \sum_{j=1}^m \mu_j \xi_j \right) \varphi_i \, d\mathbf{x} \right), \quad \mathbf{G} = \left(\int_{\Gamma} g \varphi_i \, d\mathbf{s} \right),$$

and the nonlinearity function $\mathbf{H} : \mathbb{R}^N \rightarrow \mathbb{R}^N$ given as

$$\mathbf{H}(\mathbf{v}) = \left(\int_{\Omega} \left(\sum_{j=1}^N v_j \varphi_j \right)^3 \varphi_i \, d\mathbf{x} \right) \quad \text{for } \mathbf{v} = (v_1, \dots, v_N) \in \mathbb{R}^N.$$

Inserting (17) into (16) we end up with the following non-linear system

$$(c\mathbf{K} + b\mathbf{M} + \mathbf{Q})y^N + d\mathbf{H}(y^N) = \mathbf{F}_{\mu} + \mathbf{G} \in \mathbb{R}^N. \quad (18)$$

Remark 7. *The difficulty of (18) lies in the fact that we cannot assemble $\mathbf{H}(y^N)$ efficiently in terms of a matrix-vector multiplication. Therefore we use mass lumping to compute $\mathbf{H}(y^N)$ approximately. For an introduction into mass lumping see e.g., [35, Section 15] or [39, Section 17.2]. With that we can write (18) as a root finding problem of*

$$(c\mathbf{K} + b\mathbf{M} + \mathbf{Q})y^N + d\tilde{\mathbf{M}}(y^N)^3 = \mathbf{F}_{\mu} + \mathbf{G} \in \mathbb{R}^N \quad (19)$$

with $(y^N)^3 = ((y_i^N)^3)_{1 \leq i \leq N}$ and the lumped mass matrix $\tilde{\mathbf{M}}$ defined by

$$\tilde{M}_{kj} := \delta_{kj} \sum_{l=1}^N M_{kl}.$$

182 Further details can be found in [4,28]. Let us refer to [30], where mass lumping is utilized in
183 optimal control. \diamond

Now the FE objectives are given as

$$\hat{J}_i^N(\mu) = \frac{1}{2} \int_{\Omega} |\mathcal{S}^N(\mu) - y_i^q|^2 \, d\mathbf{x} \quad \text{for } i = 1, \dots, k-1, \quad \hat{J}_k^N(\mu) = \sum_{j=1}^m |\mu_j - \mu_j^q|^2.$$

184 3.5. Reduced-order modelling (ROM)

To generate the Pareto critical set of the MOP (11) we need to evaluate the reduced objectives \hat{J}_i , $i = 1, \dots, k$, and their gradients many times. Hence, the state and adjoint equations have to be solved numerically very often. Therefore, the use of ROM is a suitable option. In this paper we will use the Reduced Basis (RB) method. The main idea is to construct a low dimensional (i.e., $\ell \ll N$) subspace V^{ℓ} of the FE space V^N spanned by FE solutions of the state and adjoint equation for appropriately chosen parameters $\mu \in \mathcal{P}_{\text{ad}}$. Here this strategy is realized by greedy algorithms. We refer to [19,26,27] for a general explanation and to [28, Section 3.2] for our specific problem (11). This is an iterative procedure where in each iteration FE solutions of the state and adjoint equation at a specific parameter value are added to the basis. An essential ingredient of greedy

algorithms is the choice of an error indicator $\eta_\ell(\mu)$. Here we use the maximal true error between the FE and the ROM gradients, i.e., we set

$$\eta_\ell(\mu) := \max_{1 \leq i \leq k} \|\nabla \hat{J}_i^N(\mu) - \nabla \hat{J}_i^\ell(\mu)\|_2. \quad (20)$$

185 Our subdivision scheme is based on gradient information. To be able to generate \mathbb{P}_1^ℓ or \mathbb{P}_2^ℓ for
 186 the approximation of \mathbb{P} , we have to ensure that the approximated objective function \hat{J}^ℓ based on
 187 the ROM model satisfies the inequality in (5). The idea is to generate a new basis element in
 188 every step until the maximum error $\eta_\ell(\mu)$ on a discrete training set S_{train} , which approximates
 189 \mathcal{P}_{ad} good enough, is smaller than a tolerance ε_{tol} . For more details see Algorithm 4.

Algorithm 4: Greedy algorithm.

Require : Tolerance $\varepsilon_{tol} > 0$, error indicator $\eta_\ell(\mu)$, discrete training set S_{train} ;
Return : RB space V^ℓ and basis Ψ_ℓ ;
 1 Set $V_0^\ell := \{0\}$, $\Psi_0 := \emptyset$, $\ell := 0$, $M := 0$;
 2 **while** $\max_{\mu \in S_{train}} \eta_\ell(\mu) > \varepsilon_{tol}$ and $M \leq |S_{train}|$ **do**
 3 Set $\mu_{M+1} \in \operatorname{argmax} \{\eta_\ell(\mu) : \mu \in S_{train}\}$;
 4 $\tilde{\psi}_{M+1}^p := p_N(\mu_{M+1})$ and $\tilde{\psi}_{M+1}^y := y_N(\mu_{M+1})$;
 5 Perform Gram-Schmidt orthonormalization (GS) of $\tilde{\psi}_{M+1}^p$ against Ψ_M and get
 ψ_{M+1}^p ;
 6 Set $\Psi_{M+1} := \{\Psi_M, \psi_{M+1}^p\}$, $V_{M+1}^\ell := V_M^\ell \oplus \{\psi_{M+1}^p\}$;
 7 **if** $\tilde{\psi}_{M+1}^y$ is linearly independent of V_{M+1}^ℓ **then**
 8 Perform GS orthonormalization of $\tilde{\psi}_{M+1}^y$ against Ψ_{M+1} and get ψ_{M+1}^y ;
 9 Set $\Psi_{M+2} := \{\Psi_{M+1}, \psi_{M+1}^y\}$, $V_{M+2}^\ell := V_{M+1}^\ell \oplus \{\psi_{M+1}^y\}$ and $M := M + 2$;
 10 **else**
 11 Set $M := M + 1$;
 12 **end**
 13 **end**

Since V^ℓ is a subset of V , we endow V^ℓ with the V -topology. Due to $\psi_j \in V^N$ ($1 \leq j \leq \ell$) there exists a coefficient matrix $\Psi \in \mathbb{R}^{N \times \ell}$ such that

$$\psi_j(\mathbf{x}) = \sum_{i=1}^N \Psi_{ij} \varphi_i(\mathbf{x}) \quad \text{for } \mathbf{x} \in \Omega. \quad (21)$$

Now we replace (16) by an RB Galerkin scheme: For each $\mu \in \mathcal{P}_{ad}$ the RB solution $y^\ell(\mu) \in V^\ell$ solves

$$\langle \mathcal{A}(y^\ell(\mu)), \psi_i \rangle_{V',V} = \langle b_\mu, \psi_i \rangle_{V',V} \quad \text{for } i = 1, \dots, \ell. \quad (22)$$

190 We suppose that (22) has a unique solution $y^\ell = y^\ell(\mu)$ for every $\mu \in \mathcal{P}_{ad}$. Hence, the parameter-
 191 to-state RB mapping $\mathcal{S}^\ell : \mathcal{P}_{ad} \rightarrow V^\ell$, $\mu \mapsto \mathcal{S}^\ell(\mu) = y^\ell(\mu)$ is well-defined.

