INDUSTRIAL AND SYSTEMS ENGINEERING

ISE

Tensor Train Solution to Uncertain Optimization Problems with Shared Sparsity Penalty

HARBIR ANTIL[1], SERGEY DOLGOV[2], AND AKWUM ONWUNTA[3]

[1]The Center for Mathematics and Artificial Intelligence (CMAI) and Department of Mathematical Sciences, George Mason University, Fairfax, VA 22030, USA

[2]Department of Mathematical Sciences, University of Bath, Bath, BA2 7AY, UK

[3]Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, 18015 USA

ISE Technical Report 24T-018

LEHIGH UNIVERSITY

TENSOR TRAIN SOLUTION TO UNCERTAIN OPTIMIZATION PROBLEMS WITH SHARED SPARSITY PENALTY*

HARBIR ANTIL†, SERGEY DOLGOV‡, AND AKWUM ONWUNTA§

Abstract. We develop both first and second order numerical optimization methods to solve non-smooth optimization problems featuring a shared sparsity penalty, constrained by differential equations with uncertainty. To alleviate the curse of dimensionality we use tensor product approximations. To handle the non-smoothness of the objective function we introduce a smoothed version of the shared sparsity objective. We consider both a benchmark elliptic PDE constraint, and a more realistic topology optimization problem. We demonstrate that the error converges linearly in iterations and the smoothing parameter, and faster than algebraically in the number of degrees of freedom, consisting of the number of quadrature points in one variable and tensor ranks. Moreover, in the topology optimization problem, the smoothed shared sparsity penalty actually reduces the tensor ranks compared to the unpenalized solution. This enables us to find a sparse high-resolution design under a high-dimensional uncertainty.

Key words. Shared sparsity, nonsmooth regularization, penalization, smoothing, tensor train, topology optimization

MSC codes. 49J55, 93E20, 49K20, 49K45, 90C15, 65D15, 15A69, 15A23

1. Introduction.

Constrained optimization is a standard approach in computer-aided design, where one needs to achieve a certain state of a mathematical model of a physical or technological process, often written in the form of Partial Differential Equations (PDE) nowadays, by minimizing a certain cost function, either fitting data or measuring the desired state (e.g. rigidity) in some other way. However, it becomes increasingly important to quantify the uncertainty emerging from the model and/or data, to both obtain and certify a control that is resilient to uncertainty, eventually paving the way towards 'digital twins' [4, 1]. Such optimization problems are starting to receive a tremendous amount of attention. There have been developments in all directions: modeling [11], analysis [32, 31], numerical analysis [2, 12, 18, 28], numerical linear algebra [9], and algorithms [18].

We consider a PDE

$$c(y, u, \xi(\omega)) = 0, \tag{1.1}$$

with $y(x, \xi) \in \mathcal{Y}$ and $u(x, \xi) \in \mathcal{U}_{ad}$ being the state and control belonging to a Hilbert state space $\mathcal{Y}$ and the admissible control subset $\mathcal{U}_{ad} \subset \mathcal{U}$ of a Hilbert space $\mathcal{U}$, respectively, $c \in \mathcal{Z}$ where $\mathcal{Z}$ is a residual Hilbert space, $x \in D \subset \mathbb{R}^{\hat{d}}$, a physical domain with Lipschitz boundary ∂D, and $\xi(\omega) : \Omega \mapsto \Xi \subset \mathbb{R}^d$ is a random vector. The latter is defined through a complete probability space $(\Omega, \mathfrak{F}, \mathbb{P})$, where Ω is a space of all possible outcomes, $\mathfrak{F} \subset 2^\Omega$ is a σ-algebra on Ω and $\mathbb{P} : \mathfrak{F} \to [0, 1]$ is an appropriate probability measure. For simplicity of exposition, we make a finite-dimension noise assumption encapsulated by the map ξ from Ω to a finite dimensional domain $\Xi = \xi(\Omega) \subset \mathbb{R}^d$ and continuous probability density function $\rho(\xi)$. The decision making then becomes a problem of optimizing a cost function,

$$\min_{y \in \mathcal{Y}, u \in \mathcal{U}_{ad}} \mathcal{J}(y, u), \quad \mathcal{J}(y, u) := \mathbb{E}[J(y, \xi)] + \mathbb{E}[S(u)] + \mathcal{R}(u), \tag{1.2}$$

constrained by (1.1). Here J is the random-variable objective, S and $\mathcal{R}$ respectively denote smooth and nonsmooth control regularizations.

