Inexact Trust-Region Methods for PDE-Constrained Optimization

Drew P. Kouri and Denis Ridzal

Abstract Numerical solution of optimization problems with partial differential equation (PDE) constraints typically requires inexact objective function and constraint evaluations, derivative approximations, and the use of iterative linear system solvers. Over the last 30 years, trust-region methods have been extended to rigorously, robustly, and efficiently handle various sources of inexactness in the optimization process. In this chapter, we review some of the recent advances, discuss their key algorithmic contributions, and present numerical examples that demonstrate how inexact computations can be exploited to enable the solution of large-scale PDE-constrained optimization problems.

1 Introduction

Numerical solution of optimization problems constrained by partial differential equations (PDEs)—and, more generally, optimization problems involving large-scale nonlinear simulations—poses a number of mathematical, algorithmic, and computational challenges. The computational challenge often lies in the sheer size of the discretized problem. Specifically, the computational expense of solving a single instance of the governing PDEs can make the solution of the optimization problem a daunting task. To make numerical solution practical, one frequently resorts to approximating the objective function and its derivatives. Similarly, one may use approximations of the constraint function, its derivatives, and their inverses. In order to ensure convergence to a solution of the original infinite-dimensional problem, however, these approximations must be intelligently managed and refined. Trust-region methods provide a robust, globally convergent framework to handle

D. P. Kouri · D. Ridzal (✉)
Center for Computing Research, Sandia National Laboratories, Albuquerque, NM 87185-9999, USA
e-mail: dpkouri@sandia.gov; dridzal@sandia.gov

H. Antil et al. (eds.), *Frontiers in PDE-Constrained Optimization*, The IMA Volumes in Mathematics and its Applications 163,
https://doi.org/10.1007/978-1-4939-8636-1_3

multiple forms of inexactness, including inexact evaluations of the objective and constraint functions and their derivatives, as well as the inexact linear system solves arising in the approximate application of constraint derivative inverses. For this reason, trust regions are a popular choice for large-scale, nonconvex multidisciplinary optimization with simulation constraints.

Early works by Moré [29], Toint [40], and Carter [10–12] pioneered the use of inexact gradients and objective function values within trust-region methods. Later in the 1990s, Alexandrov, Dennis, and Torczon analyzed trust-region algorithms as a general framework for managing approximations throughout the optimization iteration [1–4, 15, 16]. These works laid the foundation for more recent works on adaptive approximations, where the trust-region framework is used to manage the accuracy of objective and constraint function evaluations, objective gradient computations, constraint derivative operator applications, and linear system solves [5, 8, 18, 20, 21, 24–26, 43, 44]—collectively labeled *inexact trust-region methods*.

In this chapter, we review two inexact trust-region algorithms, for unconstrained and equality-constrained optimization, respectively, using PDE-constrained optimization as motivation. For each algorithm, we discuss the state of the art in terms of variable-fidelity and inexact computations. Our goal is threefold: to document the algorithms in sufficient, "ready-to-use" detail, demonstrate them on a novel numerical example, and provide a concise reference to other recent works on managing inexactness in trust-region methods for large-scale optimization. The remainder of the chapter is structured as follows. In Section 2, we introduce the notation. In Section 3, we discuss two common problem formulations used in PDE-constrained optimization, the reduced-space and the full-space formulation. In Section 4, we review an inexact trust-region algorithm for unconstrained optimization and an inexact composite-step trust-region algorithm for equality-constrained optimization. Subsequently, in Section 5 we specialize the algorithms to optimization problems constrained by PDEs with random coefficients. The discussion of full-space methods includes a novel highly parallelizable preconditioner for optimization under uncertainty where the statistics are evaluated through sampling, e.g., using sparse grids [37] or Monte-Carlo methods. The preconditioner is an extension of recent work on constraint-based "optimal" preconditioners for PDE-constrained optimization [33] and their specialization to augmented systems [36]. In Section 6, we present new numerical results for the risk-neutral optimization of thermal-fluid models with random inputs. Finally, we discuss our conclusions in Section 7.

2 Notation

The norm associated with the Banach space V is $\|\cdot\|_V$. When V is Hilbert, we denote the inner product by $\langle v, w\rangle_V$ and the norm $\|v\|_V = \sqrt{\langle v, v\rangle_V}$. If V and W are Banach spaces then we denote the space of bounded linear operators between V and W by $\mathcal{L}(V, W)$. When $V = W$, we denote $\mathcal{L}(V) := \mathcal{L}(V, V)$. Moreover, when $W = \mathbb{R}$, we denote by $V^* := \mathcal{L}(V, \mathbb{R})$ the topological dual space of V and

by $\langle f, v\rangle_{V^*,V}$ the duality pairing between $f \in V^*$ and $v \in V$. If $A \in \mathcal{L}(V, W)$, we denote its adjoint by $A^* \in \mathcal{L}(W^*, V^*)$. When V is a Hilbert space, we identify V^* with V using the Riesz Representation Theorem. We note that the algorithms described in this paper can be equivalently formulated using the notion of the Riesz isomorphism $R \in \mathcal{L}(V^*, V)$. Finally, we denote the identity operator on the Banach space V by $I_V \in \mathcal{L}(V)$.

3 Problem Formulations

An important feature of algorithms for PDE-constrained optimization is convergence in function space, leading to mesh-independent convergence after discretization. Therefore, we consider the function space setting for the formulation of PDE-constrained optimization problems. We will discuss their discrete forms, enabling numerical solution, in Section 5. Let U and Z be Hilbert spaces and C be a reflexive Banach space. U denotes the *state space* and $u \in U$ is a *state variable*. Similarly, Z is the *control space* and $z \in Z$ is a *control variable*. We wish to solve the optimization problem

$$\min_{u\in U,\ z\in Z} \quad J(u, z) \tag{1a}$$

$$\text{subject to} \quad c(u, z) = 0\,, \tag{1b}$$

where the objective $J : U \times Z \to \mathbb{R}$ and the constraint $c : U \times Z \to C$ are smooth functions, with the smoothness requirements made precise later.

In addition to the optimization problem (1), which we refer to as the *full-space* problem, we will consider its *reduced-space* companion. In general, the reduced-space problem is obtained through nonlinear elimination of the state variables. Specifically, we assume the existence of the *solution operator* $S : Z \to U$ such that

$$u = S(z) \quad \text{satisfies} \quad c(u, z) = 0\,,$$

and define the *reduced objective function* $\mathcal{J} : Z \to \mathbb{R}$ by

$$\mathcal{J}(z) := J(S(z), z)\,.$$

Instead of the optimization problem (1), we can then solve

$$\min_{z\in Z} \quad \mathcal{J}(z)\,. \tag{2}$$

We note that in general (1) and (2) are not equivalent. For example, problem (1) may have a solution even when the solution operator S does not exist, i.e., when problem (2) cannot be solved. Additionally, we will see later that certain types of inexactness in J, c, and their partial derivatives are more easily handled through the

reduced-space form, while others are better suited to the full-space form. Finally, we consider a third problem formulation, resulting from a simple change of notation in problem (1). We define $X := U \times Z$, and for $x \in X$ write

$$\min_{x \in X} \quad J(x) \tag{3a}$$

$$\text{subject to} \quad c(x) = 0\,. \tag{3b}$$

This formulation is a generalization of (1), in that it does not assume an explicit splitting of variables into state and control variables, potentially resulting in algorithmic advantages.

When discussing algorithms for the full-space problem (1), we require the Lagrangian functional

$$L(u, z, \lambda) := J(u, z) + \langle \lambda, c(u, z) \rangle_{C^*, C}.$$

Furthermore, under standard assumptions, the reduced objective function $\mathcal{J}$ of (2) is twice continuously Fréchet differentiable and the first derivative is given by

$$\nabla \mathcal{J}(z) = c_z(S(z), z)^* \Lambda(S(z), z) + \nabla_z J(S(z), z) \in Z$$

where $\Lambda(u, z) = \lambda \in C^*$ solves the adjoint equation

$$c_u(u, z)^* \lambda = -\nabla_u J(u, z). \tag{4}$$

Here, we denote the partial derivatives of c by c_u and c_z, and the partial derivatives of J by $\nabla_u J$ and $\nabla_z J$. A similar expression exists for the application of the Hessian of $\mathcal{J}$ to a vector $v \in Z$. In particular, we can represent the action of the Hessian by

$$\nabla^2 \mathcal{J}(z) v = T(S(z), z)^* H(S(z), z, \Lambda(S(z), z)) T(S(z), z) v \tag{5}$$

where the linear operator $T(u, z)$ is defined as

$$T(u, z) := \begin{pmatrix} -c_u(u, z)^{-1} c_z(u, z) \\ I_Z \end{pmatrix},$$

and the linear operator $H(u, z, \lambda)$ is defined as

$$H(u, z, \lambda) := \begin{pmatrix} \nabla_{uu} L(u, z, \lambda) & \nabla_{uz} L(u, z, \lambda) \\ \nabla_{zu} L(u, z, \lambda) & \nabla_{zz} L(u, z, \lambda) \end{pmatrix},$$

see, e.g., [22, Ch. 1]. As above, $\nabla_{uu} L$, $\nabla_{uz} L$, $\nabla_{zu} L$, and $\nabla_{zz} L$ denote the second-order partial derivatives of the Lagrangian functional.

Formulations (1), (2), and (3) of PDE-constrained optimization problems have been studied extensively in the context of inexact trust-region methods. Before

discussing the methods, we present a "menu" of possible assumptions on the full-space formulation (1), which are used to establish the applicability and global convergence of each of the methods. We note that some assumptions are shared by all methods, while others are formulation-specific and method-specific. Throughout, we assume the existence of a convex open set $\Omega \subseteq X$ such that the iterates x_k and trial steps s_k produced by the algorithms described in the subsequent sections satisfy x_k, $x_k + s_k \in \Omega$ for all k. For the reduced space formulation, we interpret $x_k = (S(z_k), z_k)$ and assume $(S(z_k + t\sigma_k), z_k + t\sigma_k) \in \Omega$ for all $t \in [0, 1]$ where $\sigma_k \in Z$ denotes the trial step produced by the reduced-space algorithm.

(A1) The functional J is bounded below and finite on Ω.
(A2) The functions J and c are twice continuously Fréchet differentiable on Ω.
(A3) The Jacobian $c_x(x)$ is uniformly bounded on Ω.
(A4) The Hessians $\nabla_{xx} J(x)$ and $c_{xx}(x)$ are uniformly bounded in $\mathcal{L}(X)$ and $\mathcal{L}(X, \mathcal{L}(X, C))$, respectively, on Ω.
(A5) The operator $c_x(x)$ is surjective for all $x \in \Omega$.
(A6) The state Jacobian $c_u(x)$ is continuously invertible for all $x \in \Omega$ and the inverse $c_u(x)^{-1}$ is uniformly bounded on Ω.
(A7) The solution operator S exists and is unique for all $z \in Z$.
(A8) The operator $T(x)^* H(x, \Lambda(x)) T(x)$ is uniformly bounded on Ω.
(A9) The following function(al)s and operators are uniformly bounded over all $x \in \Omega$: $J(x)$, $\nabla_x J(x)$, $c(x)$, and $(c_x(x) c_x(x)^*)^{-1}$.

Remark 1 When the PDE in (1) is nonlinear (e.g., semilinear), the range space of the solution operator S typically must be more regular than the space U, in order to ensure that c is Fréchet differentiable and that $c_u(u, z)$ has a bounded inverse. Although this is an important issue, we focus on the stated problem setting to simplify the presentation. We refer the interested reader to [41] for information on handling this more general setting.

4 Inexact Trust-Region Methods

We begin with the description of an inexact trust-region approach for the reduced-space formulation, (2), which was studied in [25, 26]. This is followed by the discussion of a scheme for the full-space formulation (3), originally presented in [20].

4.1 A Reduced-Space Approach

In this section, we focus on the reduced-space formulation, (2), of PDE-constrained optimization problems. Given an iterate z_k, the basic trust-region algorithm builds a smooth local model $m_k : Z \to \mathbb{R}$ of the objective function $s \mapsto \mathcal{J}(z_k + s)$ inside

the trust region $\mathcal{B}_k := \{s \in Z : \|s\| \leq \Delta_k\}$, where $\Delta_k > 0$ is the trust-region radius. The algorithm then computes a trial step s_k by approximately solving the trust-region subproblem

$$\min_{s \in Z} m_k(s) \quad \text{subject to} \quad \|s\|_Z \leq \Delta_k. \tag{6}$$

In order to ensure convergence, the trial step s_k must satisfy the fraction of Cauchy decrease condition

$$m_k(0) - m_k(s_k) \geq \kappa_{\text{fcd}} \|\nabla m_k(0)\|_Z \min\left\{\Delta_k, \frac{\|\nabla m_k(0)\|_Z}{\beta_k}\right\}, \tag{7}$$

where $\kappa_{\text{fcd}} > 0$ is fixed and $\beta_k = 1 + \sup_{s \in \mathcal{B}_k} \|\nabla^2 m_k(s)\|_{\mathcal{L}(Z)}$. When m_k in (6) is quadratic, a number of solvers exist to compute s_k that satisfies (7). For example, the Cauchy point, (double) dogleg, truncated Conjugate Gradient and truncated Lanczos algorithms all produce steps that satisfy (7), see [13, 17] and the references within for more information. Once the step s_k is computed, trust-region algorithms check whether $z_k + s_k$ is acceptable as the new iterate, and the trust-region radius Δ_k is updated accordingly. Step acceptance and the trust-region update depend on the ratio of *actual* and *predicted reduction*,

$$\text{ared}_k := (\mathcal{J}(z_k) - \mathcal{J}(z_k + s_k)) \quad \text{and} \quad \text{pred}_k := (m_k(0) - m_k(s_k)),$$

respectively. In particular, given $0 < \eta_1 < \eta_2 < 1$, the trial step s_k is accepted if the actual reduction is larger than a fraction of the predicted reduction, i.e.,

$$\text{ared}_k \geq \eta_1 \text{pred}_k.$$

If the trial step is rejected, then the trust-region radius is decreased, whereas if it is accepted and the actual reduction is sufficiently large, i.e.,

$$\text{ared}_k \geq \eta_2 \text{pred}_k,$$

then the trust-region radius is increased.

Under standard assumptions, if the objective function can be evaluated exactly for all k, the model m_k is first-order consistent with $\mathcal{J}$ in the sense that

$$\nabla \mathcal{J}(z_k) = \nabla m_k(0)$$

for all k and the trial steps s_k satisfy condition (7), then one can prove global convergence of the trust-region scheme. However, if the objective function and its gradient can only be approximated, i.e., are evaluated inexactly, additional conditions are needed to ensure global convergence. We first state the condition on gradient inexactness.