Inserting (21) and $y^\ell(\mu) = \sum_{i=1}^N y_i^\ell(\mu) \psi_i$ into (22) we derive the non-linear algebraic system (cf. (18))

$$(c\mathbf{K}^\ell + b\mathbf{M}^\ell + \mathbf{Q}^\ell) + d\Psi^\top \mathbf{H}(\Psi y^\ell) = \mathbf{F}_\mu^\ell + \mathbf{G}^\ell \in \mathbb{R}^\ell \quad (\ell \ll N) \quad (23)$$

192 with the $\ell \times \ell$ matrices $\mathbf{K}^\ell = \Psi^\top \mathbf{K} \Psi$, $\mathbf{M}^\ell = \Psi^\top \mathbf{M} \Psi$, $\mathbf{Q}^\ell = \Psi^\top \mathbf{Q} \Psi$, the ℓ -vectors $\mathbf{F}_\mu^\ell = \Psi^\top \mathbf{F}_\mu$ and
 193 $\mathbf{G}^\ell = \Psi^\top \mathbf{G}$.

Remark 8. As mentioned in Remark 7 we apply mass lumping to evaluate the non-linear function \mathbf{H} more efficiently. With that we can write (23) as a root finding problem of

$$(c\mathbf{K}^\ell + b\mathbf{M}^\ell + \mathbf{Q}^\ell) y^\ell + d\tilde{\mathbf{M}}^\ell (\Psi y^\ell)^3 = \mathbf{F}_\mu^\ell + \mathbf{G}^\ell \in \mathbb{R}^\ell \quad (\ell \ll N) \quad (24)$$

194 with $(\Psi y^\ell)^3 = ((\Psi y^\ell)_i^3)_{1 \leq i \leq N}$ and the $\ell \times N$ matrix $\tilde{\mathbf{M}}^\ell = \Psi^\top \tilde{\mathbf{M}}$. However, in the RB Galerkin
 195 scheme the evaluation of the nonlinearity is still as costly as in the FE case. Here, discrete

196 *empirical interpolation (DEIM) is applied; cf. [5,6]. We skip the detailed description here and*
 197 *refer the reader to [28, Section 3.2].* \diamond

198 3.6. Convergence Analysis

199 We prove the convergence of the RB solution with mass lumping to the weak solution of the state
 200 and adjoint equation.

201 **Remark 9.** *Since the FE space is a finite dimensional space, the error for the state and adjoint*
 202 *equation converges for increasing dimension of the RB space to zero. Therefore, the RB solution*
 203 *converges for increasing accuracy in the Greedy algorithm to the FE solution. We skip a detailed*
 204 *description of the proof here and refer the reader to [28, Section 3.2].* \diamond

Theorem 4. *Let $(S_j)_j \subset \mathcal{P}_{\text{ad}}$ with $S_j \subseteq S_{j+1}$ be a growing sequence of training sets and choose a
 monotone sequence $(\varepsilon_j)_j \subset \mathbb{R}_{>0}$ satisfying*

$$\varepsilon_j \rightarrow 0 \text{ for } j \rightarrow \infty \quad \text{and} \quad \varepsilon_j \geq \varepsilon_{j+1}.$$

Then we get

$$\limsup_{j \rightarrow \infty} \{ \|y(\mu) - y^\ell(\mu)\|_V \mid \mu \in \mathcal{P}_{\text{ad}} \} = 0, \quad \limsup_{j \rightarrow \infty} \{ \|y(\mu) - y^\ell(\mu)\|_V \mid \mu \in \mathcal{P}_{\text{ad}} \} = 0.$$

Proof. Let $\mu \in \mathcal{P}_{\text{ad}}$ be an arbitrary parameter. It holds

$$\|y(\mu) - y^\ell(\mu)\|_V \leq \|y(\mu) - y^N(\mu)\|_V + \|y^N(\mu) - y^\ell(\mu)\|_V$$

205 with $N > \ell$. From Theorem A1 (see Appendix A) we infer that the first summand converges to
 206 zero for increasing N . For the second summand we refer the reader to [28, Section 3.2.5], in this
 207 section we proved the convergence of the RB solution to the FE solution. For the adjoint equation
 208 we can do the same and the claim follows. \square

209 4. Numerical experiments

210 In this section we use our algorithm to solve multiobjective optimization problems with PDE
 211 constraints and interpret the numerical results. All computations were executed on a computer
 212 with a 2,9 GHz Intel Core i7 CPU, 8 GB of RAM and an Intel HD Graphic 4000 1536 MB
 213 GPU. The algorithms were implemented in Matlab R2017b. For the subdivision method, we
 214 used the implementation from [https://math.uni-paderborn.de/en/ag/chair-of-applied-mathematics/
 215 research/software](https://math.uni-paderborn.de/en/ag/chair-of-applied-mathematics/research/software).

In this example we will numerically investigate the application of the modified subdivision
 algorithm presented in Section 2.5 to the PDE-constrained multiobjective optimization problem
 using the RB-DEIM solver from Section 3.5. For the underlying PDE we set $\mathfrak{d} = 2$, $\Omega = (0, 1)^2$
 with elements $\mathbf{x} = (x_1, x_2)$, $\mathcal{P}_{\text{ad}} = [-2, 2]^2$, $b = c = d = 1$, the right-hand side $f(\mathbf{x}) =$
 $x_1^2 + x_2^2 - 4 + (x_1^2 + x_2^2)^3$, $m = 2$, $\xi_1(\mathbf{x}) = -25 \cdot \mathbb{1}_{x_1 > 0.5}(\mathbf{x})$, $\xi_2(\mathbf{x}) = 25 \cdot \mathbb{1}_{x_1 \leq 0.5}(\mathbf{x})$ and the
 boundary condition $g(\mathbf{x}) = 2 \cdot \mathbb{1}_{x_1=1}(\mathbf{x}) + 2 \cdot \mathbb{1}_{x_2=1}(\mathbf{x}) + x_1^2 + x_2^2$. This leads to the following PDE

$$-\Delta y + y + y^3 = f + \mu_1 \xi_1 + \mu_2 \xi_2 \text{ in } \Omega, \quad \frac{\partial y}{\partial \mathbf{n}} + y = g \text{ on } \partial\Omega. \quad (25)$$

216 In Figure 1 corresponding solutions of the state equation are shown for three values of μ .

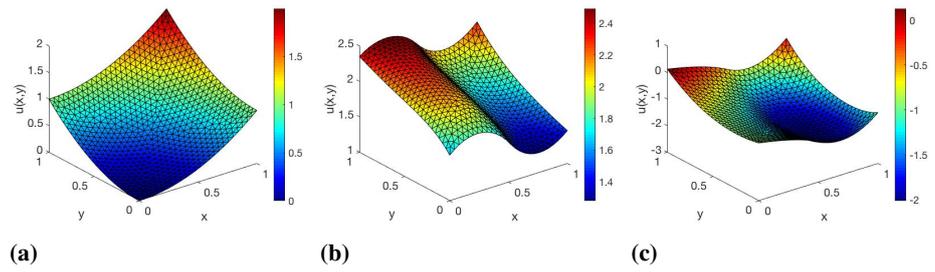


Figure 1. FE solutions of (25) for parameters (a) $\mu = (0, 0)$, (b) $\mu = (0, 1)$ and (c) $\mu = (1, 0)$.