*HA is partially supported by NSF grant DMS-2408877, the AirForce Office of Scientific Research under Award NO: FA9550-22-1-0248, and Office of Naval Research (ONR) under Award NO: N00014-24-1-2147. SD is thankful for the support from Engineering and Physical Sciences Research Council (EPSRC) New Investigator Award EP/T031255/1.

†The Center for Mathematics and Artificial Intelligence (CMAI) and Department of Mathematical Sciences, George Mason University, Fairfax, VA 22030, USA. hantil@gmu.edu

‡Department of Mathematical Sciences, University of Bath, Bath, BA2 7AY, UK. s.dolgov@bath.ac.uk

§Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA 18015, USA. ako221@lehigh.edu

There exist two versions of the uncertain PDE-constrained optimal control problem (1.1)–(1.2). Firstly, we can introduce random variables in the PDE, but seek a deterministic control $u(x)$ [27, 28, 18, 29]. A straightforward objective function of the state thus depends on the random variables too. Turning it into a deterministic total cost needs an expectation of either the objective itself (risk-neutral optimal control problem) or another function of the objective (risk-averse problem). Such controls are easy to realize, but may be restrictive or lack robustness. Secondly, we can introduce random variables in both state $y(x,\xi)$ and control $u(x,\xi)$ [12, 9, 40, 33]. This provides uncertainty quantification for control, but it is not immediately clear how to realize a random field control in practice.

An interesting compromise was introduced in [30], which develops the notion of 'shared-sparsity'. The idea is to consider an L^1-norm in the spatial (physical) domain and an L^2-norm in the random variable, i.e.,

$$\mathcal{R}(u(x,\xi)) = \beta \int_D \left(\int_\Xi |u(x,\xi)|^2 \rho(\xi) d\xi \right)^{\frac{1}{2}} dx\,, \tag{1.3}$$

where $\beta \geq 0$ is the regularization parameter. The optimization with a 1-norm penalty to promote sparsity (that is, to drive near-zero coefficients or gradients to be exactly zero) is a well-known technique in inverse problems [42, 32]. Here, we make the sparsity of the control (almost) deterministic, shared over all the realizations of the random variable. During the online stage, one can apply the mean or quantile of the control, or estimate a particular ξ from state measurements using e.g. Bayesian filtering [25, 7] and apply the control evaluated on this ξ. Crucially, since the positions of the zeros in the control are (almost) deterministic, one can fix the actuators to the rest of the spatial domain independently of the system state.

Our paper focuses on this approach. As the main application, we consider a topology optimization under uncertain Young's modulus. The goal is to find an optimal material distribution given by volume $\bar{V}$ over a given domain D by minimizing the compliance of a structure subject to elasticity equations as constraints. In this context, finding sparsity corresponds to identifying locations where one must add material. Then in the online stage, a structure can be designed using different realizations of the random variables, all producing a robust design. Topology optimization under uncertainty has received some attention recently [26, 41, 16], but these references do not focus on sparsity.

The main difficulty of uncertainty quantification is the curse of dimensionality: the number of random variables d needed to parametrize often originally infinite-dimensional random field can be tens to thousands. Direct discretization of each variable independently produces the number of unknowns growing exponentially in d, quickly becoming unfeasible. Except for particularly low d or smooth functions, where one can use sparse grids for instance [40, 33], almost all the existing algorithms in the nonsmooth setting use Monte-Carlo based sampling, which may converge rather slowly and lead to a sheer computational complexity due to the need to solve a lot of PDEs during the optimization.