Gradient Condition For all k the model m_k must approximate the objective function $s \mapsto \mathcal{J}(z_k + s)$ so that the true and approximate gradients at $s = 0$ satisfy

$$\|\nabla m_k(0) - \nabla \mathcal{J}(z_k)\|_Z \leq \kappa_{\text{grad}} \min\{\|\nabla m_k(0)\|_Z, \Delta_k\}. \tag{8}$$

Here, $\kappa_{\text{grad}} > 0$ is independent of k. A similar condition was originally proposed by Carter in [11]. However, (8) is due to Heinkenschloss and Vicente [21]. □

The definition of the actual reduction, ared_k, involves the exact value of the objective function $\mathcal{J}$, which we often cannot compute. Instead, we consider a computable approximation $\mathcal{J}_k$, where the subscript k indicates that the approximation may change from iteration to iteration. With this approximation, we can define the *computed reduction*

$$\text{cred}_k := \mathcal{J}_k(z_k) - \mathcal{J}_k(z_k + s_k). \tag{9}$$

To ensure convergence of the trust-region algorithm, we must ensure that the difference

$$|\text{ared}_k - \text{cred}_k|,$$

is "sufficiently" small. We now state the objective function conditions.

Objective Function Conditions Assume that there exists an estimator $\theta_k = \theta(z_k, s_k)$ of the error in the objective function so that for a constant $K > 0$,

$$|\text{ared}_k - \text{cred}_k| \leq K\theta_k \quad \forall k\,. \tag{10a}$$

For a fixed $\omega \in (0, 1)$, we control the error estimator θ_k via the following bound:

$$\theta_k^{\omega} \leq \eta \min\left\{\text{pred}_k, r_k\right\}, \tag{10b}$$

where

$$\eta < \min\{\eta_1, 1 - \eta_2\} \quad \text{and} \quad \{r_k\}_{k=1}^{\infty} \subset [0, \infty) \text{ satisfies } \lim_{k\to\infty} r_k = 0. \tag{10c}$$

Condition (10b) is due to Ziems and Ulbrich [43]; also see [13, Sec. 10.6]. □

The basic trust-region algorithm, accounting for inexact objective function and gradient evaluation, is listed as Algorithm 1.

To prove convergence of the inexact unconstrained trust-region algorithm, we will use some of the problem assumptions stated in Section 3 and the following model assumptions:

(R1) For each k, $m_k : Z \to \mathbb{R}$ is twice continuously Fréchet differentiable.
(R2) For each k, $\nabla^2 m_k$ is uniformly bounded on Z.
(R3) For each k, the objective function approximation $\mathcal{J}_k$ is bounded below.

Algorithm 1 (Reduced-space trust-region algorithm)

Initialization: *Choose initial point* z_0, *initial trust-region radius* $\Delta_0 > 0$, *constants* $0 < \gamma_1 \leq \gamma_2 < 1 < \gamma_3$, $0 < \eta_1 < \eta_2 < 1$, *and* $tol > 0$.

For *k=0,1,2,...*

1. ***Model selection:*** *Choose a model* m_k *that satisfies* (8).
2. ***Convergence check:*** *If* $\|\nabla m_k(0)\|_Z < tol$, *then terminate.*
3. ***Step computation:*** *Compute an approximate solution* s_k *of* (6) *that satisfies the fraction of Cauchy decrease condition* (7).
4. ***Objective function update:*** *Determine an objective function approximation* J_k *such that the corresponding error estimate* θ_k *satisfies* (10).
5. ***Step acceptance:*** *Compute* $\varrho_k = \text{cred}_k/\text{pred}_k$.
   if $\varrho_k \geq \eta_1$ ***then*** $z_{k+1} = z_k + s_k$ ***else*** $z_{k+1} = z_k$
6. ***Trust-region update:***
   if $z_{k+1} = z_k$ ***then*** $\Delta_{k+1} \in (0, \gamma_1 \|s_k\|_Z]$
   else

$$\Delta_{k+1} \in \begin{cases} (0, \gamma_2 \|s_k\|_Z] & \textbf{if } \rho_k \leq \eta_1 \\ [\gamma_2, \|s_k\|_Z, \Delta_k] & \textbf{if } \rho_k \in (\eta_1, \eta_2) \\ [\Delta_k, \gamma_3 \Delta_k] & \textbf{if } \rho_k \geq \eta_2 \end{cases}$$

End For

Theorem 1 *Let* $\Omega = U \times Z$. *If problem assumptions (A1), (A2), (A6), (A7) and (A8), and model assumptions (R1), (R2) and (R2) hold, then the iterates* $\{z_k\}$ *generated by the inexact unconstrained trust-region algorithm, Algorithm 1, satisfy*

$$\liminf_{k\to\infty} \|\nabla m_k(0)\|_Z = \liminf_{k\to\infty} \|\nabla \mathcal{J}(z_k)\|_Z = 0.$$

Proof Assumptions (A2), (A6), and (A7) together with the Implicit Function Theorem [22, Th. 1.41] ensure that $\mathcal{J}$ is twice continuously Fréchet differentiable. Moreover, (A8) ensures that the Hessian of $\mathcal{J}$ is uniformly bounded at $z_k + ts_k$ for all $t \in [0, 1]$ and for all k. The desired result then follows from assumptions (R1), (R2), and (R3), and a slight generalization of Theorem 5.6 in [26].

4.1.1 Related Work

The above approach is described in detail in [25, 26]. The authors in [25, 26] combine and modify conditions for inexact gradient and objective function values from a number of sources. For example, Moré considers inexact gradients in [29] for the case of $Z = \mathbb{R}^n$. In this case, he requires

$$z_k \to z \quad \Longrightarrow \quad \lim_{k\to\infty} \|\nabla m_k(0) - \nabla \mathcal{J}(z_k)\|_Z = 0. \tag{11}$$

Similarly, in [40] Toint analyzes an algorithm in Hilbert space for bound-constrained optimization and requires

$$\|\nabla m_k(0) - \nabla \mathcal{J}(z_k)\|_Z \leq \min\{\kappa_1, \kappa_2 \Delta_k\}$$

for appropriately chosen $\kappa_1,\ \kappa_2 > 0$. Carter, in [10], proves global convergence of Algorithm 1 using the inexactness conditions

$$|\text{ared}_k - \text{cred}_k| \leq \zeta_{f,1} \text{pred}_k$$
$$|\text{ared}_k - \text{cred}_k| \leq \zeta_{f,2} |\text{cred}_k|$$
$$\langle \nabla m_k(0) - \nabla \mathcal{J}(z_k), \nabla m_k(0) \rangle_Z \leq \zeta_g \|\nabla m_k(0)\|_Z^2$$

for constants $\zeta_g,\ \zeta_{f,1}\ \zeta_{f,2} > 0$ satisfying

$$\zeta_g + \zeta_{f,1} < 1 - \eta_2 \quad \text{and} \quad \zeta_{f,2} < 1.$$

Carter further analyzes his approach in [11, 12]. In [13, Sec. 10.6], the authors consider objective functions with dynamic accuracy for which they require

$$\max\{|\mathcal{J}(z_k) - \mathcal{J}_k(z_k)|, |\mathcal{J}(z_k + s_k) - \mathcal{J}_k(z_k + s_k)|\} \leq \tilde{\eta}\ \text{pred}_k\,, \tag{12}$$

for some $\tilde{\eta} \leq \frac{1}{2}\eta_1$. The challenge with each of these conditions (other than Moré's very general requirement (11)) is that the constants, e.g., ζ_g, $\zeta_{f,1}$, $\zeta_{f,2}$ and $\tilde{\eta}$, depend explicitly on algorithmic parameters. Therefore, it is difficult to determine a priori if these conditions can be satisfied in practice. As a practical alternative, the inexact gradient conditions of Toint and Carter are combined by Heinkenschloss and Vicente in [21], giving rise to the condition (8), which permits an arbitrary constant scaling κ_{grad} on the error bound and enables easy implementation. In a similar vein, Ulbrich and Ziems in [43] relax the dependence of the inexact objective function condition (12) on algorithmic parameters, motivating the more practical conditions (10).

Kelley and Sachs in [24] take a different approach to that presented here. They work in the setting where the value and gradient approximations are provided by "black-box" calculations that satisfy controllable absolute and relative error tolerances. They suggest modifications to the basic trust-region algorithm so that the resulting algorithm performs as if there were no errors in the computation of the value and gradient.

The authors of [25, 26] apply Algorithm 1 to PDE-constrained optimization problems for which the governing PDE has uncertain coefficients. The inexactness conditions (8) and (10) are used to adaptively refine sparse-grid quadrature approximations of the objective function. In [5, 18], the authors employ the inexact gradient condition (8) to adaptively refine reduced-order models of the PDE constraint using proper orthogonal decomposition (POD). However, they evaluate the discretized

objective function exactly since the state equation must be solved to build the POD model of the state and adjoint variables. Similarly, the authors of [8] use inexact gradients to adaptively refine Monte Carlo sample sizes for mixed logit optimization.

4.2 A Full-Space Approach

In this section, we focus on the full-space formulation (3) of PDE-constrained optimization problems. We review a sequential quadratic programming (SQP) approach to solving (3), originating in the composite-step trust-region scheme of Byrd and Omojokun [32]. We assume that inexactness in solving (3) is due to the approximate solution of a variety of subproblems, such as quadratic optimization problems or linear systems, which comprise the SQP scheme. The handling of inexactness is based on [20].

Remark 2 To further simplify the presentation, in this section we assume that the constraint space C is a Hilbert space. As a reminder, we identify its dual space C^* with C. Finally, we note that a treatment of more general constraint spaces such as reflexive Banach spaces is possible in the context of full-space composite-step methods, see, e.g., [28].

Recall $X = U \times Z$, and write the Lagrangian functional $L : X \times C \to \mathbb{R}$ for (3),

$$L(x, \lambda) = J(x) + \langle \lambda, c(x) \rangle_C .$$

We let x_k be the k-th SQP iterate, λ_k the Lagrange multiplier estimate at x_k, and $B_k = B(x_k, \lambda_k)$ the Hessian $\nabla_{xx} L(x_k, \lambda_k)$ of the Lagrangian or a self-adjoint approximation thereof. Trust-region SQP methods compute an approximate solution of (3) by approximately solving a sequence of subproblems derived from

$$\min_s \quad \frac{1}{2} \langle B_k s, s \rangle_X + \langle \nabla_x L(x_k, \lambda_k), s \rangle_X + L(x_k, \lambda_k) \tag{13a}$$

$$\text{subject to} \quad c_x(x_k)s + c(x_k) = 0 \tag{13b}$$

$$\|s\|_X \leq \Delta_k. \tag{13c}$$

To deal with the possible incompatibility of the constraints (13b), (13c), we apply a composite-step approach, where the trial step s_k is computed as the sum of a quasi-normal step n_k and a tangential step t_k. The role of the quasi-normal step n_k is to reduce linear infeasibility. It is computed as an approximate solution of

$$\min_n \quad \|c_x(x_k)n + c(x_k)\|_C^2 \tag{14a}$$

$$\text{subject to} \quad \|n\|_X \leq \zeta \Delta_k, \tag{14b}$$

where $\zeta \in (0, 1)$ is a fixed constant. To ensure global convergence of the SQP algorithm, the quasi-normal step must satisfy two conditions.

Quasi-Normal Step Conditions The quasi-normal step, n_k, must satisfy the boundedness condition

$$\|n_k\|_X \leq \kappa_1 \|c(x_k)\|_C, \tag{15}$$

where $\kappa_1 > 0$ is independent of k, and the fraction of Cauchy decrease condition

$$\|c(x_k)\|_C^2 - \|c_x(x_k)n_k + c(x_k)\|_C^2 \geq \kappa_2 \|c(x_k)\|_C \min\{\kappa_3 \|c(x_k)\|_C, \Delta_k\}, \tag{16}$$

where $\kappa_2, \kappa_3 > 0$ are independent of k. These conditions on the quasi-normal step are derived by Dennis, El-Alem, and Maciel in [14]. They are adopted by Heinkenschloss and Vicente in [21]. A more restrictive version, requiring κ_2 and κ_3 to be in the interval $(0, 1)$, is used by Ziems and Ulbrich in [43]. □

To understand the computation of the tangential step, we consider subproblem (13), where we substitute the computed quasi-normal step, i.e., $s = t + n_k$:

$$\min_t \quad \frac{1}{2}\langle B_k(t + n_k), t + n_k\rangle_X + \langle \nabla_x L(x_k, \lambda_k), t + n_k\rangle_X \tag{17a}$$

$$\text{subject to} \quad c_x(x_k)t = 0 \tag{17b}$$

$$\|t + n_k\|_X \leq \Delta_k. \tag{17c}$$

To solve (17), one typically eliminates the constraints (17b) using a representation of the null space of $c_x(x_k)$. Several null-space representations can be considered in the development of algorithms. Let E be a Hilbert space and let $W_k : E \to X$ be a bounded linear operator such that

$$\text{Range}(W_k) = \text{Null}(c_x(x_k)).$$

For instance, under the assumption (A6) and recalling the notation from Section 3, one can use $E = Z$ and define

$$W_k = T(x_k) = \begin{pmatrix} -c_u(x_k)^{-1} c_z(x_k) \\ I_Z \end{pmatrix}.$$

This null-space representation is a natural choice for many PDE-constrained optimization problems. In the context of inexact trust-region methods, it is used by Heinkenschloss and Vicente [21] and Ziems and Ulbrich [43]. Specifically, the authors in [21, 43] study inexact applications of the operator $c_u(x_k)^{-1}$. A more general alternative, not requiring assumption (A6), i.e., the invertibility of $c_u(x_k)$, is to choose $E = X$ and $W_k = W_k^* = W_k^2$, and to compute $t = W_k w$ by solving the so-called *augmented system*

$$\begin{pmatrix} I_X & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} t \\ z \end{pmatrix} = \begin{pmatrix} w \\ 0 \end{pmatrix}. \tag{18}$$

We note that in either case we can set $t = W_k w$ and replace (17) by

$$\min_w \quad \frac{1}{2} \langle W_k^* B_k W_k w, w \rangle_X + \langle W_k^* g_k, w \rangle_X \tag{19a}$$

$$\text{subject to} \quad \|n_k + W_k w\|_X \leq \Delta_k, \tag{19b}$$

where $g_k = \nabla_x L(x_k, \lambda_k) + B_k n_k$. A potential benefit of the second null-space representation, (18), is that the linear system can be solved using modern iterative saddle-point and Karush-Kuhn-Tucker (KKT) system solvers, see, e.g., [9, 33]. Moreover, these methods can be used to solve not only (18), but also the full KKT system, i.e., the system where I_X in (18) is replaced by B_k, thereby circumventing the need for preconditioning of the operator $W_k^* B_k W_k$, which is typically very challenging.