In [28] we have already observed that the error between the FE- and RB-DEIM-solution of the state and adjoint equation decreases if the FE grids get finer. We skip the detailed description here and refer the reader to [28, Section 5].

Notice that $y(\mathbf{x}) = x_1^2 + x_2^2$ solves (25) for $\mu = (0, 0)$. For the FE-solver we used linear finite elements with $\Delta H_{\max} = 0.04$ and the finite elements have 762 degrees of freedom. We choose the following two objective functions

$$\hat{J}_1(\mu) = \frac{1}{2} \int_{\Omega} |y(\mu) - y^d|^2 \, d\mathbf{x}, \quad \hat{J}_2(\mu) = \frac{1}{2} \sum_{j=1}^2 |\mu_j - \mu_j^d|^2$$

with $\mu^d = (1, 1)$. For the desired state y^d we take the FE solution for $\mu = 0$, i.e., $y^d = y^N(0)$. Thus, $y^d = \sum_{i=1}^N y_i^d \varphi_i$ is a piecewise linear approximation of $x_1^2 + x_2^2$. The associated FE objectives are now given as

$$\hat{J}_1^N(\mu) = \frac{1}{2} (y^N(\mu) - y^d)^\top \mathbf{M} (y^N(\mu) - y^d), \quad \hat{J}_2^N(\mu) = \frac{1}{2} \sum_{j=1}^2 |\mu_j - \mu_j^d|^2$$

with $y^d = (y_i^d)_{1 \leq i \leq N}$. The gradients are

$$\nabla \hat{J}_1^N(\mu) = \begin{pmatrix} \mathbf{p}^N(\mu)^\top \bar{F}_1 \\ \mathbf{p}^N(\mu)^\top \bar{F}_2 \end{pmatrix}, \quad \nabla \hat{J}_2^N(\mu) = \mu - \mu^d$$

where $\mathbf{p}^N(\mu) = \sum_{j=1}^N \mathbf{p}_j^N(\mu) \varphi_j$ is the FE solution to (15), $\mathbf{p}^N(\mu) = (\mathbf{p}_j^N(\mu))_{1 \leq j \leq N}$ and $\bar{F}_i = \int_{\Omega} \varphi_j \xi_i \, d\mathbf{x}$ for $1 \leq j \leq N$. The associated RB objective functions have the form

$$\hat{J}_1^\ell(\mu) = \frac{1}{2} (y^\ell(\mu) - y^{d,\ell})^\top \mathbf{M}^\ell (y^\ell(\mu) - y^{d,\ell}), \quad \hat{J}_2^\ell(\mu) = \frac{1}{2} \sum_{j=1}^m |\mu_j - \mu_j^d|^2$$

with $y^{d,\ell} = \Psi^\top y^d$. In [28] we have already observed a good agreement between the approximated Pareto critical sets with the FE- and RB-DEIM-solver, if the error between the gradients is sufficiently small. However, we cannot always guarantee this agreement. Thus, we will instead use the supersets \mathbb{P}_1^ℓ and \mathbb{P}_2^ℓ from Section 2.3 for the approximation of \mathbb{P} (and \mathbb{P}_c), which are only based on the reduced objective function \hat{J}^ℓ and the error bounds ε . To generate \mathbb{P}_1^ℓ we compute the steepest descent direction for all components of \hat{J}^ℓ . Similar to Algorithm 1 we first calculate α_k as solution of (4) for μ_k . Then the descent direction is $v_k := -D\hat{J}^\ell(\mu_k)^\top \alpha_k$. As stopping condition we choose $\sigma = \|\varepsilon\|_\infty$ and set $a(\mu_k) = \mu_k$ if it holds

$$\|D\hat{J}^\ell(\mu_k)^\top \alpha_k\|_2 = \|v_k\|_2 < \sigma = \|\varepsilon\|_\infty.$$

To generate \mathbb{P}_2^ℓ we use Algorithm 2.

To save computational time during the modified subdivision algorithm, we calculate

$$\max_{1 \leq i \leq k} \|\nabla \hat{J}_i^N(\mu) - \nabla \hat{J}_i^\ell(\mu)\|_2$$

before the algorithm starts (i.e., in an offline phase) on a training set $\mathcal{P}_{\text{train}}$ which approximates \mathcal{P}_{ad} and store these errors. During the subdivision algorithm we use them to generate the new ε_{i+1}^ℓ faster and without calculating the FE solution again.

To see a significant difference between the modified subdivision algorithm and the subdivision algorithm and \mathbb{P}_2^ℓ and \mathbb{P}_1^ℓ , we need to have a big error between the gradients. To achieve this, we choose a rough training set

$$S_{\text{train}} = \{(-2 + 0.51k, -2 + 0.427j)^\top \mid (k, j) \in \{0, \dots, 7\} \times \{0, \dots, 9\}\}$$

with $|S_{\text{train}}| = 80$. For the Greedy algorithm we choose the tolerance $\varepsilon_{\text{tol}} = 0.06$ and the true error $\eta_\ell(\mu) := \max_{1 \leq i \leq k} \|\nabla \hat{J}_i^N(\mu) - \nabla \hat{J}_i^\ell(\mu)\|_2$ as error indicator, for the DEIM algorithm we choose the tolerance $\bar{\varepsilon} = 0.5$.