Another approach that can approximate high-dimensional functions (like discretized random fields) is hierarchical tensor decompositions [21]. The idea is that we discretize a function using a Cartesian product of univariate basis functions, but instead of storing the tensor of expansion coefficients explicitly, we approximate it by a sum of products of low-dimensional factors. Smooth functions [22, 38], or probability density functions of weakly correlated random variables [36], were proven to admit low tensor ranks of such approximations, e.g. depending poly-logarithmically on the approximation error. Moreover, hierarchical tensor decompositions (specifically, the Tensor-Train (TT) decomposition [34] we use in this paper) are equipped with fast adaptive cross approximation algorithms [35, 14].

However, tensor approximations struggle with non-smooth functions. To apply them to non-smooth optimization, [6, 5] introduced smoothed objective functions such that the solution of the smoothed problem converges to the solution of the original problem as the smoothing parameter goes to zero, whereas for any positive smoothing parameter the solution exhibits a rapidly converging TT approximation.

We develop an ε-smoothing of the shared sparsity objective (1.3). Specifically, for any $\varepsilon > 0$ we re-

place (1.3) by

$$\mathcal{R}_\varepsilon(u(x,\xi)) = \beta \int_D \left(\int_\Xi u(x,\xi)^2 \rho(\xi) d\xi + \varepsilon^2 \right)^{\frac{1}{2}} dx \tag{1.4}$$

in the objective function (1.2).

The remainder of the article is organized as follows. Section 2 details the smoothed shared sparsity objective, its derivatives, optimality conditions in both continuous and discrete settings, and the fully discrete approximate Newton method with fast application of the Hessian. This formalism is independent of the particular approximation of functions and expectations over the random parameters. However, for a practical method free from the curse of dimensionality, we overview the Tensor-Train decomposition in Section 3. In Section 4 we test our methodology using a couple of numerical examples: an elliptic PDE in Section 4.1 for faster computations and more illustrative behaviour of the method, and a robust topology optimization problem [26, 41, 8] in Section 4.2.

2. Discretization and optimality conditions. We assume that both $y(x,\xi)$ and $u(x,\xi)$ are approximated by $y_h(x,\xi)$ and $u_h(x,\xi)$ which are expanded by $\hat{N}$ basis functions $\psi_1(x), \dots, \psi_{\hat{N}}(x)$ in the physical domain, discretised with a grid of the maximal element diameter $h > 0$. Without loss of generality, we can assume that the same basis functions are used for y and u (we can always concatenate different bases and expand both functions in the joint basis). For the TT decomposition method we further assume that $y_h(x,\xi)$ and $u_h(x,\xi)$ are expanded by N basis functions $\phi_1(\xi), \dots, \phi_N(\xi)$ on the random variable. Ultimately,

$$y_h(x,\xi) = \sum_{i=1}^{\hat{N}} \sum_{j=1}^{N} y_{i,j} \psi_i(x) \phi_j(\xi), \qquad u_h(x,\xi) = \sum_{i=1}^{\hat{N}} \sum_{j=1}^{N} u_{i,j} \psi_i(x) \phi_j(\xi),$$

and we can collect coefficients into vectors

$$\mathbf{y} = \left[y_{1,1}, \dots, y_{\hat{N},N} \right], \quad \mathbf{u} = \left[u_{1,1}, \dots, u_{\hat{N},N} \right]. \tag{2.1}$$

For Monte Carlo methods, we assume that ξ is sampled in N independent identically distributed points $\xi_1, \dots, \xi_N$, so we can expand

$$y_h(x,\xi_j) = \sum_{i=1}^{\hat{N}} y_{i,j} \psi_i(x), \qquad u_h(x,\xi_j) = \sum_{i=1}^{\hat{N}} u_{i,j} \psi_i(x), \qquad j = 1, \dots, N,$$

and collect the coefficients into vectors similarly to (2.1).