For the remainder of the chapter, we focus on the second null-space representation. A key source of inexactness in the application of the null-space operator W_k is the iterative solution of the linear system (18), using Krylov methods. In this case, the vector $W_k^* g_k$ and the operator $W_k^* B_k W_k$ are no longer available exactly. It is shown in [20] that the inexact solution of linear systems like (18) leads to an approximation $\widetilde{W}_k$ of W_k. While the operator W_k is self-adjoint (that is, assuming exact linear system solves), the approximation $\widetilde{W}_k$ is in general not self-adjoint and may not even be linear. Nonetheless, it is also shown, [20, p. 1525], that there exists a fixed linear operator that replicates the action of $\widetilde{W}_k$ on the set of vectors involved in the algorithm for solving (19). Therefore, an algorithm that computes a solution w_k of (19) with inexact applications of the operator W_k also solves

$$\min_w \quad \frac{1}{2} \langle \widetilde{W}_k^* B_k \widetilde{W}_k w, w \rangle_X + \langle \widetilde{W}_k^* \widetilde{W}_k g_k, w \rangle_X \tag{20a}$$

$$\text{subject to} \quad \|n_k + \widetilde{W}_k w\|_X \leq \Delta_k \tag{20b}$$

for some fixed linear operator $\widetilde{W}_k$. We note that the vector $\widetilde{W}_k^* \widetilde{W}_k g_k$ above replaces the vector $\widetilde{W}_k^* g_k$. This modification is needed for the convergence proof.[1] Problem (20) is equivalent to

$$\min_{\widetilde{t}} \quad \frac{1}{2} \langle B_k \widetilde{t}, \widetilde{t} \rangle_X + \langle \widetilde{W}_k g_k, \widetilde{t} \rangle_X \tag{21a}$$

$$\text{subject to} \quad \widetilde{t} \in \text{Range}(\widetilde{W}_k) \tag{21b}$$

$$\|n_k + \widetilde{t}\|_X \leq \Delta_k. \tag{21c}$$

[1] Clearly, with exact linear system solves we have $W_k^* W_k g_k = W_k W_k g_k = W_k g_k$, however, in the presence of inexactness the distinction is important.

We call (21) the *tangential subproblem*. Solving (21) is the first stage of computing the tangential step. To guarantee global convergence of the SQP algorithm, the approximate solution $\widetilde{t}_k$ of (21) must satisfy the following conditions.

Tangential Subproblem Conditions First, the quantity $\widetilde{W}_k g_k$ needs to satisfy

$$\|\widetilde{W}_k g_k - W_k g_k\|_X \leq \tau_1 \min\left\{\|\widetilde{W}_k g_k\|_X, \Delta_k\right\}, \tag{22}$$

for some $\tau_1 > 0$ independent of k. Second, we define the inexact quadratic model

$$\widetilde{q}_k(t) := \frac{1}{2}\langle B_k t, t\rangle_X + \left\langle \widetilde{W}_k g_k, t\right\rangle_X,$$

and impose the fraction of Cauchy decrease condition on $\widetilde{t}_k$ as follows,

$$\widetilde{q}_k(0) - \widetilde{q}_k(\widetilde{t}_k) \geq \kappa_4 \|\widetilde{W}_k g_k\|_X \min\left\{\kappa_5 \|\widetilde{W}_k g_k\|_X, \kappa_6 \Delta_k\right\}, \tag{23}$$

for $\kappa_4, \kappa_5, \kappa_6 > 0$, independent of k. Condition (23) is analogous to the fraction of Cauchy decrease condition (7) in the reduced-space setting. It is an extension of a condition discussed by Dennis, El-Alem, and Maciel in [14]; in our context, the "inexact" quantity $\widetilde{W}_k g_k$ is introduced and the quadratic model is defined according to the discussion in the previous paragraph. Condition (22) is derived from a similar condition by Heinkenschloss and Vicente [21]. It is analogous to the inexact gradient condition (8) in the reduced-space setting. Establishing the existence of $\widetilde{W}_k$ that is compatible with (22) and (23) is an important challenge, discussed in [20]. □

With inexactness $\widetilde{t}_k = \widetilde{W}_k w_k$ is no longer in the null space of $c_x(x_k)$, and may destroy some of the linear feasibility gained by the quasi-normal step n_k. To compensate for this, we compute the tangential step t_k from $\widetilde{t}_k$ to restore linear feasibility as needed. This computation is intimately tied to the global convergence mechanisms used in SQP methods. Once the trial step $s_k = n_k + t_k$ is computed, one must decide whether to accept the step and how to update the trust-region radius Δ_k. To perform these tasks, we use the *augmented Lagrangian merit function*

$$\phi(x, \lambda; \rho) = J(x) + \langle \lambda, c(x)\rangle_C + \rho \|c(x)\|_C^2 = L(x, \lambda) + \rho \|c(x)\|_C^2. \tag{24}$$

In a conventional trust-region SQP algorithm, the step s_k is accepted or rejected and the trust-region radius Δ_k is updated based on the ratio between the actual reduction

$$\text{ared}(s_k; \rho_k) = \phi(x_k, \lambda_k; \rho_k) - \phi(x_k + s_k, \lambda_{k+1}; \rho_k) \tag{25}$$

and the predicted reduction

$$\widehat{\text{pred}}(s_k; \rho_k) = \phi(x_k, \lambda_k; \rho_k) - \Big[L(x_k, \lambda_k) + \langle \nabla_x L(x_k, \lambda_k), s_k\rangle_X + \frac{1}{2}\langle B_k s_k, s_k\rangle_X + \langle \lambda_{k+1} - \lambda_k, c_x(x_k)s_k + c(x_k)\rangle_C + \rho_k \|c_x(x_k)s_k + c(x_k)\|_C^2\Big]. \tag{26}$$

Here λ_{k+1} is a Lagrange multiplier estimate corresponding to the trial iterate $x_k + s_k$. To account for the inexactness in the constraint null space projection, the definition of the predicted reduction must be modified. Using

$$r_k^t = c_x(x_k)t_k \,,$$

which can be interpreted as an indicator of the loss of linear feasibility, the predicted reduction (26) is redefined,

$$\widehat{\text{pred}}(s_k;\, \rho_k) := \text{pred}(n_k, \widetilde{t}_k;\, \rho_k) + \text{rpred}(r_k^t;\, \rho_k),$$

with the following components:

$$\begin{aligned}
&\text{pred}(n_k, \widetilde{t}_k;\, \rho_k) \\
&= -\left\langle \widetilde{W}_k g_k, \widetilde{t}_k \right\rangle_X - \frac{1}{2}\left\langle B_k \widetilde{t}_k, \widetilde{t}_k \right\rangle_X - \left\langle \nabla_x L(x_k, \lambda_k), n_k \right\rangle_X - \frac{1}{2}\left\langle B_k n_k, n_k \right\rangle_X \\
&\quad - \left\langle \lambda_{k+1} - \lambda_k, c_x(x_k)n_k + c(x_k) \right\rangle_C \\
&\quad + \rho_k \left(\|c(x_k)\|_C^2 - \|c_x(x_k)n_k + c(x_k)\|_C^2 \right)
\end{aligned} \tag{27}$$

and

$$\text{rpred}(r_k^t;\, \rho_k) = -\left\langle \lambda_{k+1} - \lambda_k, r_k^t \right\rangle_C - \rho_k \|r_k^t\|_C^2 - 2\rho_k \left\langle r_k^t, c_x(x_k)n_k + c(x_k) \right\rangle_C . \tag{28}$$

The splitting of the predicted reduction into a term that only involves $\widetilde{t}_k$ and a term that only involves t_k is discussed in [20, p. 1514–1515]; it is partially motivated by the arguments made in [21, p. 292].[2] In our algorithm, we first compute a penalty parameter ρ_k satisfying

$$\text{pred}(n_k, \widetilde{t}_k;\, \rho_k) \geq \frac{\rho_k}{2}\left(\|c(x_k)\|_C^2 - \|c_x(x_k)n_k + c(x_k)\|_C^2 \right),$$

and then "postprocess" $\widetilde{t}_k$ to compute a tangential step t_k that satisfies the conditions stated below. The postprocessing can be performed by applying another null-space projection, similar to solving (18).

Tangential Step Conditions The tangential step t_k must satisfy the requirement

$$|\text{rpred}(r_k^t;\, \rho_k)| \leq \eta_0\, \text{pred}(n_k, \widetilde{t}_k;\, \rho_k), \tag{29}$$

where $\eta_0 \in (0,\, 1 - \eta_1)$, and $\eta_1 \in (0,\, 1)$ is the smallest acceptable ratio of the actual and predicted reduction. Additionally, to control how much the tangential step t_k can deviate from the projection $W_k\widetilde{t}_k$ of $\widetilde{t}_k$ we require

[2] For all details, see the proofs in [20, p. 1536–1538] and [21, p. 295–298].

$$\|t_k - W_k\widetilde{t}_k\|_X \leq \tau_3 \Delta_k \min\{\Delta_k, \|s_k\|_X\}, \tag{30}$$

for some $\tau_3 > 0$ independent of k. Finally, we impose the boundedness condition

$$\|\widetilde{t}_k\|_X \leq \tau_4 \|s_k\|_X, \tag{31}$$

for $\tau_4 > 0$ independent of k. These conditions are discussed in [20]. □

So far, we discussed general conditions needed for the global convergence of a composite-step trust-region SQP scheme where the null-space operator W_k is applied inexactly. Following [20], we now present a concrete instance of the algorithm, specifically designed to robustly and efficiently handle inexactness in the iterative solution of linear systems comprising the application of W_k. This algorithm is useful whenever iterative solvers, such as Krylov methods, are applied to solve linear systems based on discretizations of operators c_u and c_u^*, i.e., the state Jacobians and their adjoints. In PDE-constrained optimization, the matrices resulting from, e.g., finite element discretizations of c_u and c_u^* are often very large, prohibiting direct computation of matrix inverses and matrix factorizations. Additionally, in, e.g., optimization under uncertainty, the computational challenge is exacerbated by the dependence of the constraint equation on the random input vector ξ, resulting in enormous linear systems that cannot be formed explicitly. This prompts the need for matrix-free methods, where the action of a linear operator on a vector is specified, rather than the operator (matrix) itself.

We will formulate concrete subalgorithms for the quasi-normal step computation, the solution of the tangential subproblem, the tangential step computation, and the Lagrange multiplier update that satisfy the previously discussed conditions. After specifying the subalgorithms, we state the master algorithm in Section 4.2.5. The solution of augmented systems, which are key components of the subalgorithms, is discussed in Section 5.2, in the context of PDE-constrained optimization under uncertainty.

4.2.1 Computation of the Quasi-Normal Step

An approximate solution of (14) can be computed using the dogleg method. Let n_k^{cp} be the solution of $\min\left\{\|c_x(x_k)n + c(x_k)\|_C^2 \ : \ n = -\alpha c_x(x_k)^* c(x_k), \alpha \geq 0\right\}$, also known as the *Cauchy point*. It is easy to verify that

$$n_k^{cp} = -\frac{\|c_x(x_k)^* c(x_k)\|_X^2}{\|c_x(x_k)c_x(x_k)^* c(x_k)\|_C^2} c_x(x_k)^* c(x_k). \tag{32}$$

If $\|n_k^{cp}\|_X \geq \zeta \Delta_k$, then we set the quasi-normal step to $n_k = \zeta \Delta_k n_k^{cp} / \|n_k^{cp}\|_X$.

If $\|n_k^{cp}\|_X < \zeta \Delta_k$, to accelerate convergence we take a step toward an approximate minimum-norm solution n_k^N of $\min \|c_x(x_k)n + c(x_k)\|_C^2$, sometimes called the *Newton point*. The quasi-normal step is then computed by moving from

n_k^{cp} as far as possible toward n_k^N while staying within the trust region with radius $\zeta \Delta_k$. Specifically, we solve for $\delta n_k = n_k^N - n_k^{cp}$ the augmented system

Algorithm 2 (Dogleg method for the quasi-normal subproblem)

Initialization: Choose $0 < \zeta, \tau^{qn} < 1$.

1. Compute n_k^{cp} as defined in (32).
2. If $\|n_k^{cp}\|_X \geq \zeta \Delta_k$, then set $n_k = \zeta \Delta_k n_k^{cp} / \|n_k^{cp}\|_X$;
   Else compute δn_k via (33) so that e^1 and e^2 satisfy (34).
3. If $\|n_k^{cp} + \delta n_k\|_X \leq \zeta \Delta_k$, then set $n_k = n_k^{cp} + \delta n_k = n_k^N$;
   Else compute $\theta_k \in (0, 1)$ such that $\|n_k^{cp} + \theta_k \delta n_k\|_X = \zeta \Delta_k$, and set $n_k = n_k^{cp} + \theta_k \delta n_k$.

$$\begin{pmatrix} I_X & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} \delta n_k \\ y \end{pmatrix} = \begin{pmatrix} -n_k^{cp} + e^1 \\ -c_x(x_k) n_k^{cp} - c(x_k) + e^2 \end{pmatrix}. \tag{33}$$

To comply with the convergence conditions (15) and (16), the size of the residual $\begin{pmatrix} e^1 & e^2 \end{pmatrix} \in X \times C$ is restricted via

$$\|e^1\|_X^2 + \|e^2\|_C^2 \leq \left(\tau^{qn}\right)^2 \|c_x(x_k) n_k^{cp} + c(x_k)\|_C^2 , \tag{34}$$

where $0 < \tau^{qn} \leq 1$. In summary, the algorithm for computing the quasi-normal step is given as follows:

Additional algorithms that satisfy conditions (15) and (16) are discussed in [21, p. 298–299].

4.2.2 Solution of the Tangential Subproblem

The tangential subproblem (21) is solved using a modified truncated Steihaug-Toint conjugate gradient (STCG) method. Aside from handling the nonstandard objective function in (21), the modifications involve a full orthogonalization of search directions, several important tunings of the classical STCG truncation criteria and the related exit computations, and a special termination condition related to an estimate of the accumulated error in the constraint null-space. The latter is discussed next.