With these settings we generate an RB-basis with six elements and a DEIM-basis with 18 elements. This leads to the following estimations for the gradients of \hat{J}^N and \hat{J}^ℓ :

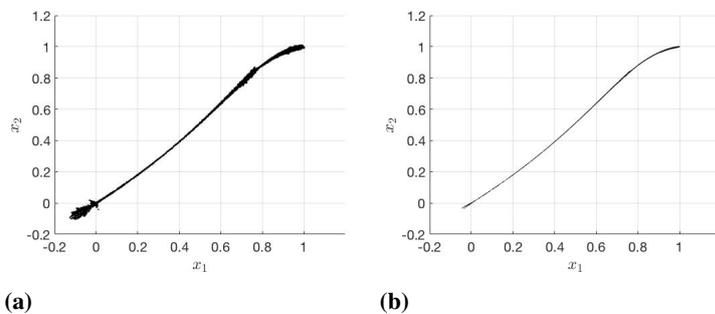
$$\begin{aligned} \max_{\mu \in \mathcal{P}_{\text{ad}}} \|\nabla \hat{J}_1^N(\mu) - \nabla \hat{J}_1^\ell(\mu)\|_2 &\approx \max_{\mu \in \mathcal{P}_{\text{train}}} \|\nabla \hat{J}_1^N(\mu) - \nabla \hat{J}_1^\ell(\mu)\|_2 \approx 0.0491 = \varepsilon_1, \\ \max_{\mu \in \mathcal{P}_{\text{ad}}} \|\nabla \hat{J}_2^N(\mu) - \nabla \hat{J}_2^\ell(\mu)\|_2 &= \max_{\mu \in \mathcal{P}_{\text{ad}}} \|\nabla \hat{J}_2(\mu) - \nabla \hat{J}_2^\ell(\mu)\|_2 = 0 = \varepsilon_2. \end{aligned}$$

To generate these estimations we choose a training set $\mathcal{P}_{\text{train}} \subset \mathcal{P}_{\text{ad}}$ with 3105 equally distributed test points and generate the error for these points. Thus, we have

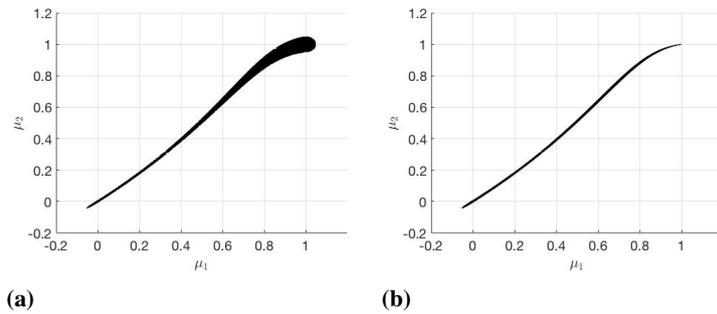
$$\varepsilon^0 = \begin{pmatrix} 0.0491 \\ 0 \end{pmatrix}.$$

217 Now we test the subdivision and the modified subdivision algorithm with the following conditions.
 218 To compute the descent direction, we use the Matlab function *fmincon* and solve (4) or (7). The
 219 algorithm stops when the box size is small enough, which is after 25 iteration steps in our case.
 220 In every step we halve the boxes. We choose five sample points in each box during the first five
 221 iteration steps, four sample points in the next five iteration steps, three sample points for the
 222 iteration steps ten to 14, two sample points for the next five steps and one sample points for the
 223 last five iteration steps, see Remark 4. Figure 2 shows the generated approximated Pareto sets for
 224 the FE-solver after 20 and 25 iteration steps.

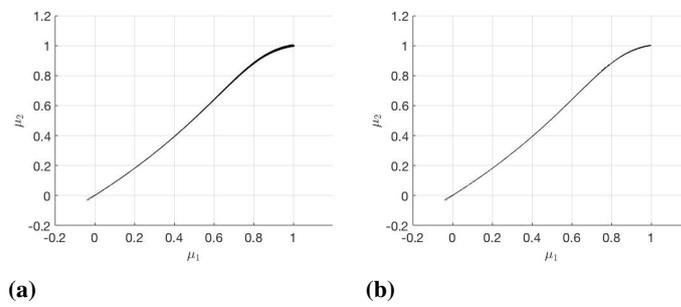
Figure 2. Pareto critical set \mathbb{P}_C in iteration step (a) 20 and (b) 25 for the FE-solver.



225 The results with the subdivision algorithm and RB-DEIM-solver are shown in Figure 3.

Figure 3. (a) \mathbb{P}_1^ℓ and (b) \mathbb{P}_2^ℓ generated with the subdivision algorithm.

226 The modified subdivision algorithm and RB-DEIM-solver lead to the results plotted in Figure 4.

Figure 4. (a) \mathbb{P}_1^ℓ and (b) \mathbb{P}_2^ℓ generated with the modified subdivision algorithm.

227 The runtime, number of boxes and number of function and gradient evaluations needed in iteration
 228 step 10, 15, 20, 25 are shown in Tables 1 and 2-5. The total runtime and number of function and
 229 gradient evaluations needed for the different methods and the speed-up are shown in Table 6.

Table 1. The performance of the subdivision algorithm with the FE-solver and the steepest descent method.

FE				
it. step	time [s]	# grad. solves	# del. boxes	# boxes
10	370.7	6872	30	64
15	1077.1	16786	166	322
20	1239.1	20092	938	1388
25	3364.2	64721	2567	4749

Table 2. Subdivision algorithm with the steepest descent method.

RB-DEIM					
it. step	time [s]	# grad. solves	# del. boxes	# boxes	ε
10	18.8	6809	32	64	(0.0491, 0)
15	42.4	14279	168	326	(0.0491, 0)
20	44.3	13781	622	3412	(0.0491, 0)
25	848.6	196405	149	97551	(0.0491, 0)

Table 3. Subdivision algorithm with the modified descent direction.

RB-DEIM						
it. step	time [s]	# grad. solves	# del. boxes	# boxes	ε	
10	21.0	6792	31	63	(0.0491, 0)	
15	43.1	15546	167	315	(0.0491, 0)	
20	38.0	12655	878	1438	(0.0491, 0)	
25	289.5	64606	233	30607	(0.0491, 0)	

Table 4. Modified subdivision algorithm with the steepest descent method.

RB-DEIM						
it. step	time [s]	# grad. solves	# del. boxes	# boxes	ε	
10	18.5	6829	32	64	(0.041, 0)	
15	44.1	16307	170	318	(0.0084, 0)	
20	45.6	16677	919	1357	(0.0072, 0)	
25	121.2	29811	560	12230	(0.0072, 0)	

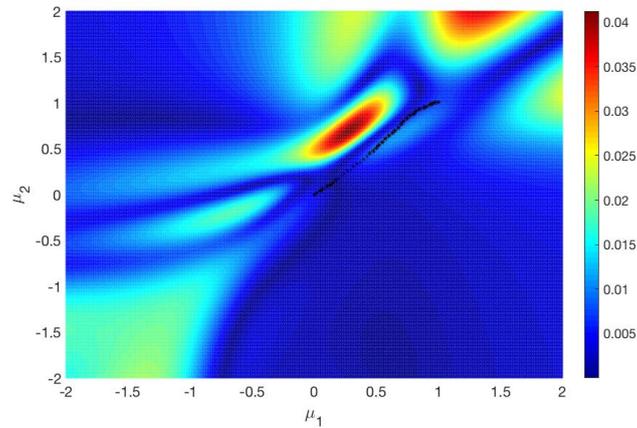
Table 5. Modified subdivision algorithm with the modified descent direction.

RB-DEIM						
it. step	time [s]	# grad. solves	# del. boxes	# boxes	ε	
10	18.7	6805	31	263	(0.041, 0)	
15	44.1	16459	168	318	(0.0084, 0)	
20	49.7	18266	937	1329	(0.0072, 0)	
25	109.7	37170	1615	4129	(0.0043, 0)	

Table 6. Total runtime, number of function and gradient evaluations for the different methods and the speed-up.