2.1. Full space formulation. First, we consider the Lagrangian formulation. Let $\lambda(x,\xi) \in \mathcal{Z}^*$ be the Lagrange multiplier belonging to the dual of the residual Hilbert space $\mathcal{Z} \ni c$, so we can identify $\mathcal{Z}^*$ with $\mathcal{Z}$. Then we need to find the KKT conditions of the Lagrangian

$$\mathcal{L}(y,u,\lambda) := \mathbb{E}[J(y,\xi)] + \mathbb{E}[S(u)] + \mathcal{R}_\varepsilon(u) - \int_D \int_\Xi \lambda(x,\xi) c(y,u,\xi)(x,\xi) \rho(\xi) d\xi dx. \tag{2.2}$$

We assume that the discretized Lagrange multiplier $\lambda_h(x,\xi)$ is expanded in the same basis as y_h and u_h. Plugging in the discrete solutions into (2.2), we obtain $\mathcal{L}_h(\mathbf{y},\mathbf{u},\boldsymbol{\lambda}) := \mathcal{L}(y_h,u_h,\lambda_h)$. Differentiating the latter with respect to the coefficient vectors $\mathbf{y},\mathbf{u},\boldsymbol{\lambda} \in \mathbb{R}^{\hat{N}N}$, we get the state and adjoint equations, and an expression of the gradient

$$(\nabla_{\mathbf{y}} \mathcal{L}_h)_{i,j} = \int_D \int_\Xi \Big(\nabla_y J(y_h,\xi) - \lambda_h \nabla_y c(y_h,u_h,\xi) \Big) \psi_i \phi_j \rho d\xi dx = 0, \tag{2.3}$$

$$(\nabla_{\boldsymbol{\lambda}} \mathcal{L}_h)_{i,j} = - \int_D \int_\Xi c(y_h,u_h,\xi) \psi_i \phi_j \rho d\xi dx = 0, \tag{2.4}$$

$$(\nabla_{\mathbf{u}} \mathcal{L}_h)_{i,j} = \int_D \int_\Xi \Big(\nabla_u S(u_h) - \lambda_h \nabla_u c(y_h,u_h,\xi) \Big) \psi_i \phi_j \rho d\xi dx + (\nabla_{\mathbf{u}} \mathcal{R}_\varepsilon)_{i,j}, \tag{2.5}$$

where ∇_y and ∇_u in the right-hand-side are Fréchet derivatives, in contrast to the gradient with respect to discrete coefficients in the left-hand-side. The latter finite-dimensional gradients will be used for the TT method. For the Monte Carlo method, we replace the integrals over ξ in the Lagrangian by the averages over samples at $\xi_1, \dots, \xi_N$:

$$\mathcal{L}_N(\mathbf{y}, \mathbf{u}, \boldsymbol{\lambda}) := \frac{1}{N} \sum_{j=1}^{N} \left[J(y_h, \xi_j) + S(u_h(x, \xi_j)) - \int_D \lambda_h(x, \xi_j) c(y_h, u_h, \xi_j) dx \right] + \mathcal{R}_{\varepsilon,N}(\mathbf{u}),$$

where

$$\mathcal{R}_{\varepsilon,N}(\mathbf{u}) = \beta \int_D \left(\frac{1}{N} \sum_{j=1}^{N} u_h(x, \xi_j)^2 + \varepsilon^2 \right)^{\frac{1}{2}} dx. \tag{2.6}$$

Since the coefficients corresponding to different samples are independent, the sum over j collapses in the following gradients:

$$\begin{aligned}
(\nabla_{\mathbf{y}} \mathcal{L}_N)_{i,j} &= \frac{1}{N} \left[(\nabla_{\mathbf{y}} J)_{i,j} - \int_D \lambda_h(x, \xi_j) (\nabla_y c(y_h, u_h, \xi_j)) \psi_i dx \right] = 0, \\
(\nabla_{\boldsymbol{\lambda}} \mathcal{L}_N)_{i,j} &= -\frac{1}{N} \int_D c(y_h, u_h, \xi_j) \psi_i dx = 0, \\
(\nabla_{\mathbf{u}} \mathcal{L}_N)_{i,j} &= \frac{1}{N} \left[(\nabla_{\mathbf{u}} S)_{i,j} - \int_D \lambda_h(x, \xi_j) (\nabla_u c(y_h, u_h, \xi_j)) \psi_i dx \right] + (\nabla_{\mathbf{u}} \mathcal{R}_{\varepsilon,N})_{i,j}.
\end{aligned}$$

2.2. Derivatives of the shared sparsity penalty. Both first and second derivatives of J, S and c will be instantiated in concrete examples to simplify their derivation. Here, we are concerned with the derivatives of (1.4) and (2.6).