The algorithm for the solution of the tangential subproblem (21), Algorithm 3, repeatedly applies an inexact null-space projector $\widetilde{W}_k$ by iteratively solving augmented systems of type (18). We note that $\widetilde{W}_k$ is not explicitly available; only the results of its action on the vector g_k and its action on the STCG residuals $\widetilde{r}_i$ used in

Algorithm 3 are known.[3] We introduce the operator $R_i : \mathbb{R}^{i+1} \to X$, given by

$$R_i = [g_k, \widetilde{r}_1, \ldots, \widetilde{r}_i],$$

Algorithm 3 (STCG method with inexact null-space projections)

Initialization: Given relative tolerance $tol^{CG} \in (0, 1)$. Given iteration maximum $i_{\max}^{CG}$. Choose $0 < \tau^{pg}, \tau^{proj} \leq 1$. Let $\widetilde{t}_{k,0} = 0 \in X$. Compute $\widetilde{r}_0 = \widetilde{W}_k g_k$ by solving (36) with the linear solver tolerance (37). If $\|\widetilde{r}_0\|_X = 0$, **stop**.

For $i = 0, 1, 2, \ldots, i_{\max}^{CG}$

1. If $i = 0$ set $\widetilde{z}_0 = \widetilde{r}_0$; Else compute $\widetilde{z}_i = \widetilde{W}_k \widetilde{r}_i$ via (38) with the linear solver tolerance (39).
   If $\|\widetilde{z}_i\|_X \leq tol^{CG} \|\widetilde{r}_0\|_X$ and $i > 0$, return $\widetilde{t}_k = \widetilde{t}_{k,i}$ and $\widetilde{t}_k^{cp} = \widetilde{t}_{k,1}$, and **stop**.
2. Compute $\widehat{S}_i$ defined in (35). If $\|\widehat{S}_i\|_2 > 1/2$, return $\widetilde{t}_k = \widetilde{t}_{k,i}$ and $\widetilde{t}_k^{cp} = \widetilde{t}_{k,1}$, and **stop**.
3. Set $\widetilde{p}_i = -\widetilde{z}_i + \sum_{j=0}^{i-1} \frac{\langle \widetilde{z}_i, H_k \widetilde{p}_j \rangle_X}{\langle \widetilde{p}_j, H_k \widetilde{p}_j \rangle_X} \widetilde{p}_j$.
4. If $\langle \widetilde{r}_i, \widetilde{p}_i \rangle_X \neq 0$ and $\langle \widetilde{p}_i, H \widetilde{p}_i \rangle_X \leq 0$, compute θ such that $\text{sign}(\theta) = \text{sign}(-\langle \widetilde{r}_i, \widetilde{p}_i \rangle_X)$ and $\|n_k + \widetilde{t}_{k,i} + \theta \widetilde{p}_i\|_X = \Delta_k$, and return $\widetilde{t}_k = \widetilde{t}_{k,i+1} = \widetilde{t}_{k,i} + \theta \widetilde{p}_i$ and $\widetilde{t}_k^{cp} = \widetilde{t}_{k,1}$, and **stop**.
   If $\langle \widetilde{r}_i, \widetilde{p}_i \rangle_X = 0$ and $\langle \widetilde{p}_i, H \widetilde{p}_i \rangle_X < 0$, compute θ such that $\|n_k + \widetilde{t}_{k,i} + \theta \widetilde{p}_i\| = \Delta_k$, and return $\widetilde{t}_k = \widetilde{t}_{k,i+1} = \widetilde{t}_{k,i} + \theta \widetilde{p}_i$ and $\widetilde{t}_k^{cp} = \widetilde{t}_{k,1}$, and **stop**.
5. If $\langle \widetilde{r}_i, \widetilde{p}_i \rangle_X = 0$, return $\widetilde{t}_k = \widetilde{t}_{k,i}$ and $\widetilde{t}_k^{cp} = \widetilde{t}_{k,1}$, and **stop**.
6. Set $\widetilde{\alpha}_i = -\frac{\langle \widetilde{r}_i, \widetilde{p}_i \rangle_X}{\langle \widetilde{p}_i, H_k \widetilde{p}_i \rangle_X}$.
7. Set $\widetilde{t}_{k,i+1} = \widetilde{t}_{k,i} + \widetilde{\alpha}_i \widetilde{p}_i$.
8. If $\|n_k + \widetilde{t}_{k,i+1}\|_X \geq \Delta_k$, compute θ such that $\text{sign}(\theta) = \text{sign}(\widetilde{\alpha}_i)$ and $\|n_k + \widetilde{t}_{k,i} + \theta \widetilde{p}_i\|_X = \Delta_k$, and return $\widetilde{t}_k = \widetilde{t}_{k,i+1} = \widetilde{t}_{k,i} + \theta \widetilde{p}_i$ and $\widetilde{t}_k^{cp} = \widetilde{t}_{k,1}$, and **stop**.
9. Set $\widetilde{r}_{i+1} = \widetilde{r}_i + \widetilde{\alpha}_i H_k \widetilde{p}_i$.

End For

the operator $\widetilde{Y}_i : \mathbb{R}^{i+1} \to X$, given by

$$\widetilde{Y}_i = [\widetilde{W}_k g_k, \widetilde{W}_k \widetilde{r}_1, \ldots, \widetilde{W}_k \widetilde{r}_i],$$

and the diagonal matrix

$$D_i = \text{diag}(\|\widetilde{W}_k g_k\|_X, \|\widetilde{W}_k \widetilde{r}_1\|_X, \ldots, \|\widetilde{W}_k \widetilde{r}_i\|_X).$$

[3] Only the scope of the index k extends from Algorithm 4 to Algorithm 3 – indices i and j are independent, i.e., their scope is local to each algorithm.

Finally, we define the matrix

$$\widehat{S}_i = D_i^{-1}(\widetilde{Y}_i^T R_i - D_i^2)D_i^{-1}\,. \tag{35}$$

In [20] it is shown that $\|\widehat{S}_i\|_2$ can be used to control the cumulative effect of inexactness in the projections $\widetilde{W}_k$. The modified STCG algorithm is specified next.

The augmented system residuals related to the application of the inexact projector $\widetilde{W}_k$ are controlled as follows. In Step 3 of Algorithm 3, the inexact projected gradient $\widetilde{r}_0 = \widetilde{W}_k g_k$ is computed. The iterative linear system solver returns $\widetilde{r}_0$ satisfying

$$\begin{pmatrix} I_X & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} \widetilde{r}_0 \\ y \end{pmatrix} = \begin{pmatrix} g_k \\ 0 \end{pmatrix} + \begin{pmatrix} e^1 \\ e^2 \end{pmatrix}\,. \tag{36}$$

The residual $\begin{pmatrix} e^1 & e^2 \end{pmatrix} \in X \times C$ must satisfy

$$\|e^1\|_X + \|e^2\|_C \leq \tau^{pg} \min\{\|\widetilde{r}_0\|_X, \Delta_k, \|g_k\|_X\}\,, \tag{37}$$

where $0 < \tau^{pg} \leq 1$. In Step 1 in Algorithm 3, we compute $\widetilde{z}_i = \widetilde{W}_k \widetilde{r}_i$. The iterative linear system solver returns $\widetilde{z}_i$ satisfying

$$\begin{pmatrix} I_X & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} \widetilde{z}_i \\ y \end{pmatrix} = \begin{pmatrix} \widetilde{r}_i \\ 0 \end{pmatrix} + \begin{pmatrix} e_i^1 \\ e_i^2 \end{pmatrix}, \tag{38}$$

where the residual $\begin{pmatrix} e_i^1 & e_i^2 \end{pmatrix} \in X \times C$ is controlled by the condition

$$\|e_i^1\|_X + \|e_i^2\|_C \leq \tau^{proj} \min\{\|\widetilde{z}_i\|_X, \|\widetilde{r}_i\|_X\}\,, \tag{39}$$

with $0 < \tau^{proj} \leq 1$.

4.2.3 Computation of the Tangential Step

Once the approximate solution $\widetilde{t}_k$ of the tangential subproblem (21) has been obtained, the tangential step t_k is computed. The goal is to restore some of the linear feasibility lost in Algorithm 3. To this end, another inexact null space projection is performed,

$$\begin{pmatrix} I_X & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix} \begin{pmatrix} t_k \\ y \end{pmatrix} = \begin{pmatrix} \widetilde{t}_k \\ 0 \end{pmatrix} + \begin{pmatrix} e^1 \\ e^2 \end{pmatrix}, \tag{40}$$

where the residual $\begin{pmatrix} e^1 & e^2 \end{pmatrix} \in X \times C$ must satisfy

$$\|e^1\|_X + \|e^2\|_C \leq \Delta_k \min\left\{\Delta_k, \|n_k + t_k\|_X, \tau^{tang}\|\widetilde{t}_k\|_X/\Delta_k\right\}\,, \tag{41}$$

for $0 < \tau^{tang} \leq 1$.

4.2.4 Computation of the Lagrange Multipliers

For global convergence, we require only that the sequence of Lagrange multipliers be bounded. For fast convergence, we may compute the Lagrange multiplier estimate λ_{k+1} by approximately minimizing $\|\nabla J(\widehat{x}_k) + c_x(\widehat{x}_k)^*\lambda\|_X$, where $\widehat{x}_k = x_k + n_k + t_k$. Specifically, we solve for $\Delta\lambda = \lambda_{k+1} - \lambda_k$, where λ_k is the previous Lagrange multiplier estimate, as follows:

$$\begin{pmatrix} I_X & c_x(\widehat{x}_k)^* \\ c_x(\widehat{x}_k) & 0 \end{pmatrix} \begin{pmatrix} z \\ \Delta\lambda \end{pmatrix} = \begin{pmatrix} -\nabla J(\widehat{x}_k) - c_x(\widehat{x}_k)^*\lambda_k + e^1 \\ e^2 \end{pmatrix}. \tag{42}$$

The residual $\begin{pmatrix} e^1 & e^2 \end{pmatrix} \in X \times C$ must satisfy

$$\|e^1\|_X + \|e^2\|_C \leq \min\left\{\tau^{lmg},\ \tau^{lmh}\|\nabla J(\widehat{x}_k) + c_x(\widehat{x}_k)^*\lambda_k\|_X\right\}, \tag{43}$$

for $0 < \tau^{lmh} \leq 1$ and a fixed $\tau^{lmg} > 0$ independent of k. Here τ^{lmh} governs the relative size of the linear system residual, while τ^{lmg} is used to enforce boundedness of the multipliers. Clearly, there are many other ways to compute Lagrange multipliers satisfying the boundedness condition.

4.2.5 Full-Space Trust-Region SQP Algorithm with Inexact Linear System Solves

Here we state the complete full-space trust-region SQP algorithm with inexact linear system solves.

To prove convergence of the inexact full-space trust-region algorithm, we use some of the problem assumptions from Section 3 and the following algorithmic assumptions:

(F1) The sequence $\{\lambda_k\}_{k\in\mathbb{N}}$ is bounded.
(F2) The sequence of operators $\{B_k\}_{k\in\mathbb{N}}$ is bounded.
(F3) For each k, the projection $W_k : X \to X$ onto $\mathrm{Null}(c_x(x_k))$ satisfies $\|W_k\|_{\mathcal{L}(X)} = 1$.

Theorem 2 *Let $\Omega = X = U \times Z$. If problem assumptions (A2), (A3), (A4), (A5), and (A9), and algorithmic assumptions (F1), (F2), and (F3) are satisfied, then the sequences of iterates generated by Algorithm 4 satisfy*

$$\liminf_{k\to\infty}\{\|\widetilde{W}_k g_k\|_X + \|c(x_k)\|_C\} = 0. \tag{44}$$

Additionally, we have

$$\liminf_{k\to\infty}\{\|W_k\nabla_x J(x_k)\|_X + \|c(x_k)\|_C\} = 0. \tag{45}$$

Proof In [20] it is shown that Algorithm 4 is a specific instance of a more general composite-step trust-region SQP algorithm, [20, Algorithm 3.3]. Under the given problem assumptions and algorithmic assumptions, the global convergence result follows directly from [20, Theorem 3.5].

Algorithm 4 (Trust-region SQP algorithm with inexact linear system solves)
Initialization: Choose initial point x_0, initial trust-region radius Δ_0, constants $0 < \alpha_1, \eta_1 < 1$, $0 < \eta_0 < 1 - \eta_1$, $\rho_{-1} \geq 1$, $\bar{\rho} > 0$, and $tol^{SQP} > 0$. Set $\Delta_{min}, \Delta_{max}$ so that $0 < \Delta_{min} < \Delta_{max}$. Set forcing parameters $\tau^{qn}, \tau^{pg}, \tau^{proj}, \tau^{tang}, \tau^{lmh} \in (0, 1)$, $\tau^{lmg} > 0$ and $\tau_4 > 1$. Choose initial Lagrange multiplier λ_{-1} and compute λ_0 by solving (42) with linear solver tolerance (43).

For $k = 0, 1, 2, \ldots$

1. **Convergence check:** If $\|\nabla_x L(x_k, \lambda_k)\|_X < tol^{SQP}$ and $\|c(x_k)\|_C < tol^{SQP}$, then **stop**.
2. **Step computation:**
   a. Compute quasi-normal step n_k using Algorithm 2 and linear solver tolerance (34).
   b. Compute $\widetilde{t}_k, \widetilde{t}_k^{cp}$ using Algorithm 3 and linear solver tolerances (37), (39).
3. **Step acceptance:**

   For $i = 0, 1, 2, \ldots$

   a. **For** $j = 0, 1, 2, \ldots$

      i. Compute tangential step t_k by solving (40) with linear solver tolerance (41).
      ii. Compute Lagrange multiplier estimate λ_{k+1} at $x_k + n_k + t_k$ by solving (42) with linear solver tolerance (43).
      iii. Update the penalty parameter: If

$$\text{pred}(n_k, \widetilde{t}_k; \rho_{k-1}) \geq \frac{\rho_{k-1}}{2} \left(\|c(x_k)\|_C^2 - \|c_x(x_k)n_k + c(x_k)\|_C^2 \right)$$

      then set $\rho_k = \rho_{k-1}$. Otherwise set

$$\rho_k = \frac{-2\,\text{pred}(n_k, \widetilde{t}_k; \rho_{k-1})}{\|c(x_k)\|_C^2 - \|c_x(x_k)n_k + c(x_k)\|_C^2} + 2\rho_{k-1} + \bar{\rho}.$$

      iv. If $|\text{rpred}(c_x(x_k)t_k; \rho_k)| > \eta_0\,\text{pred}(n_k, \widetilde{t}_k; \rho_k)$, set $\tau^{tang} = 10^{-3}\,\tau^{tang}$, else **break**.