Algorithm	Solver	Method	time [min]	speed-up	# grad. solves
subdivision	FE	steep. desc.	495.83	-	610728
subdivision	RB-DEIM	steep. desc.	41.77	11.9	622034
subdivision	RB-DEIM	mod. desc.	24.77	20.0	390362
mod. subdivision	RB-DEIM	steep. desc.	21.00	23.6	372551
mod. subdivision	RB-DEIM	mod. desc.	21.91	22.6	412667

²³⁰ The Greedy algorithm and DEIM together take 14.06 seconds in the offline phase. To ensure the
²³¹ error ε (cf. (5)) for the gradients on the training set $\mathcal{P}_{\text{train}}$, it takes 151.13 seconds. It follows from
²³² Figures 3-4 that \mathbb{P}_2^ℓ is significantly smaller than \mathbb{P}_1^ℓ for the subdivision algorithm as well as for
²³³ the modified subdivision algorithm. Therefore, we have a much better approximation for \mathbb{P}_c if
²³⁴ we choose \mathbb{P}_2^ℓ instead of \mathbb{P}_1^ℓ . If we compare the two different subdivision algorithms, we notice
²³⁵ that with the modified subdivision algorithm we have a better approximation of \mathbb{P}_c than with the
²³⁶ subdivision algorithm.

Figure 5. Difference $\|\nabla \hat{J}_1^N(\mu) - \nabla \hat{J}_1^\ell(\mu)\|_2$ for $\mu \in \mathcal{P}_{\text{ad}}$.

237 The reason for this is that in the modified subdivision algorithm we have a monotonically
 238 decreasing sequence ε^l . In Figure 5 the error between $\nabla \hat{J}_1^N$ and $\nabla \hat{J}_1^\ell$ is plotted on the parameter
 239 set \mathcal{P}_{ad} . The black markers are some points on the Pareto critical set \mathbb{P}_c . It turns out that the
 240 difference between the two gradients is significantly smaller near \mathbb{P}_c than in other regions of \mathcal{P}_{ad} .
 241 Due to this, in the modified subdivision algorithm, the sequence ε^l decreases noticeably so that
 242 it is useful to update ε after each iteration step. As we have already mentioned, \mathbb{P}_2^ℓ is a better
 243 approximation of \mathbb{P}_c . If \mathbb{P}_2^ℓ is generated by the modified subdivision algorithm, the result is even
 244 better. Comparing Figure 2 (b) and Figure 4 (b), there is no significant difference between the
 245 two sets. Therefore, the modified subdivision algorithm yields a good approximation \mathbb{P}_2^ℓ for \mathbb{P}_c ,
 246 although the error ε_1 is not small.

247 Regarding the computational time (cf. Tables 1, 2-5 and 6) we notice that in the first 20 iterations
 248 the four methods take almost the same time. In these iteration steps the FE-solver takes between 19
 249 and 30 more time for one iteration step than the four RB-based methods. Only in the subsequent
 250 iteration steps a significant difference in the four RB-based methods appears. For these iteration
 251 steps, the FE-solver takes between 3.9 and 30.8 times as much time as one of the four methods for
 252 one iteration step. We notice that the computation of \mathbb{P}_1^ℓ takes around two times as much time as
 253 the computation of \mathbb{P}_2^ℓ with the subdivision algorithm. The main reason for this is the much larger
 254 number of function and gradient evaluations that we have for \mathbb{P}_1^ℓ . This, in turn, can be attributed
 255 to the significantly larger number of boxes for \mathbb{P}_1^ℓ . If we use the modified subdivision algorithm,
 256 the computation of \mathbb{P}_2^ℓ takes a bit more time than the computation of \mathbb{P}_1^ℓ . The main reason for this
 257 is again the larger number of function and gradient evaluations. Unlike the previous case, this can
 258 not be lead back to the number of boxes, but probably to the calculation of the modified descent
 259 method, which requires more function and gradient evaluations. Nevertheless, the difference in
 260 the computational time is marginal (a factor about 1.04) for the modified subdivision algorithm.
 261 For \mathbb{P}_1^ℓ we notice another behavior: Here the generation of \mathbb{P}_1^ℓ with the modified subdivision
 262 algorithm is 2 times faster than with the subdivision algorithm. The main reason for that is the
 263 larger number of function evaluations, which is due to the larger number of boxes. The FE-solver
 264 takes around 23.6 times as much time as the computation of \mathbb{P}_1^ℓ and around 22.6 times as much
 265 as the time as the computation of \mathbb{P}_2^ℓ with the modified subdivision. This is another advantage
 266 of the modified subdivision algorithm. Since we get a better approximation in this algorithm,
 267 we have fewer number of boxes in the iteration steps and therefore we have a smaller number of
 268 function and gradient evaluations than we have for the subdivision algorithm. Finally, we see that
 269 the modified subdivision algorithm works better and faster than the subdivision algorithm. Since
 270 \mathbb{P}_2^ℓ is a tighter approximation of \mathbb{P}_c , it is better to generate \mathbb{P}_2^ℓ rather than \mathbb{P}_1^ℓ .

271 5. Conclusion

272 In this article, we present a way to solve multiobjective parameter optimization problems of
 273 semilinear elliptic PDEs by combining an RB approach and DEIM with the set-oriented method

274 based on gradient evaluations. To deal with the error introduced by the surrogate model, we
 275 derived an additional condition for the descent direction, which allows us to consider the errors
 276 for the objective functions independently and derive a superset \mathbb{P}_2^ℓ of the Pareto critical set \mathbb{P}_c . To
 277 get an even tighter superset, we update these error bounds in our subdivision algorithm after each
 278 iteration step. To summarize the numerical results, we first investigated the influence of the error
 279 bounds for the gradients of the objective functions. By individually adapting the components of
 280 the error bounds we obtained a tighter covering of the Pareto critical set. When we additionally
 281 adjusted the error bounds in each iteration step, the result became even tighter and almost coincided
 282 with the exact solution of the MOP (solution with the FE-solver and the steepest descent method).
 283 Furthermore, we have compared the computational time for each method. The FE-solver needed
 284 between 11.9 times and 23.6 times more than the four different RB-based methods we presented
 285 in this work. For future work, it could be interesting to improve the results in [28, Section 3.2.4]
 286 and to develop an efficient a-posteriori error estimator for the error in the objectives and their
 287 gradients; cf., e.g., [18,21]. These error bounds can then be used in a weak greedy algorithm and
 288 beyond that for the error bounds which are needed in the subdivision algorithm.

289 **Acknowledgments:** The authors gratefully acknowledge partial support by the German DFG-Priority Pro-
 290 gram 1962. Furthermore, S. Banholzer acknowledges his partial funding by the Landesgraduiertenförderung
 291 of Baden-Württemberg.

292 **Conflicts of Interest:** The authors declare no conflict of interest.