For the TT method, plugging in the discretized solution into (1.4), we obtain

$$\mathcal{R}_\varepsilon(u_h(x, \xi)) = \beta \int_D \left(\int_\Xi \left(\sum_i \sum_j u_{i,j} \psi_i(x) \phi_j(\xi) \right)^2 \rho(\xi) d\xi + \varepsilon^2 \right)^{\frac{1}{2}} dx, \tag{2.7}$$

$$(\nabla_{\mathbf{u}} \mathcal{R}_\varepsilon)_{i,j} = \beta \int_D \frac{\int_\Xi \left(\sum_{i'} \sum_{j'} u_{i',j'} \psi_{i'}(x) \phi_{j'}(\xi) \right) \psi_i(x) \phi_j(\xi) \rho d\xi}{\sqrt{\int_\Xi \left(\sum_{i'} \sum_{j'} u_{i',j'} \psi_{i'}(x) \phi_{j'}(\xi) \right)^2 \rho(\xi) d\xi + \varepsilon^2}} dx. \tag{2.8}$$

In practice, we can always use the Lagrange interpolation basis functions $\phi_1(\xi), \dots, \phi_N(\xi)$, centered at nodes $\xi_1, \dots, \xi_N$ of a multivariate Gaussian grid, such that $\phi_j(\xi_{j'}) = \delta_{j,j'}$. The integral over ξ can then be approximated by a quadrature,

$$\int_\Xi f(\xi) \rho(\xi) d\xi \approx \sum_{j=1}^{N} w_j f(\xi_j),$$

where $w_1, \dots, w_N$ are quadrature weights. Moreover, the Gaussian quadrature is exact for polynomials of degree up to $2N - 1$. This gives

$$\int_\Xi \left(\sum_i \sum_j u_{i,j} \psi_i(x) \phi_j(\xi) \right)^2 \rho(\xi) d\xi = \sum_{j=1}^{N} w_j \left[\sum_{i,i'=1}^{\hat{N}} \left[u_{i,j} u_{i',j} \psi_i(x) \psi_{i'}(x) \right] \right].$$

The integral over x is taken of a non-polynomial function, and needs a quadrature too. The usual finite element practice is to take $\psi_1(x), \dots, \psi_{\hat{N}}(x)$ to be nodal functions on a grid $x_1, \dots, x_N$, i.e. $\psi_i(x_{i'}) = \delta_{i,i'}$. We can introduce a quadrature on the same nodes,

$$\int_D f(x)dx \approx \sum_{i=1}^{\hat{N}} \hat{w}_i f(x_i).$$

For example, the Trapezoidal rule with piecewise linear basis functions satisfies this assumption and gives the second order of consistency. This gives

$$\mathcal{R}_{\varepsilon,h}(\mathbf{u}) = \beta \sum_{i=1}^{\hat{N}} \hat{w}_i \sqrt{\sum_{j=1}^{N} \left[w_j u_{i,j}^2\right] + \varepsilon^2}, \tag{2.9}$$

$$(\nabla_{\mathbf{u}} \mathcal{R}_{\varepsilon,h})_{i,j} = \beta \hat{w}_i \frac{u_{i,j} w_j}{\sqrt{\sum_{j'=1}^{N} \left[w_{j'} u_{i,j'}^2\right] + \varepsilon^2}}, \tag{2.10}$$

$$(\nabla^2_{\mathbf{uu}} \mathcal{R}_{\varepsilon,h})_{i,j,i',j'} = \frac{\beta \hat{w}_i w_j \delta_{j,j'} \delta_{i,i'}}{\sqrt{\sum_{j''=1}^{N} \left[w_{j''} u_{i,j''}^2\right] + \varepsilon^2}} - \frac{\beta \hat{w}_i w_j w_{j'} u_{i,j} u_{i,j'} \delta_{i,i'}}{\left(\sum_{j''=1}^{N} \left[w_{j''} u_{i,j''}^2\right] + \varepsilon^2\right)^{\frac{3}{2}}}. \tag{2.11}$$

For the Monte Carlo method, $\nabla_{\mathbf{u}} \mathcal{R}_{\varepsilon,N}$ and $\nabla^2_{\mathbf{uu}} \mathcal{R}_{\varepsilon,N}$ have the same form as (2.10) and (2.11), respectively, with $w_j = 1/N$.