   End For (j)
   Reset τ^{tang} to its value at outer iteration i prior to Step 3a.

b. If $\|\widetilde{t}_k\|_X > \tau_4 \|n_k + t_k\|_X$ and $\widetilde{t}_k = \widetilde{t}_k^{cp}$

Set $\tau^{qn} = 10^{-1}\tau^{qn}$, $\tau^{pg} = 10^{-1}\tau^{pg}$, $\tau^{proj} = 10^{-1}\tau^{proj}$, $\tau^{tang} = 10^{-1}\tau^{tang}$.
Recompute n_k using Algorithm 2 and linear solver tolerance (34).
Recompute $\widetilde{t}_k$, $\widetilde{t}_k^{cp}$ using Algorithm 3 and linear solver tolerances (37), (39).

Else If $\|\widetilde{t}_k\|_X > \tau_4 \|n_k + t_k\|_X$ and $\widetilde{t}_k \neq \widetilde{t}_k^{cp}$

Set $\widetilde{t}_k = \widetilde{t}_k^{cp}$.

Else

Optional: Reset τ^{qn}, τ^{pg}, τ^{proj}, and τ^{tang} to their values from the Initialization step.
break

End For (i)

4. **Trust-region update:**

a. Compute trial step $s_k = n_k + t_k$.
b. Compute ratio $\theta_k = \text{ared}(s_k; \rho_k)/\text{pred}(n_k, \widetilde{t}_k; \rho_k)$.
c. If $\theta_k \geq \eta_1$, set $x_{k+1} = x_k + s_k$, and choose Δ_{k+1} as follows:

If $\theta_k \geq 0.9$, set
$\Delta_{k+1} = \min\{\max\{7\,\|s_k\|_{\mathcal{X}}, \Delta_k, \Delta_{min}\}, \Delta_{max}\}$
Else If $\theta_k \geq 0.8$, set
$\Delta_{k+1} = \min\{\max\{2\,\|s_k\|_{\mathcal{X}}, \Delta_k, \Delta_{min}\}, \Delta_{max}\}$
Else set
$\Delta_{k+1} = \max\{\Delta_k, \Delta_{min}\}$;

Else set $x_{k+1} = x_k$, $\lambda_{k+1} = \lambda_k$, and $\Delta_{k+1} = \alpha_1 \|s_k\|_X$.

End For (k)

4.2.6 Related Work

The above approach is based on the general composite-step trust-region framework presented by Dennis, El-Alem, and Maciel in [14]. To accommodate inexact computations, Algorithm 4 includes several modifications of [14]: (i) the gradient condition (22); (ii) the redefinition of the predicted reduction to evaluate progress, given in (27) and (28); and (iii) the tangential step conditions (29), (30), and (31). Modification (i) is derived from Heinkenschloss and Vicente [21]. It is related to the gradient condition (8) for reduced-space (unconstrained) formulations, which is discussed in detail in Section 4.1.1. Modifications (ii) and (iii) are related to similar such conditions proposed in [21]. However, we note that Heinkenschloss and Vicente assume a decomposition of optimization variables into basic and

nonbasic (state and control) variables, i.e., solve problem (1), whereas Algorithm 4 is applied to problem (3). The latter requires a different algorithmic strategy to enforce (ii) and (iii). Specifically, in [21] the state/control decomposition assumption allows one to set the control component of the quasi-normal step to zero, to formulate and solve the tangential subproblem for the control component only, and to separately compute the state component of the tangential step. In Algorithm 4, these computations are interconnected and therefore more involved when it comes to specifying concrete subalgorithms. However, Algorithm 4 is more general in the sense that it can be extended to the case where the constraint Jacobian is rank deficient. Additionally, Algorithm 4 enables the use of efficient iterative solvers and preconditioners for linear optimality systems.

Another composite-step trust-region approach using the basic/nonbasic decomposition of the equality constraints is presented by Ziems and Ulbrich in [43], where the emphasis is on an efficient management of adaptive PDE discretizations, i.e., control of finite element discretization error. In [44], Ziems extends the approach to handle additional constraints, such as bounds, on the control (nonbasic) variables. To rigorously incorporate finite element error estimates in the trust-region algorithm, in addition to the previously reviewed concepts, Ziems and Ulbrich require the notion of inexact actual reductions and propose implementable conditions to control such inexactness.

5 Application to Risk-Neutral Optimization

In this section, we specialize the reviewed algorithms to optimization problems where the governing PDEs include uncertain or random coefficients and the objective function is an expectation. First we present a reduced-space method that enables efficient use of dimension-adaptive sparse grids in the computation of the (reduced) objective function and its gradient. Subsequently, we discuss the numerical solution of augmented systems arising in the full-space approach to PDE-constrained optimization under uncertainty.

We denote the random inputs to a PDE model by ξ, which is a random vector defined on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Here, Ω is the set of outcomes, $\mathcal{F} \subset 2^{\Omega}$ is a σ-algebra of all possible events, and $\mathbb{P} : \mathcal{F} \to [0, 1]$ is a probability measure. As is common in the literature, we assume finite-dimensional noise, i.e., $\Xi := \xi(\Omega) \subset \mathbb{R}^m$ for some $m \in \mathbb{N}$. We further assume that Ξ is the m-fold tensor product of one-dimensional intervals Ξ_ℓ, $\ell = 1, \ldots, m$, and the components of ξ are independent and continuously distributed with one-dimensional Lebesgue densities $\rho_\ell : \Xi_\ell \to [0, \infty)$. In this setting, the PDE solution operator $S(z)$ is a random field with realizations in U. We denote the dependence of S on the random input ξ by $S(z; \xi)$ for a fixed control $z \in Z$ and note that $u = S(z; \xi) \in U$ solves the parametrized PDE

$$c(u, z; \xi) = 0 \quad \text{a.s.}$$

where $c : U \times Z \times \Xi \to C$. Here "a.s." is an abbreviation for "almost surely"; in other words, "up to a set of probability zero." Similarly, the objective function is parametrized as $J : U \times Z \times \Xi \to \mathbb{R}$. As in Section 3, we consider the full space problem

$$\min_{u \in U,\, z \in Z} \quad \mathbb{E}[J(u, z; \xi)] \tag{46a}$$

$$\text{subject to} \quad c(u, z; \xi) = 0 \quad \text{a.s.}, \tag{46b}$$

where $\mathbb{E}[X] := \int_\Omega X(\omega)\, \mathrm{d}\mathbb{P}(\omega)$ denotes the expectation of the random variable X. When evaluating the expectation of random variables with the form $f(\xi)$ where $f : \Xi \to \mathbb{R}$, the finite-dimensional noise assumption permits the following substitution:

$$\mathbb{E}[\, f(\xi)\,] = \int_{\Xi_1} \rho_1(\xi_1) \cdots \int_{\Xi_m} \rho_m(\xi_m) f(\xi)\, \mathrm{d}\xi_m \cdots \mathrm{d}\xi_1. \tag{47}$$

We slightly abuse notation and use $\xi = (\xi_1, \ldots, \xi_m)$ to denote both the random vector of inputs and its realizations. Substituting $S(z; \xi)$ into J produces the random-variable objective function $\widehat{J}(z; \xi) = J(S(z; \xi), z; \xi)$, leading to the reduced problem

$$\min_{z \in Z} \{\mathcal{J}(z) := \mathbb{E}[\, \widehat{J}(z; \xi)\,]\}. \tag{48}$$

To approximate the expectation in (46) and (48), we employ sparse-grid quadrature [6, 7, 19, 31, 37, 42]. Let $\{\mathbb{E}_\ell^i\}_{i \geq 1}$ be a sequence of one-dimensional quadrature operators of increasing order in the $\ell = 1, \ldots, m$ direction. That is, $\mathbb{E}_\ell^{i+1}$ is exact for higher-order monomials than $\mathbb{E}_\ell^i$. Define the 1-D difference quadrature operators

$$\delta_\ell^1 := \mathbb{E}_\ell^1 \quad \text{and} \quad \delta_\ell^i := \mathbb{E}_\ell^i - \mathbb{E}_\ell^{i-1}, \quad \text{for} \quad i \geq 2.$$

To define the m-dimensional quadrature rule on $\Xi = \Xi_1 \times \cdots \times \Xi_M$ let $\mathbf{i} = (i_1, \ldots, i_m)$ be a multi-index and let $\mathcal{I} \subset \mathbb{N}^m$ be a finite multi-index set. The general sparse-grid quadrature operator is defined as

$$\mathbb{E}_\mathcal{I} := \sum_{\mathbf{i} \in \mathcal{I}} (\delta_1^{i_1} \otimes \cdots \otimes \delta_m^{i_m}). \tag{49}$$

To ensure consistency of (49), $\mathcal{I}$ must satisfy the following condition: if $\mathbf{i} = (i_1, \ldots, i_m) \in \mathcal{I}$, $\mathbf{j} = (j_1, \ldots, j_m) \in \mathbb{N}^m$, and $j_\ell \leq i_\ell$ for all $\ell = 1, \ldots, m$, then $\mathbf{j} \in \mathcal{I}$. If $\mathcal{I}$ satisfies this condition, then we say that $\mathcal{I}$ is *admissible*. In one dimension, admissibility guarantees that (49) is a telescoping sum and recovers $\mathbb{E}_1^i$ where i denotes the maximum element of $\mathcal{I}$. An example of an admissible index set is

$$\mathcal{I} = \left\{ \mathbf{i} \in \mathbb{N}^m \, : \, |i_1| + \ldots + |i_m| \leq \ell + m - 1 \right\}$$

for $\ell \in \mathbb{N}$ which results in the standard isotropic sparse grid.

In general, $\mathbb{E}_{\mathcal{I}}[f]$ for $f : \Xi \to \mathbb{R}$ can be written as

$$\mathbb{E}_{\mathcal{I}}[f] = \sum_{k=1}^{Q} w_k f(\xi_k) \tag{50}$$

where w_k are the quadrature weights associated with the quadrature points ξ_k, $k = 1, \ldots, Q$. The form of approximation (50) is not unique to sparse grids as virtually all quadrature and sampling methods have this form. Applying $\mathbb{E}_{\mathcal{I}}$ for fixed index set $\mathcal{I}$ to (46) and (48) results in the approximate optimization problems

$$\min_{u \in U, \, z \in Z} \quad \sum_{k=1}^{Q} w_k J(u_k, z; \xi_k)] \tag{51a}$$

$$\text{subject to} \quad c(u_k, z; \xi_k) = 0, \quad k = 1, \ldots, Q \, , \tag{51b}$$

and

$$\min_{z \in Z} \sum_{k=1}^{Q} w_k \widehat{J}(z; \xi_k) \tag{52}$$

where $\widehat{J}(z; \xi_k) = J(S(z; \xi_k), z; \xi_k)$ and $S(z; \xi_k) = u_k \in U$ solves (51b) for $k = 1, \ldots, Q$. When considering the full-space algorithm, we require the Lagrangian functional associated with (51), i.e.,

$$L(u_1, \ldots, u_Q, z, \lambda_1, \ldots, \lambda_Q) := \sum_{k=1}^{Q} w_k J(u_k, z; \xi_k) + \sum_{k=1}^{Q} v_k \langle \lambda_k, c(u_k, z; \xi_k) \rangle_C \tag{53}$$

where v_k, $k = 1, \ldots, Q$, are fixed weights. We can choose v_k to be $v_k = 1$ or $v_k = w_k$ for $k = 1, \ldots, Q$. The later choice corresponds to an infinite-dimensional view of the problem since

$$\sum_{k=1}^{Q} w_k \langle \lambda_k, c(u_k, z; \xi_k) \rangle_C$$

is an approximation of the expectation

$$\mathbb{E}[\langle \lambda, c(u(\xi), z; \xi) \rangle_C].$$

5.1 Sparse-Grid Adaptivity

In the subsequent subsections, we define the adaptive sparse-grid subalgorithms used to satisfy (8) and (10). To do so, we require the definition of the forward neighborhood of the admissible index set $\mathcal{I}$. The forward neighborhood of $\mathcal{I}$ is

$$\mathcal{N}(\mathcal{I}) := \{\mathbf{i} \in \mathbb{N}^m \setminus \mathcal{I} \,:\, \mathcal{I} \cup \{\mathbf{i}\} \text{ is admissible}\}.$$

We employ dimension-adaptive sparse grids [19] in an attempt to satisfy (8) and (10). The dimension-adaptive sparse grid algorithm approximates the quadrature error on a subset $\mathcal{A}$ of the forward neighborhood of the current admissible index set $\mathcal{O}$, i.e., $\mathcal{A} \subseteq \mathcal{N}(\mathcal{O})$.

5.1.1 Computation of Inexact Gradient

Given the current iterate $z_k \in Z$, we must construct a model m_k that satisfies (8). To do so, we employ an admissible index set $\mathcal{I}_k^{\mathrm{g}} \subset \mathbb{N}^m$ and the associated quadrature approximation of $\mathcal{J}(z) = \mathbb{E}[\,\widehat{J}(z;\xi)\,]$, i.e.,

$$\mathcal{J}_{\mathcal{I}_k^{\mathrm{g}}}(z) = \sum_{\mathbf{i}\in\mathcal{I}_k^{\mathrm{g}}} (\delta_1^{i_1} \otimes \cdots \otimes \delta_m^{i_m})[\,\widehat{J}(z;\xi)\,].$$

We then choose our model m_k to satisfy the first-order condition $\nabla m_k(0) = \nabla \mathcal{J}_{\mathcal{I}_k^{\mathrm{g}}}(z_k)$. Under the assumption of convergence of (49), we can write the quadrature error associated with the index set $\mathcal{I}_k^{\mathrm{g}}$ as the sum of all differential quadrature rules $(\delta_1^{i_1} \otimes \cdots \otimes \delta_m^{i_m})$ for $\mathbf{i} \notin \mathcal{I}_k^{\mathrm{g}}$. Thus, the inexact gradient condition (8) becomes

$$\left\| \sum_{\mathbf{i}\notin\mathcal{I}_k^{\mathrm{g}}} (\delta_1^{i_1} \otimes \cdots \otimes \delta_M^{i_M})[\nabla \widehat{J}(z;\xi)] \right\|_Z \leq \kappa_{\mathrm{grad}} \min\left\{ \left\| \nabla \mathcal{J}_{\mathcal{I}_k^{\mathrm{g}}}(z_k) \right\|_Z, \Delta_k \right\}. \tag{54}$$

The goal now is to determine the smallest admissible index set $\mathcal{I}_k^{\mathrm{g}}$ such that (54) holds. Since it is not computationally feasible to explicitly evaluate the left-hand side of (54), we employ the dimension-adaptive approach presented in [19] to approximately satisfy this condition. Although there is no proof that this approach satisfies (8), numerical experience suggests that (8) is typically satisfied. The dimension-adaptive gradient computation algorithm is listed in Algorithm 5.