293 Abbreviations

294 The following abbreviations are used in this manuscript:

295	MOP	Multiobjective optimization problem
	DEIM	Discrete Empirical Interpolation Method
	FE	Finite Element
	GS	Gram Schmidt
296	KKT	Karush Kuhn-Tucker
	PDE	Partial-differential equation
	POD	Proper Orthogonal Decomposition
	RB	Reduced Basis
	ROM	Reduced Order Modeling

297 Appendix A Mass Lumping

We follow [38]. Let $\Omega \subset \mathbb{R}^d$, $d \in \{1, 2, 3\}$ be an open, bounded Lipschitz domain, e.g.
 $\Omega = (0, 1)^d$. We set $H = L^2(\Omega)$ and $V = H^1(\Omega)$, where V is endowed with the inner product

$$\langle \varphi, \varphi \rangle_V = \int_{\Omega} \nabla \varphi \cdot \nabla \varphi \, dx + \int_{\Gamma} \varphi \varphi \, ds \quad \text{for } \varphi, \varphi \in V$$

298 and its induced norm. Let \mathcal{T}^N denotes an underlying triangulation of Ω and $V(\mathcal{T}^N) = \{z^1, \dots, z^N\}$
 299 denotes the set of interior vertices of \mathcal{T}^N and let $V(\tau)$ be the set of vertices belongig to $\tau \in \mathcal{T}^N$.
 300 $V^N = \text{span}\{\varphi^1, \dots, \varphi^N\}$ be the FE space generated by first order Lagrange elements. It holds
 301 $\varphi^j(z^l) = \delta_{jl}$.

Define for $z^j \in V(\mathcal{T}^N)$ a lumped mass region Ω^j by joining the centroids of the triangles,
 which have z^j as a common vertex, to the midpoint of the edges, which have z^j as a common
 extremity. We define the two sets

$$V^N = \{v \in C(\bar{\Omega}) \mid v|_{\tau_j} \in P_1 \text{ for all } \tau_i \in \mathcal{T}^N\} \subset V,$$

$$H^N = \left\{ v \in L^\infty(\Omega) \mid v = \sum_{j=1}^N \mathbb{1}_{\Omega^j} v^j, v^j \in \mathbb{R} \right\} \subset H$$

and the operator

$$\mathcal{R}^N : C(\overline{\Omega}) \rightarrow H^N, \quad \mathcal{R}^N(\omega) = \sum_{j=1}^N \mathbb{1}_{\Omega^j} \omega(z^j).$$

We consider the following semilinear elliptic partial differential equation:

$$-\Delta y + f(y) = h \text{ in } \Omega, \quad \frac{\partial y}{\partial \mathbf{n}} + y = g \text{ on } \Gamma = \partial\Omega. \quad (\text{A1})$$

A weak solution of (A1) is a function $y \in \mathcal{Y} = V \cap L^\infty(\Omega)$ such that

$$\int_{\Omega} \nabla y \cdot \nabla \varphi + f(y)\varphi \, dx + \int_{\Gamma} y\varphi \, ds = \int_{\Omega} h\varphi \, dx + \int_{\Gamma} g\varphi \, ds \quad \text{for all } \varphi \in V. \quad (\text{A2})$$

The function f is supposed to be sufficiently smooth, bounded and measurable for a fixed y , monotonically increasing and satisfies $f \geq 0$. Moreover, $h \in H$ and $g \in L^r(\Gamma)$ with $s > \mathfrak{d} - 1$. Then, there exists a unique solution $y \in \mathcal{Y}$ of (A1), c.f. [36, Section 4.2.3], for instance. This solution is even continuous and for a constant c_∞ it holds

$$\|y\|_V + \|y\|_{C(\overline{\Omega})} \leq c_\infty (\|h\|_H + \|g\|_{L^r(\Gamma)}).$$

The lumped mass finite element approximation of (A1) is find $y^N \in V^N$ such that

$$\int_{\Omega} \nabla y^N \cdot \nabla \varphi + f(\mathcal{R}^N(y^N))\varphi^N \, dx + \int_{\Gamma} y^N \varphi \, ds = \int_{\Omega} h\varphi \, dx + \int_{\Gamma} g\varphi \, ds \quad (\text{A3})$$

302 holds for all $\varphi \in V^N$ and for $\varphi^N = \mathcal{R}^N(\varphi)$.

Remark A1. For any $v \in C(\overline{\Omega})$ it holds $\mathcal{R}^N(v) = \sum_{j=1}^N \mathbb{1}_{\Omega^j} v(z^j)$. Therefore, $\mathcal{R}^N(v) \rightarrow v$ for $n \rightarrow \infty$, which implies

$$\|\mathcal{R}^N(v) - v\|_H \rightarrow 0 \quad \text{for } n \rightarrow \infty.$$

303

◇

Theorem A1. Assume that $y \in V$ and $y^N \in V^N$ are solutions of (A2) and (A3), respectively. Then it holds

$$\|y - y^N\|_V \rightarrow 0 \quad \text{for } N \rightarrow \infty.$$

Proof. Because y solves (A2), it holds $y \in C(\overline{\Omega})$. Let $y^p \in V^N$ be the interpolation polynomial of y in V^N . Then it follows that $\mathcal{R}^N(y^p) = \mathcal{R}^N(y)$. Utilizing (A3) we derive

$$-\int_{\Omega} \nabla y^N \cdot \nabla y^p - \int_{\Gamma} y^N y^p \, ds = \int_{\Omega} f(\mathcal{R}^N(y^N))\mathcal{R}^N(y) - h y^p \, dx - \int_{\Gamma} g y^p \, ds. \quad (\text{A4})$$

Let $\varepsilon > 0$ be an arbitrary tolerance. For N large enough we have $\|y - y^p\|_V < \varepsilon$. Furthermore we have $\|\mathcal{R}^N(y) - y\|_H < \varepsilon$ and $\|\mathcal{R}^N(y^p - y^N) - (y^p - y^N)\|_H < \varepsilon$. From (A4) we conclude

$$\begin{aligned} \|y^N - y\|_V^2 &= \int_{\Omega} |\nabla(y^N - y)|^2 dx + \int_{\Gamma} |y^N - y|^2 ds \\ &= \int_{\Omega} \left(\nabla(y^N - y) \cdot \nabla(y^p - y) + \nabla(y^N - y) \cdot \nabla y^N + \nabla(y - y^N) \cdot \nabla y^p \right) dx \\ &\quad + \int_{\Gamma} \left((y^N - y)(y^p - y) + (y^N - y)y^N + (y - y^N)y^p \right) ds \\ &= \int_{\Omega} \left(\nabla(y^N - y) \cdot \nabla(y^p - y) + \nabla(y^N - y) \cdot \nabla y^N + \nabla y \cdot \nabla y^p \right) dx \\ &\quad + \int_{\Gamma} \left((y^N - y)(y^p - y) + (y^N - y)y^N + yy^p \right) ds \\ &\quad + \int_{\Omega} \left(f(\mathcal{R}^N(y^N))\mathcal{R}^N(y) - hy^p \right) dx - \int_{\Gamma} gy^p ds. \end{aligned}$$