5.1.2 Computation of Inexact Objective Function Value

Similar to the inexact gradient computation, we define our objective function approximation for the computation of cred_k as

$$\mathcal{J}_k(z) = \mathcal{J}_{\mathcal{I}_k^{\mathrm{o}}}(z) = \sum_{\mathbf{i}\in\mathcal{I}_k^{\mathrm{o}}} (\delta_1^{i_1} \otimes \cdots \otimes \delta_m^{i_m})[\widehat{J}(z;\xi)].$$

Here, $\mathcal{I}_k^{\mathrm{o}} \subset \mathbb{N}^m$ is some admissible index set. The error associated with this approximation, in the context of (10), is

$$|\mathrm{ared}_k - \mathrm{cred}_k| = \left| \sum_{\mathbf{i}\notin\mathcal{I}_k^{\mathrm{o}}} (\delta_1^{i_1} \otimes \cdots \otimes \delta_m^{i_m})[\widehat{J}(z_k + s_k;\xi) - \widehat{J}(z_k;\xi)] \right| = \theta_k. \tag{55}$$

Algorithm 5 (Gradient computation using adaptive sparse grids)

Initialization: *Set* $\mathbf{i} = (1,\ldots,1)$, $\mathcal{A} = \{\mathbf{i}\}$, $\mathcal{O} = \emptyset$, $g_{\mathbf{i}} = (\delta_1^{i_1} \otimes \cdots \otimes \delta_M^{i_M})[\nabla\widehat{J}(z_k;\xi)]$ *and* $\beta = \beta_{\mathbf{i}} = \|g_{\mathbf{i}}\|_{\mathcal{Z}}$, $g = g_{\mathbf{i}}$, *and* $\mathrm{TOL} = \kappa_{\mathrm{grad}} \min\{\|g\|_{\mathcal{Z}}, \Delta_k\}$.

While $\beta > \mathrm{TOL}$

1. *Select* $\mathbf{i} \in \mathcal{A}$ *corresponding to the largest* $\beta_{\mathbf{i}}$
2. *Set* $\mathcal{A} \leftarrow \mathcal{A} \setminus \{\mathbf{i}\}$ *and* $\mathcal{O} \leftarrow \mathcal{O} \cup \{\mathbf{i}\}$
3. *Update the error indicator* $\beta \leftarrow \beta - \beta_{\mathbf{i}}$
4. ***For*** $\ell = 1,\ldots,m$
   a. *Set* $\mathbf{j} = \mathbf{i} + \mathbf{e}_\ell$
   b. ***If*** $\mathcal{O} \cup \{\mathbf{j}\}$ *is admissible*
      i. *Set* $\mathcal{A} \leftarrow \mathcal{A} \cup \{\mathbf{j}\}$
      ii. *Set* $g_{\mathbf{j}} = (\delta_1^{j_1} \otimes \cdots \otimes \delta_m^{j_m})[\nabla\widehat{J}(z_k;\xi)]$
      iii. *Set* $\beta_{\mathbf{j}} = \|g_{\mathbf{j}}\|_{\mathcal{Z}}$
      iv. *Update the gradient approximation* $g \leftarrow g + g_{\mathbf{j}}$
      v. *Update the error indicator* $\beta \leftarrow \beta + \beta_{\mathbf{j}}$
      vi. *Update the stopping tolerance* $\mathrm{TOL} = \kappa_{\mathrm{grad}} \min\{\|g\|_{\mathcal{Z}}, \Delta_k\}$
   c. ***EndIf***
5. ***EndFor***

EndWhile

Set $\mathcal{I}_k^{\mathrm{g}} = \mathcal{A} \cup \mathcal{O}$ *and* $\nabla m_k(0) = g$.

Again, we use dimension-adaptive sparse grids to determine $\mathcal{I}_k^{\mathrm{o}}$ where we estimate (55) only on a subset of the forward neighborhood of the current index set. Similar to the gradient computation, there is no guarantee that θ_k will satisfy (10). However, numerical experience suggest that this is often the case. The dimension-adaptive objective function approximation algorithm is listed in Algorithm 6.

5.2 *Iterative Linear System Solves*

In this section, we focus on the key computational component of the subalgorithms of Algorithm 4, namely, the numerical solution of augmented systems. As mentioned earlier, in optimization under uncertainty these systems are enormous and cannot be formed explicitly, requiring iterative, matrix-free methods.

Augmented systems are KKT systems for a special type of quadratic optimization problems. Solution methods for KKT systems have received significant attention recently in the context of PDE-constrained optimization. Efficient preconditioning

Algorithm 6 (Objective function evaluation using adaptive sparse grids)

Initialization: *Set* $\mathbf{i} = (1, \ldots, 1)$, $\mathcal{A} = \{\mathbf{i}\}$, $\mathcal{O} = \emptyset$, $\mathrm{TOL} = (\eta \min\{\mathrm{pred}_k, r_k\})^{1/\omega}$, $\tilde{\theta}_k = \vartheta_{\mathbf{i}} = (\delta_1^{i_1} \otimes \cdots \otimes \delta_M^{i_M})[\widehat{J}(z_k + s_k; \xi) - \widehat{J}(z_k; \xi)]$ *and* $\mathrm{cred}_k = \vartheta_{\mathbf{i}}$.

While $|\tilde{\theta}_k| > \mathrm{TOL}$

1. *Select* $\mathbf{i} \in \mathcal{A}$ *corresponding to the largest* $|\vartheta_{\mathbf{i}}|$
2. *Set* $\mathcal{A} \leftarrow \mathcal{A} \setminus \{\mathbf{i}\}$ *and* $\mathcal{O} \leftarrow \mathcal{O} \cup \{\mathbf{i}\}$
3. *Update the error indicator* $\tilde{\theta}_k \leftarrow \tilde{\theta}_k - \vartheta_{\mathbf{i}}$
4. **For** $\ell = 1, \ldots, m$
   a. *Set* $\mathbf{j} = \mathbf{i} + \mathbf{e}_\ell$
   b. **If** $\mathcal{O} \cup \{\mathbf{j}\}$ *is admissible*
      i. *Set* $\mathcal{A} \leftarrow \mathcal{A} \cup \{\mathbf{j}\}$
      ii. *Set* $\vartheta_{\mathbf{j}} = (\delta_1^{j_1} \otimes \cdots \otimes \delta_m^{j_m})[\widehat{J}(z_k + s_k; \xi) - \widehat{J}(z_k; \xi)]$
      iii. *Update the computed reduction* $\mathrm{cred}_k \leftarrow \mathrm{cred}_k + \vartheta_{\mathbf{j}}$
      iv. *Update the error indicator* $\tilde{\theta}_k \leftarrow \tilde{\theta}_k + \vartheta_{\mathbf{j}}$
   c. **EndIf**
5. **EndFor**

EndWhile

Return $\mathcal{I}_k^{\mathrm{o}} = \mathcal{A} \cup \mathcal{O}$ *and* cred_k.

approaches based on Schur complements in the constraint null space have been developed, see, e.g., [33–35, 38, 39]. Augmented systems are treated in [36], where it is shown that Schur-complement ideas lead to preconditioners that perform well for a variety of physics models, i.e., constraint equations, independent of the mesh size. A crucial difference between the KKT systems for the subproblem (13a)–(13b) and augmented systems is that the latter do not depend at all on the

objective function J, which lowers the bar for efficient preconditioning and affords generality.[4]

Recalling the assumption in full-space methods that C is a Hilbert space (with $C^* = C$), in PDE-constrained optimization the augmented system operator $G : X \times C \to X \times C$, in general, written as

$$G = \begin{pmatrix} I_X & c_x(x_k)^* \\ c_x(x_k) & 0 \end{pmatrix},$$

takes the form $A : U \times Z \times C \to U \times Z \times C$,

$$A = \begin{pmatrix} I_U & 0 & c_u(u_k, z_k)^* \\ 0 & I_Z & c_z(u_k, z_k)^* \\ c_u(u_k, z_k) & c_z(u_k, z_k) & 0 \end{pmatrix} =: \begin{pmatrix} I_U & 0 & C_u^* \\ 0 & I_Z & C_z^* \\ C_u & C_z & 0 \end{pmatrix},$$

where the latter notation is used as shorthand. Assuming the existence of $(c_u(u_k, z_k))^{-1}$, we consider two preconditioners for the operator A:

$$P^\star = \begin{pmatrix} I_U & 0 & 0 \\ 0 & I_Z & 0 \\ 0 & 0 & (C_u C_u^* + C_z C_z^*)^{-1} \end{pmatrix} \quad \text{and} \quad P = \begin{pmatrix} I_U & 0 & 0 \\ 0 & I_Z & 0 \\ 0 & 0 & C_u^{-*} C_u^{-1} \end{pmatrix}.$$

The preconditioner $P^\star$ is an exact Schur-complement preconditioner, in the sense that a $P^\star$-preconditioned Krylov solver for a system given by the operator A converges in at most three iterations [30]. However, in PDE-constrained optimization under uncertainty, the operator $C_u C_u^* + C_z C_z^*$ is never formed explicitly, due to its sheer size. Also, approximating the inverse of the sum of matrix products is in general very difficult. The approximate preconditioner P is a practical alternative. The application of P amounts to a "linearized state solve" followed by an "adjoint solve," which are readily available in practice. Following the notation in problem (51), with Lagrangian (53), for a given number Q of samples and weights the matrices C_u and C_z have special structure, namely that of a block-diagonal and a block-column matrix with Q nonzero blocks, respectively,

$$C_u = \begin{pmatrix} v_1 C_u^1 & 0 & \dots & 0 \\ 0 & v_2 C_u^2 & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \dots & v_Q C_u^Q \end{pmatrix} \quad \text{and} \quad C_z = \begin{pmatrix} v_1 C_z^1 \\ v_2 C_z^2 \\ \vdots \\ v_Q C_z^Q \end{pmatrix}.$$

[4] Under the assumptions of this chapter, augmented systems can always be related to *strictly convex quadratic problems* of the form $\min \frac{1}{2}\langle s^1, s^1\rangle_X - \langle b^1, s^1\rangle_X$ subject to $c_x(x_k)s^1 = b_2$, where $(b^1 \;\; b^2)^T$ is the right-hand side vector of the augmented system and s^1 is the first block of the left-hand side vector.

$$\begin{pmatrix}
I_U & 0 & \dots & 0 & 0 & v_1(C_u^1)^* & 0 & \dots & 0 \\
0 & I_U & \dots & 0 & 0 & 0 & v_2(C_u^2)^* & \dots & 0 \\
\vdots & \vdots & \ddots & 0 & \vdots & \vdots & \vdots & \ddots & 0 \\
0 & 0 & \dots & I_U & 0 & 0 & 0 & \dots & v_Q(C_u^Q)^* \\
0 & 0 & \dots & 0 & I_Z & v_1(C_z^1)^* & v_2(C_z^2)^* & \dots & v_Q(C_z^Q)^* \\
v_1C_u^1 & 0 & \cdots & 0 & v_1C_z^1 & 0 & 0 & \dots & 0 \\
0 & v_2C_u^2 & \cdots & 0 & v_2C_z^2 & 0 & 0 & \dots & 0 \\
\vdots & \vdots & \ddots & 0 & \vdots & \vdots & \vdots & \ddots & 0 \\
0 & 0 & \dots & v_QC_u^Q & v_QC_z^Q & 0 & 0 & \dots & 0
\end{pmatrix}$$

Fig. 1 The augmented system in PDE-constrained optimization under uncertainty. The application of the preconditioner P to this system is highly parallelizable, due to the block-diagonal structure of the highlighted C_u and C_u^* blocks

This gives rise to the augmented system depicted in Figure 1. We note that preconditioning this system with P is highly parallelizable, due to the block-diagonal structure of C_u and C_u^*. We also note that only approximations of the inverses C_u^{-1} and C_u^{-*} are needed in the application of the preconditioner, enabling efficient iterative schemes that rely on whatever solvers are provided for the linearized forward and adjoint systems.

6 Numerical Examples

We consider a thermal fluid application motivated by the transport process in high-pressure chemical vapor deposition (CVD) reactors (see Section 5.2 in [23]). Such reactors are used to produce compound semiconductors. Reactant gases are injected into the top of the reactor and must flow down to the substrate in order to form an epitaxial film. However, the substrate is maintained at a high temperature causing vorticities due to buoyancy-driven convection. For this application, we control the thermal flux on the side walls of the reactor to minimize vorticity. Let $D = (0, 1) \times (0, 1)$ and consider the following control problem:

$$\min_{z \in Z} \frac{1}{2}\mathbb{E}\left[\int_D (\nabla \times u(z))\,\mathrm{d}x\right] + \frac{\gamma}{2}\int_{\Gamma_c} |z|^2\,\mathrm{d}x$$

where $S(z) = (u(z), p(z), T(z)) = (u, p, T) : \Omega \to U$ solves the Boussinesq flow equations,

$$
\begin{aligned}
-\nu(\xi)\nabla^2 u + (u\cdot\nabla)u + \nabla p + \eta(\xi)Tg &= 0, && \text{in } D, \text{ a.s.},\\
\nabla\cdot u &= 0, && \text{in } D, \text{ a.s.},\\
-\kappa(\xi)\Delta T + u\cdot\nabla T &= 0, && \text{in } D, \text{ a.s.},\\
u - u_i &= 0, & T &= 0, \text{ on } \Gamma_i, \text{ a.s.},\\
u - u_o &= 0, & \kappa(\xi)\tfrac{\partial T}{\partial n} &= 0, \text{ on } \Gamma_o, \text{ a.s.},\\
u &= 0, & T &= T_b(\xi), \text{ on } \Gamma_b, \text{ a.s.},\\
u &= 0, & \kappa(\xi)\tfrac{\partial T}{\partial n} + h(\xi)(z-T) &= 0, \text{ on } \Gamma_c, \text{ a.s.},
\end{aligned}
$$

where $\Gamma_i = [1/3, 2/3]\times\{1\}$, $\Gamma_o = ([0, 1/3]\cup[2/3, 1])\times\{1\}$, $\Gamma_b = [0, 1]\times\{0\}$, and $\Gamma_c = \{0, 1\}\times[0, 1]$. The inflow and outflow velocities, u_i and u_o, are deterministic while the coefficients ν, η, κ, h, and T_b are uncertain. In this model,

$$
\nu = \frac{1}{\mathrm{Re}} = \frac{100}{1+0.01\xi_{N+1}}, \quad \eta = \frac{\mathrm{Gr}}{\mathrm{Re}^2} = 0.72\frac{1+0.01\xi_{N+1}}{1+0.01\xi_{N+2}},
$$

$$
\text{and} \quad \kappa = \frac{1}{\mathrm{Re}\,\mathrm{Pr}} = 10^5\frac{1+0.01\xi_{N+3}}{(1+0.01\xi_{N+1})^2}
$$

where Re is the Reynolds number, Gr is the Grashof number, and Pr is the Prandtl number. The offset N is the total number of random variables associated with T_0 and h. The uncertainty in T_0 is modeled by the expansion

$$
T_0(x,\xi) = 1 + 0.025\sum_{k=1}^{n_b}\xi_k\frac{\sqrt{2}\sin(\pi k x)}{\pi k}.
$$

The coefficient h has a similar expansion for $x = 0$ and for $x = 1$ with n_ℓ and n_r terms, respectively. All ξ_k are uniformly distributed on $[-1, 1]$. Figure 2 depicts the computational domain including boundary conditions (left) and the scenarios of the uncertain substrate temperature (right). The curves on top of the computational domain schematic (left) are the inflow and outflow profiles of the velocity, given by

$$
u(x) = \begin{cases} 2\left(\frac{1}{3}-x\right)x & \text{if } 0\le x\le\frac{1}{3}\\ -4\left(x-\frac{1}{3}\right)\left(\frac{2}{3}-x\right) & \text{if } \frac{1}{3}<x<\frac{2}{3}\\ 2\left(x-\frac{2}{3}\right)(1-x) & \text{if } \frac{2}{3}\le x\le 1\end{cases}.
$$

We study both the stated reduced-space formulation of the control problem and the corresponding full-space formulation. We use the Trilinos package Rapid Optimization Library [27] and its PDE-OPT Application Development Kit, available in the directory

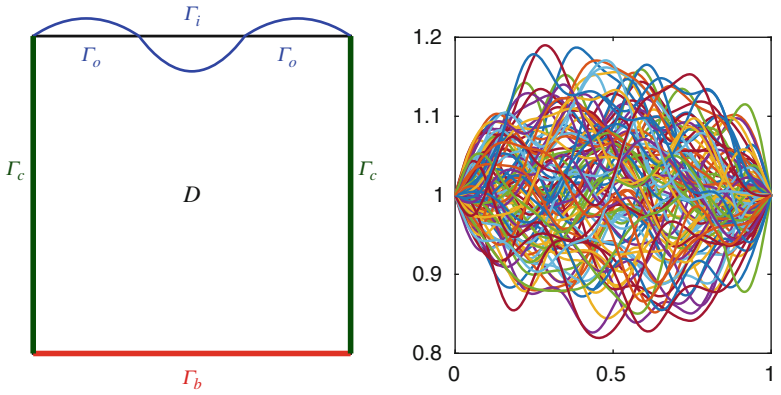

Fig. 2 Left: Computational domain for the CVD reactor. Right: Scenarios of T_0

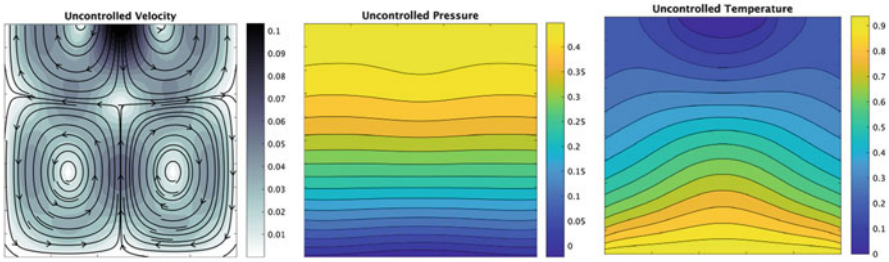

Fig. 3 Expected values of the uncontrolled velocity field (left), pressure (middle) and temperature (right)

Trilinos/packages/rol/examples/PDE-OPT,

to implement and solve the control problem. A reproducibility statement is given in Section 6.3.

To discretize the PDE, we use finite elements on a uniform mesh of 33×33 quadrilaterals. We note that to properly represent the boundary segments Γ_o and Γ_i the number of mesh cells in the horizontal direction should be divisible by three. For the velocity and pressure discretization, we use the Q2-Q1 Taylor–Hood finite element pair, and for the temperature we use the Q2 finite element. The sparse grids used to approximate the risk neutral objective function are built on one-dimensional Clenshaw–Curtis quadrature rules. We set the maximum sparse grid as the level-3 isotropic Clenshaw–Curtis sparse grid. This rule has $Q = 2245$ points. Figure 3 shows the expected values of the uncontrolled ($z = 0$) velocity field (streamlines and magnitude), pressure, and temperature.

6.1 Reduced-Space Results with Adaptive Sparse Grids

We solve the optimal control problem using the reduced-space trust-region approach with dimension-adaptive sparse grids described in Section 5.1. We terminate the algorithm when the gradient of the model at zero, $\|\nabla m_k(0)\|_Z$, is below $tol = 10^{-6}$. We set the initial guess to $z = 0$ and the initial trust-region radius to $\Delta_0 = 10$. We solve the trust-region subproblem using truncated conjugate gradients. We terminate the subproblem solve if the step has exceeded the trust-region radius, the algorithm encountered negative curvative or the residual is below the minimum of 10^{-4} and 10^{-2} times the norm of the initial residual. The problem is solved using a single computational core of a dual-socket 2.1 GHz Intel Broadwell E5-2695 compute node with 128 GB RAM. The linearized state and adjoint equations are solved using a direct solver. The solves are performed sequentially.

Figure 4 shows the expected values of the controlled velocity field (streamlines and magnitude), pressure, and temperature. We see a significant reduction of vorticity near the heated substrate. The left-wall and right-wall controls are given in Figure 5, respectively.

Table 1 displays the iteration history for the adaptive sparse grid algorithm described in Sections 4.1 and 5.1. The columns from left to right include the iteration count (iter), the computed objective function value ($\mathcal{J}_k(z_k)$), the norm of the model gradient ($\|\nabla m_k(0)\|_Z$), the trial step size ($\|s_k\|_Z$), the trust-region radius (Δ_k), the number of truncated conjugate gradient iterations (cg), a Boolean corresponding to whether the step was accepted or rejected (accept), the number of sparse-grid points for the objective function computation (obj), and the number of sparse-grid points for the gradient evaluation (grad). The algorithm starts with very few sparse-grid points (i.e., state and adjoint PDE solvers) and only refines the sparse grid as needed for global convergence. For this example, there were no "unimportant" directions resulting in a final sparse grid that is identical to the isotropic sparse grid. For examples with anisotropy, see [25, 26].

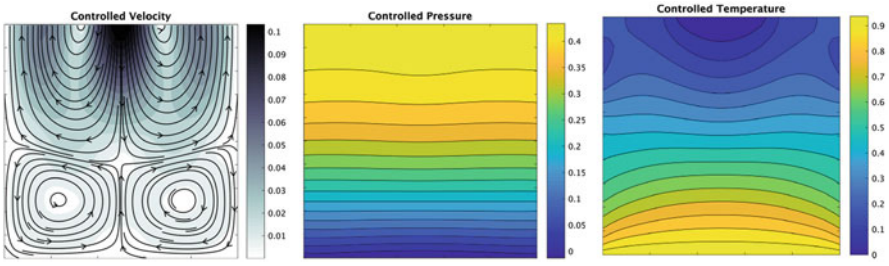

Fig. 4 Expected values of the controlled velocity field (left), pressure (middle), and temperature (right), obtained using the reduced-space method with adaptive sparse grids

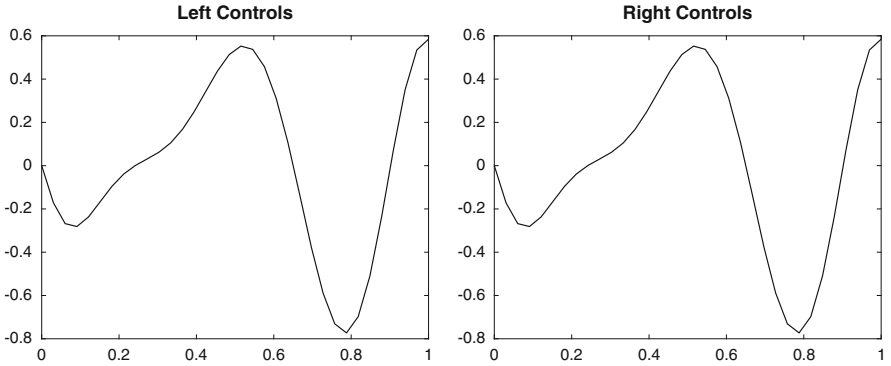

Fig. 5 Optimal controls along the left vertical side wall (left image) and the right vertical side wall (right image) of the problem domain D, obtained using the reduced-space method with adaptive sparse grids

Table 1 Iteration history for reduced-space adaptive sparse-grid approach

iter	$\mathcal{J}_k(z_k)$	$\|\nabla m_k(0)\|_Z$	$\|s_k\|_Z$	Δ_k	cg	accept	obj	grad
0	0.07457916	5.063×10^{-2}	–	10.000	–	–	1	3
1	0.07469930	5.063×10^{-2}	10.000	1.445	1	0	3	3
2	0.07469930	5.063×10^{-2}	1.445	0.361	1	0	3	3
3	0.05636707	4.875×10^{-2}	0.361	0.903	1	1	3	3
4	0.05636707	4.875×10^{-2}	0.903	0.226	1	0	3	3
5	0.04757099	2.059×10^{-2}	0.226	0.226	1	1	3	3
6	0.04680338	1.143×10^{-2}	0.226	0.226	2	1	103	117
7	0.04611002	3.468×10^{-3}	0.226	0.564	2	1	139	195
8	0.04511802	3.255×10^{-3}	0.564	1.411	2	1	117	233
9	0.04494516	1.085×10^{-3}	1.411	3.527	3	1	229	579
10	0.04499733	2.331×10^{-4}	2.838	8.818	6	1	579	949
11	0.04499338	6.211×10^{-5}	0.967	22.045	7	1	2245	1219
12	0.04499329	1.002×10^{-6}	0.127	55.113	8	1	2245	2245
13	0.04499327	7.034×10^{-9}	0.072	137.784	11	1	2245	2245

6.2 Full-Space Results with Iterative Linear System Solves

We now solve the optimal control problem using the full-space trust-region algorithm with iterative augmented system solves described in Section 5.2. The parameters for Algorithm 4 are

tol^{SQP}	tol^{CG}	ζ	Δ_0	Δ_{min}	Δ_{max}	α_1	η_1	η_0	ρ_{-1}	$\bar{\rho}$
10^{-6}	10^{-2}	0.8	10^4	10^{-10}	10^8	0.5	10^{-8}	0.5	1	10^{-8}

The nominal augmented system solver tolerances are set to $\tau^{qn} = \tau^{pg} = \tau^{proj} = \tau^{tang} = \tau^{lmh} = 10^{-6}$; however, we note that these tolerances are adjusted as needed by Algorithm 4. We set $\tau_4 = 2$ and $\tau^{lmg} = 10^4$. To solve augmented systems, we use the flexible generalized minimal residual (F-GMRES) method preconditioned with the Schur-complement preconditioner P discussed in Section 5.2. To apply the augmented system preconditioner P, for each block $v_k C_u^k$, $k = 1, \ldots, Q$, and its adjoint, we use GMRES preconditioned with a non-overlapping additive Schwarz domain-decomposition approach, where the domain D is partitioned into four horizontal strips of roughly equal size (the top strip is the largest one). For the (inner) F-GMRES stopping tolerance, we choose 10^{-4}. The linear solves on the subdomains are performed using a direct solver. As the initial guess for the control variables, we use $z = 0$. To obtain the initial guess for the state variables, we solve the nonlinear state equations with $z = 0$ for each sparse-grid point. We choose the infinite-dimensional view of the Lagrangian for the risk-neutral problem (51), i.e., $v_k = w_k$, for $k = 1, \ldots, Q$. We use a fixed level-3 Clenshaw–Curtis sparse grid with $Q = 2245$.

All studies were executed on the commodity cluster Serrano at Sandia National Labs. The results were obtained using 80 dual-socket 2.1 GHz Intel Broadwell E5-2695 nodes with 128 GB RAM. Each node has 36 cores, amounting to a total of 2880 cores. We utilized the hierarchical parallelism afforded by the Rapid Optimization Library, partitioning the cores into 720 groups, with each group using four cores for linear algebra tasks such as matrix assembly and iterative solves of the linearized state and adjoint equations needed to apply the preconditioner P. These solves are executed concurrently across the 720 groups. Considering that we process 2245 sparse-grid points, each group performs only three or four linearized state and adjoint solves per preconditioner application, enabling a high degree of concurrency in the computation. Informative studies can be performed without high-performance computing resources, using, e.g., level-2 sparse grids, which amounts to changing the "Maximum Sparse Grid Level" parameter in the input scripts described in Section 6.3 from 3 to 2.

Figure 6 shows the expected values of the controlled velocity field (streamlines and magnitude), pressure, and temperature. The left-wall and right-wall controls are given in Figure 7, respectively. As before, we see a significant reduction of vorticity near the heated substrate. In Table 2, we observe that the final objective value is different than in the reduced-space case. Figure 7 reveals that the optimal controls are also different. This is not unexpected, as our optimal control problem is nonconvex and may have multiple local minima. Substituting the full-space optimal controls into the reduced-space method, and vice versa, confirms that these are indeed locally optimal for both methods and that multiple numerical minima exist.