Utilizing $\|y - y^p\|_V < \varepsilon$, $\|\mathcal{R}^N(y) - y\|_H < \varepsilon$ and $\|\mathcal{R}^N(y^p - y^N) - (y^p - y^N)\|_H < \varepsilon$ it follows that

$$\begin{aligned} \int_{\Omega} \left(f(\mathcal{R}^N(y^N))\mathcal{R}^N(y) \right) dx &= \langle f(\mathcal{R}^N(y^N)), \mathcal{R}^N(y) \rangle_H \\ &= \langle f(\mathcal{R}^N(y^N)) - f(\mathcal{R}^N(y)), \mathcal{R}^N(y) - \mathcal{R}^N(y^N) \rangle_H \\ &\quad + \langle f(\mathcal{R}^N(y^N)), \mathcal{R}^N(y^N) \rangle_H + \langle f(\mathcal{R}^N(y)), \mathcal{R}^N(y) - \mathcal{R}^N(y^N) \rangle_H \\ &\leq \langle f(\mathcal{R}^N(y^N)), \mathcal{R}^N(y^N) \rangle_H + \langle f(\mathcal{R}^N(y)), \mathcal{R}^N(y) - \mathcal{R}^N(y^N) \rangle_H \\ &= \langle f(\mathcal{R}^N(y)), \mathcal{R}^N(y - y^N) - (y - y^N) \rangle_H + \langle f(y), y - y^N \rangle_H \\ &\quad + \langle f(\mathcal{R}^N(y)) - f(y), (y - y^N) \rangle_H + \langle f(\mathcal{R}^N(y^N)), \mathcal{R}^N(y^N) \rangle_H \\ &\leq c(\|y - y^p\|_H + \|\mathcal{R}^N(y^p - y^N) - (y^p - y^N)\|_H) + \langle f(y), y - y^N \rangle_H \\ &\quad + c\|\mathcal{R}^N(y) - y\|_H\|y - y^N\|_H + \langle f(\mathcal{R}^N(y^N)), \mathcal{R}^N(y^N) \rangle_H \\ &\leq c(2\varepsilon + \varepsilon\|y - y^N\|_H) + \langle f(y), y - y^N \rangle_H + \langle f(\mathcal{R}^N(y^N)), \mathcal{R}^N(y^N) \rangle_H. \end{aligned}$$

Inserting this estimation into the equality before thus gives

$$\begin{aligned} \|y - y^N\|_V^2 &\leq \varepsilon\|y - y^N\|_V^2 + \frac{\|y^p - y\|_V^2}{4\varepsilon} + \langle \nabla y^N, \nabla(y^N - y) \rangle_{H^0} + \langle \nabla y^p, \nabla y \rangle_{H^0} \\ &\quad + \langle y^N - y, y^p - y \rangle_{L^2(\Gamma)} + \langle y^N, y^N - y \rangle_{L^2(\Gamma)} + \langle y^p, y \rangle_{L^2(\Gamma)} - \langle y^p, h \rangle_H \\ &\quad - \langle y^p, g \rangle_{L^2(\Gamma)} + c\varepsilon + c\varepsilon\|y - y^N\|_H + \langle g, y - y^N \rangle_{L^2(\Gamma)} - \langle y, y - y^N \rangle_{L^2(\Gamma)} \\ &\quad + \langle h, y - y^N \rangle_H - \langle \nabla y, \nabla(y - y^N) \rangle_{H^0} + \langle f(\mathcal{R}^N(y^N)), \mathcal{R}^N(y^N) \rangle_H \\ &\leq c\varepsilon\|y - y^N\|_V^2 + \frac{\varepsilon^2}{4\varepsilon} + c\varepsilon\langle \nabla y^N, \nabla y \rangle_{H^0} + \langle \nabla y^p, \nabla y \rangle_{H^0} \\ &\quad + \langle g, y - y^p \rangle_{L^2(\Gamma)} + \langle h, y - y^p \rangle_H + \langle y^N, y^p - y \rangle_{L^2(\Gamma)} - \langle \nabla y, \nabla(y - y^N) \rangle_{H^0} \\ &\leq c\varepsilon + c\varepsilon\|y - y^N\|_V^2 \end{aligned}$$

with $H^0 = \otimes_{i=1}^d H$. In the last inequality we used the fact that $y^p \rightarrow y$ for $N \rightarrow \infty$. Therefore we get

$$\|y - y^N\|_V^2 \leq c_0\varepsilon$$

304 and it follows $\|y - y^N\|_V \rightarrow 0$ for $N \rightarrow \infty$. □

Remark A2. For $y^N \in V^N$ and $j = 1, \dots, N$ we find

$$\begin{aligned} \int_{\Omega} f(\mathcal{R}^N(y^N))\mathcal{R}^N(\varphi^j) \, dx &= \sum_{i=1}^N \int_{\Omega_i} f\left(\sum_{l=1}^N y_l^N \mathbb{1}_{\Omega_l}\right) \mathbb{1}_{\Omega_j} \, dx \\ &= \int_{\Omega_j} f\left(\sum_{l=1}^N y_l^N \mathbb{1}_{\Omega_l}\right) \, dx = f(y_j^N) \int_{\Omega_j} \mathbb{1}_{\Omega_j} \, dx \\ &= f(y_j^N) \sum_{\tau \in \mathcal{T}^N: z^j \in V(\tau)} \frac{|\tau|}{n+1} = \tilde{M}_{jj} f(y_j^N). \end{aligned}$$

For the last step we use

$$\tilde{M}_{jj} = \sum_{k=1}^N M_{jk} = \sum_{\tau \in \mathcal{T}^N: z^j \in V(\tau)} \int_{\tau} \varphi_j \, dx = \sum_{\tau \in \mathcal{T}^N: z^j \in V(\tau)} \frac{|\tau|}{n+1}.$$