Table 2 shows the iteration history for the full-space approach. The columns from left to right denote the iteration count (iter), the computed objective function value ($J(x_k)$), the norm of the constraint($\|c(x_k)\|_C$), the norm of the gradient of the Lagrangian ($\|\nabla L(x_k, \lambda_k)\|_X$), the trust-region radius (Δ_k), the number of projected conjugate gradient iterations (pcg) per SQP iteration, a Boolean corresponding to whether the step was accepted or rejected (accept), the cumulative number of

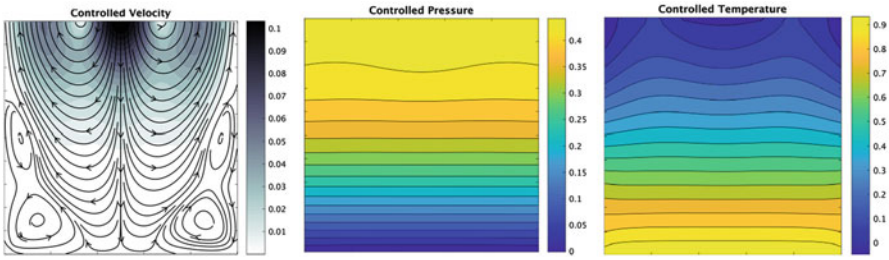

Fig. 6 Expected values of the controlled velocity field (left), pressure (middle), and temperature (right), obtained using the full-space method with iterative augmented system solves

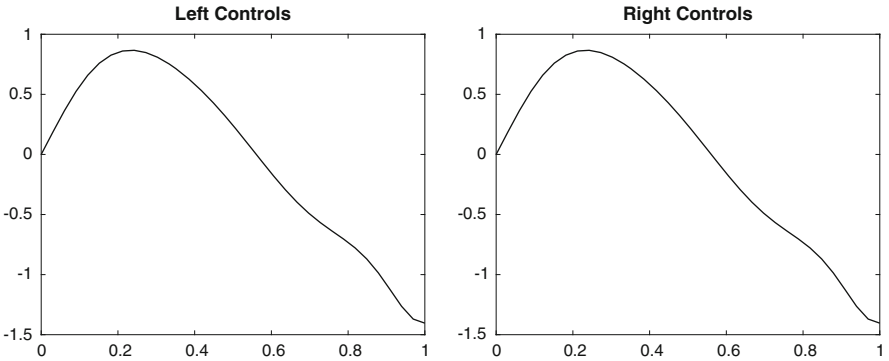

Fig. 7 Optimal controls along the left vertical side wall (left image) and the right vertical side wall (right image) of the problem domain D, obtained using the full-space method with iterative augmented system solves

Table 2 Iteration history for the full-space approach with iterative augmented system solves

iter	$J(x_k)$	$\|c(x_k)\|_C$	$\|\nabla L(x_k, \lambda_k)\|_X$	Δ_k	pcg	accept	ls calls	ls iters
0	0.07484675	7.820623×10^{-15}	8.377793×10^{-3}	1.00×10^{4}	–	–	–	–
1	0.05533699	1.661657×10^{-2}	3.641571×10^{-4}	1.00×10^{4}	11	1	16	597
2	0.03588474	3.052458×10^{-3}	9.338262×10^{-5}	1.00×10^{4}	13	1	33	1292
3	0.03515891	1.017679×10^{-4}	7.117806×10^{-5}	1.00×10^{4}	20	1	56	2303
4	0.03480817	1.444319×10^{-4}	2.439603×10^{-5}	1.00×10^{4}	15	1	75	3108
5	0.03480817	1.444319×10^{-4}	2.439321×10^{-5}	4.08×10^{0}	20	0	98	4157
6	0.03465050	2.237452×10^{-6}	4.364539×10^{-6}	3.03×10^{1}	2	1	104	4438
7	0.03464773	2.716452×10^{-7}	1.042585×10^{-7}	3.03×10^{1}	8	1	116	4989

calls to F-GMRES for augmented system solves (ls calls), and the cumulative number of F-GMRES iterations (ls iters). The first observation is that the full-space scheme converges robustly to the desired tolerance despite inexactness in the augmented system solves and inexactness in the Schur-complement preconditioner applications, i.e., linearized state and adjoint solves. Second, the average number

of P-preconditioned F-GMRES iterations per augmented system solve is roughly 43, which is encouraging considering that the size of the state space alone is 30,900,180 and that we have used fairly tight nominal tolerances for augmented system solves. Nonetheless, opportunities exist for preconditioner research in the context of optimization under uncertainty, and additional studies with larger nominal tolerances for augmented system solves are necessary.

6.3 Reproducibility

The numerical studies are contained in the directory

rol/examples/PDE-OPT/published/IMAvolumes_KouriRidzal2017

of the Rapid Optimization Library. The driver source file for the reduced-space studies is `example_RS.cpp`, with the accompanying input script `input_RS.xml`. The driver source file for the full-space studies is `example_FS.cpp`, with the accompanying input script `input_FS.xml`. The version of the Trilinos git repository used to generate all numerical results is labeled with the commit tag

`3958350daababd03f37fc422bf6a546d2d5ab5f5`,

and the branch is "`develop`." We report results with the Intel 17.0.0.098 compiler; however, we observed virtually identical results with GCC 6.1.0.

7 Conclusions

In recent years, trust-region methods have been extended to rigorously, robustly, and efficiently handle many sources of inexactness in the optimization process, including inexact evaluations of the objective and constraint functions and their derivatives as well as the inexact linear system solves arising in the approximation of constraint derivative inverses. In this chapter, we reviewed in some detail two such methods, which are particularly well suited to the solution of large-scale PDE-constrained optimization problems. The first method tackles the challenges of inexact objective function and gradient evaluations in unconstrained (reduced-space) formulations of PDE-constrained optimization problems, and is demonstrated in the context of sparse-grid adaptivity for risk-neutral optimization of thermal fluids. The second method deals with inexact linear system solves in constrained (full-space) formulations, and is demonstrated on large risk-neutral thermal-fluid optimization problems with fixed sparse grids, but with iterative, and therefore inexact, linearized state and adjoint solves. A principal remaining challenge in inexact trust-region methods for PDE-constrained optimization is in the handling of general inequality constraints on the control and state variables, with research opportunities in formulation and algorithm development, large-scale solvers for optimality systems, and efficient software implementations.

Acknowledgements This work was supported by DARPA EQUiPS grant SNL 014150709 and the DOE NNSA ASC ATDM program.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

References

1. N. Alexandrov. Robustness properties of a trust-region framework for managing approximation models in engineering optimization. In *Proceedings from the AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Work-in-progress Paper AIAA–96–4102-CP*, pages 1056–1059, 1996.
2. N. Alexandrov. A trust–region framework for managing approximations in constrained optimization problems. In *Proceedings of the First ISSMO/NASA Internet Conference on Approximation and Fast Reanalysis Techniques in Engineering Optimization, June 14–27, 1998*, 1998.
3. N. Alexandrov and J. E. Dennis. Multilevel algorithms for nonlinear optimization. In J. Borggaard, J. Burkardt, M. D. Gunzburger, and J. Peterson, editors, *Optimal Design and Control*, pages 1–22, Basel, Boston, Berlin, 1995. Birkhäuser Verlag.
4. N. Alexandrov, J. E. Dennis Jr., R. M. Lewis, and V. Torczon. A trust region framework for managing the use of approximation models in optimization. *Structural Optimization*, 15: 16–23, 1998. Appeared also as ICASE report 97–50.
5. E. Arian, M. Fahl, and E. W. Sachs. Trust–region proper orthogonal decomposition for flow control. Technical Report 2000–25, ICASE, NASA Langley Research Center, Hampton VA 23681–2299, 2000.
6. I. Babuška, F. Nobile, and R. Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Rev.*, 52(2):317–355, 2010.
7. V. Barthelmann, E. Novak, and K. Ritter. High dimensional polynomial interpolation on sparse grids. *Adv. Comput. Math.*, 12(4):273–288, 2000. Multivariate polynomial interpolation.
8. F. Bastin, C. Cirillo, and Ph. L. Toint. An adaptive Monte Carlo algorithm for computing mixed logit estimators. *Comput. Manag. Sci.*, 3(1):55–79, 2006.
9. M. Benzi, G. H Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14(1):1–137, 2005.
10. R. G. Carter. Numerical optimization in Hilbert space using inexact function and gradient evaluations. Technical Report 89–45, ICASE, Langley, VA, 1989.
11. R. G. Carter. On the global convergence of trust region algorithms using inexact gradient information. *SIAM J. Numer. Anal.*, 28:251–265, 1991.
12. R. G. Carter. Numerical experience with a class of algorithms for nonlinear optimization using inexact function and gradient information. *SIAM Journal on Scientific Computing*, 14(2):368–388, 1993.
13. A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust–Region Methods*. SIAM, Philadelphia, 2000.
14. J. E. Dennis, M. El-Alem, and M. C. Maciel. A Global Convergence Theory for General Trust–Region–Based Algorithms for Equality Constrained Optimization. *SIAM J. Optimization*, 7:177–207, 1997.
15. J. E. Dennis and V. Torczon. Approximation model managemet for optimization. In *Proceedings from the AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Work-in-progress Paper AIAA–96–4099-CP*, pages 1044–1046, 1996.

16. J. E. Dennis and V. Torczon. Managing approximation models in optimization. In N. Alexandrov and M. Y. Hussaini, editors, *Multidisciplinary Design Optimization. State of the Art*, pages 330–347, Philadelphia, 1997. SIAM.
17. J. E. Dennis, Jr. and R. B. Schnabel. *Numerical Methods for Nonlinear Equations and Unconstrained Optimization*. SIAM, Philadelphia, 1996.
18. M. Fahl and E.W. Sachs. Reduced order modelling approaches to PDE–constrained optimization based on proper orthogonal decompostion. In L. T. Biegler, O. Ghattas, M. Heinkenschloss, and B. van Bloemen Waanders van Bloemen Waanders, editors, *Large-Scale PDE-Constrained Optimization*, Lecture Notes in Computational Science and Engineering, Vol. 30, Heidelberg, 2003. Springer-Verlag.
19. T. Gerstner and M. Griebel. Dimension-adaptive tensor-product quadrature. *Computing*, 71(1):65–87, 2003.
20. M. Heinkenschloss and D. Ridzal. A matrix-free trust-region SQP method for equality constrained optimization. *SIAM Journal on Optimization*, 24(3):1507–1541, 2014.
21. M. Heinkenschloss and L. N. Vicente. Analysis of inexact trust–region SQP algorithms. *SIAM J. Optimization*, 12:283–302, 2001.
22. M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich. *Optimization with Partial Differential Equations*, volume 23 of *Mathematical Modelling, Theory and Applications*. Springer Verlag, Heidelberg, New York, Berlin, 2009.
23. K. Ito and S. S. Ravindran. Optimal control of thermally convected fluid flows. *SIAM J. on Scientific Computing*, 19:1847–1869, 1998.
24. C.T. Kelley and E.W. Sachs. Truncated newton methods for optimization with inaccurate functions and gradients. *Journal of Optimization Theory and Applications*, 116(1):83–98, 2003.
25. D. P. Kouri, M. Heinkenschloss, D. Ridzal, and B. G. van Bloemen Waanders. A trust-region algorithm with adaptive stochastic collocation for PDE optimization under uncertainty. *SIAM Journal on Scientific Computing*, 35(4):A1847–A1879, 2013.
26. D. P. Kouri, M. Heinkenschloss, D. Ridzal, and B. G. van Bloemen Waanders. Inexact objective function evaluations in a trust-region algorithm for PDE-constrained optimization under uncertainty. *SIAM Journal on Scientific Computing*, 36(6):A3011–A3029, 2014.
27. D. P. Kouri, G. von Winckel, and D. Ridzal. ROL: Rapid Optimization Library. https://trilinos.org/packages/rol, 2017.
28. L. Lubkoll, A. Schiela, and M. Weiser. An affine covariant composite step method for optimization with PDEs as equality constraints. *Optimization Methods and Software*, 32(5):1132–1161, 2017.
29. J. J. Moré. Recent developments in algorithms and software for trust region methods. In A. Bachem, M. Grötschel, and B. Korte, editors, *Mathematical Programming, The State of The Art*, pages 258–287. Springer Verlag, Berlin, Heidelberg, New-York, 1983.
30. M. F. Murphy, G. H. Golub, and A. J. Wathen. A note on preconditioning for indefinite linear systems. *SIAM Journal on Scientific Computing*, 21(6):1969–1972, 2000.
31. E. Novak and K. Ritter. High-dimensional integration of smooth functions over cubes. *Numer. Math.*, 75(1):79–97, 1996.
32. E. O. Omojokun. *Trust region algorithms for optimization with nonlinear equality and inequality constraints*. PhD thesis, Department of Computer Science, University of Colorado, Boulder, Colorado, 1989.
33. T. Rees, H. S. Dollar, and A. J. Wathen. Optimal Solvers for PDE-Constrained Optimization. *SIAM Journal on Scientific Computing*, 32(1):271–298, 2010.
34. T. Rees, M. Stoll, and A. Wathen. All-at-once preconditioning in PDE-constrained optimization. *Kybernetika*, 46(2):341–360, 2010.
35. T. Rees and A. J. Wathen. Preconditioning iterative methods for the optimal control of the Stokes equation. *SIAM J. Sci. Comput*, 33(5), 2010.
36. D. Ridzal. Preconditioning of a Full-Space Trust-Region SQP Algorithm for PDE-constrained Optimization. In *Report No. 04/2013: Numerical Methods for PDE Constrained Optimization with Uncertain Data*. Mathematisches Forschungsinstitut Oberwolfach, 2013.

37. S. A. Smoljak. Quadrature and interpolation formulae on tensor products of certain function classes. *Soviet Math. Dokl.*, 4:240–243, 1963.
38. M. Stoll. One-shot solution of a time-dependent time-periodic PDE-constrained optimization problem. *IMA Journal of Numerical Analysis*, 34(4):1554–1577, 2014.
39. M. Stoll and A. Wathen. All-at-once solution of time-dependent Stokes control. *Journal of Computational Physics*, 232(1):498–515, 2013.
40. Ph. L. Toint. Global convergence of a class of trust-region methods for nonconvex minimization in Hilbert space. *IMA Journal of Numerical Analysis*, 8:231–252, 1988.
41. S. Ulbrich and J. C. Ziems. Adaptive multilevel trust-region methods for time-dependent PDE-constrained optimization. *Portugaliae Mathematica*, 74(1):37–67, 2017.
42. D. Xiu and J. S. Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM J. Sci. Comput.*, 27(3):1118–1139 (electronic), 2005.
43. J. C. Ziems and S. Ulbrich. Adaptive multilevel inexact SQP methods for PDE-constrained optimization. *SIAM Journal on Optimization*, 21(1):1–40, 2011.
44. J. Carsten Ziems. Adaptive Multilevel Inexact SQP-Methods for PDE-Constrained Optimization with Control Constraints. *SIAM Journal on Optimization*, 23(2):1257–1283, 2013.