³⁰⁵ A detailed proof for this equality can be found in [3, Appendix A]. ◇

References

1. Banholzer, S.; Gebken, B.; Dellnitz, M.; Peitz, S.; Volkwein, S. ROM-based multiobjective optimization of elliptic PDEs via numerical continuation. Accepted, available at arXiv:1906.09075v1
2. Beermann, D.; Dellnitz, M.; Peitz, S.; Volkwein, S. Set-oriented multiobjective optimal control of PDEs using proper orthogonal decomposition. In *Reduced-Order Modeling (ROM) for Simulation and Optimization: Powerful Algorithms as Key Enablers for Scientific Computing*, Springer International Publishing, Cham, pp. 47-72, **2018**.
3. Bernreuther, M. *RB-Based PDE-Constrained Non-Smooth Optimization*. Master thesis, University of Konstanz, **2019**, available at <http://nbn-resolving.de/urn:nbn:de:bsz:352-2-t4k1djy77yn3>.
4. Brenner S.; Scott, R. *The Mathematical Theory of Finite Element Methods*, Springer-Verlag Berlin Heidelberg, **2008**.
5. Chaturantabut, S.; Sorensen, D.C. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, **2010**, *32*, 2737–2764.
6. Chaturantabut, S.; Sorensen, D.C. A state space estimate for POD-DEIM nonlinear model reduction. *SIAM Journal on Numerical Analysis*, **2012**, *50*, 46–63.
7. Dautray, R; Lions, J.-L. *Mathematical Analysis and Numerical Methods for Science and Technology. Volume 2: Functional and Variational Methods*. Springer-Verlag Berlin Heidelberg, **2000**.
8. Dellnitz, M.; Hohmann, A. A subdivision algorithm for the computation of unstable manifolds and global attractors. *Numer. Math.* **75**,293-316 **1997**, <https://doi.org/10.1007/s002110050240>
9. Dellnitz, M.; Schütze, O.; Hestermeyer, T. Covering Pareto sets by multilevel subdivision techniques. *Journal of Optimization Theory and Applications*, **2005**, *124*, 113-136.
10. Dellnitz, M.; Witting, K. Computation of robust Pareto points. *International Journal of Computing Science and Mathematics*, **2009**, *2*, 243-266.
11. Deb, K. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Inc., USA, **2001**.
12. Ehrgott, M. *Multicriteria Optimization*. Springer Berlin Heidelberg New York, **2005**.
13. Evans, L.C. *Partial Differential Equations*. American Math. Society, Providence, Rhode Island, **2002**.
14. Fliege, J; Graña Drummond, L.M.; Svaiter, B.F. Netwon’s Method for multiobjective optimization. *SIAM Journal on Optimization*, **2008**, *20*, 602-626.
15. Fliege, J; Svaiter, B.F. Steepest descent methods for multicriteria optimization. *Mathematical Methods of Operations Research*, **2000**, *51*, 479-494.
16. Gebken, B.; Peitz, S.; Dellnitz, M. A descent method for equality and inequality constrained multiobjective optimization problems. In *Numerical and Evolutionary Optimization – NEO 2017*, L. Trujillo, O. Schütze, Y. Maldonado, and P. Valle (eds). Studies in Computational Intelligence, vol 785. Springer, Cham, **2017**.
17. Graña Drummond, L.M.; Svaiter, B.F. A steepest descent method for vector optimization. *J. Comput. Appl. Math.*, **2005**, *175*, 395-414.
18. Grepl, M.A.; Maday, Y.; Nguyen, N.C.; Patera, A.T. Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. *ESAIM: Mathematical Modelling and Numerical Analysis*, **2007**, *41*, 575-605.
19. Hesthaven, J.S.; Rozza, G.; Stamm, B. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. SpringerBriefs in Mathematics, Heidelberg, **2016**.
20. Hillermeier, C. *Nonlinear Multiobjective Optimization: A Generalized Homotopy Approach*. Birkhäuser, **2001**.
21. Hinze, M.; Korolev, D. Reduced basis methods for quasilinear elliptic PDEs with applications to permanent magnet synchronous motors. Submitted, **2020**, available at arXiv:2002.04288

22. Iapichino, L.; Ulbrich, S.; Volkwein, S. Multiobjective PDE-constrained optimization using the reduced-basis method. *Advances in Computational Mathematics*, **2017**, *43*, 945-972.
23. Jahn, J. Multiobjective search algorithm with subdivision technique. *Computational Optimization and Applications*, **2006**, *35*, 161-175.
24. Kuhn, H.; Tucker, A. Nonlinear programming. In Proceedings of the Berkeley Symposium on Mathematical and Statistical Probability, J. Neumann (ed.), University of California at Berkeley, Berkeley, California, pages 481-492, **1951**.
25. Miettinen, K. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, **1999**.
26. Patera, A.T.; Rozza, G. *Reduced Basis Approximation and A Posteriori Error Estimation for Parametrized Partial Differential Equations*. MIT Pappalardo Graduate Monographs in Mechanical Engineering, **2007**.
27. Quarteroni, A.; Manzoni, A.; Negri, F. *Reduced Basis Methods for Partial Differential Equations*. Unitext Series. Springer, **2016**.
28. Reichle, L. *Set-oriented multiobjective optimal control of elliptic non-linear partial differential equations using POD objectives and gradient*. Master thesis, University of Konstanz, **2020**, available at <http://nbn-resolving.de/urn:nbn:de:bsz:352-2-1h6pp1cxptbap6>
29. Rogg, S.; Trenz, S.; Volkwein, S. Trust-region POD using a-posteriori error estimation for semilinear parabolic optimal control problems. *Konstanzer Schriften in Mathematik*, **2017**, *359* available at <https://kops.uni-konstanz.de/handle/123456789/38240>
30. Rösch A.; Wachsmuth, G. Mass lumping for the optimal control of elliptic partial differential equations. *SIAM Journal on Numerical Analysis*, **2017**, *55*, 1412-1436.
31. Schäffler, S.; Schultz, R.; Weinzierl; K. Stochastic method for the solution of unconstrained vector optimization problems. *Journal of Optimization Theory and Applications*, **2002**, *114*, 209-222.
32. Schilders, W.H.; Van der Vorst, H.A.; Rommes, J. *Model Order Reduction*. Springer-Verlag Berlin Heidelberg, **2008**.
33. Schütze, O.; Dell'Aere, A.; Dellnitz, M. On continuation methods for the numerical treatment of multi-objective optimization problems. In *Practical Approaches to Multi-Objective Optimization*, J. Branke, K. Deb, K. Miettinen, and R.E. Steuer (eds.), Dagstuhl Seminar Proceedings, Dagstuhl, Deutschland, **2005**.
34. Schütze, O.; Witting, K.; Ober-Blöbaum, S.; Dellnitz, M. Set oriented methods for the numerical treatment of multiobjective optimization problems. In *EVOLVE – A Bridge between Probability, Set Oriented Numerics and Evolutionary Computation*, E. Tantar, A.-A. Tantar, P. Bouvry, P. Del Moral, P. Legrand, C.A. Coello Coello, and O. Schütze (eds.), Studies in Computational Intelligence, Springer Berlin Heidelberg, **2013**, *447*, 187-219.
35. Thomée, V. *Galerkin Finite Element Methods for Parabolic Problems*. Springer-Verlag Berlin Heidelberg, **2006**.
36. Tröltzsch, F. *Optimal Control of Partial Differential Equations: Theory, Methods and Applications*. American Mathematical Society, Graduate Studies in Mathematics, vol. 112, **2010**.
37. Zadeh, L. Optimality and non-scalar-valued performance criteria. *IEEE Transactions on Automatic Control*, **1963**, *8*, 59-60.
38. Zeng, J.; Yu, H. Error estimates of the lumped mass finite element method for semilinear elliptic problems. *Journal of Computational and Applied Mathematics*, **2012**, *236*, 993-2004.
39. Zienkiewicz, O.C.; Taylor R. L. *The finite element method*. Vol. 3. Butterworth-Heinemann, Oxford, fifth edition, **2000**. Fluid dynamics.