2.3. Linear-Quadratic problem. Practical expressions are more convenient to write assuming linear constraints and quadratic costs. Let us assume that the spaces $\mathcal{Y}, \mathcal{Z}$ and $\mathcal{U}$ are Bochner spaces $\mathcal{Y} = L^2_\rho(\Xi; Y)$, $\mathcal{Z} = L^2_\rho(\Xi; Z)$, and $\mathcal{U} = L^2_\rho(\Xi; U)$, where Y, Z and U are Hilbert spaces of functions of x only (for example, $L^2(D)$), and L^2_ρ is a space of functions of ξ with the L^2 inner product weighted with the probability density function $\rho(\xi)$. Now we can assume that

$$J(y,\xi) = \frac{1}{2}\|y - y_d(x,\xi)\|^2_{M_y}, \quad S(u) = \frac{\alpha}{2}\|u\|^2_{M_u}, \quad c(y,u,\xi) = A(\xi)y - B(\xi)u - f(\xi),$$

where $M_y : Y \to Y$ is a self-adjoint positive semi-definite operator, $y_d(x,\xi) \in \mathcal{Y}$ is a desired state, $M_u : U \to U$ is a self-adjoint positive definite operator, $\|f\|_M := \sqrt{\langle f, Mf\rangle_{L^2(D)}}$ is the induced M-norm, $\alpha \geq 0$ is a regularization parameter, $A(\xi) : Y \to Z$ for any $\xi \in \Xi$ is a linear PDE operator on the state, $B(\xi) : U \to Z$ is a linear actuator operator of the control, and $f(\xi) \in Z$ is an uncontrollable right hand side of the PDE. In this case,

$$\begin{aligned} \nabla_y J(y_h,\xi) &= M_y \left(y_h(x,\xi) - y_d(x,\xi)\right), \\ \nabla_u S(u_h) &= \alpha M_u u_h(x,\xi), \\ \nabla_y c &= A(\xi), \\ \nabla_u c &= -B(\xi). \end{aligned}$$

To assemble (2.3)–(2.5), we introduce the mass matrices with elements

$$(\mathbf{M}_y)_{i,i'} = \int_D \psi_i(x) M_y \psi_{i'}(x)dx, \qquad (\mathbf{M}_u)_{i,i'} = \int_D \psi_i(x) M_u \psi_{i'}(x)dx,$$

the desired state vector $(\mathbf{y}_d(\xi))_i = \int_D \psi_i(x) y_d(x,\xi)dx$, the stiffness matrices with elements

$$(\mathbf{A}(\xi))_{i,i'} = \int_D \psi_i(x) A(\xi) \psi_{i'}(x)dx, \qquad (\mathbf{B}(\xi))_{i,i'} = \int_D \psi_i(x) B(\xi) \psi_{i'}(x)dx,$$

and the right hand side $(\mathbf{f}(\xi))_i = \int_D \psi_i(x) f(x,\xi) dx$. Replacing the integral over ξ by the quadrature, we can write the fully discrete Lagrangian gradient

$$(\nabla_{\mathbf{y}} \mathcal{L}_h)_{i,j} = w_j \sum_{i'=1}^{\hat{N}} [(\mathbf{M}_y)_{i,i'} y_{i',j} - \lambda_{i',j} (\mathbf{A}(\xi_j))_{i',i}] - w_j (\mathbf{y}_d(\xi_j))_i, \tag{2.12}$$

$$(\nabla_{\mathbf{u}} \mathcal{L}_h)_{i,j} = w_j \sum_{i'=1}^{\hat{N}} [\alpha (\mathbf{M}_u)_{i,i'} u_{i',j} + \lambda_{i',j} (\mathbf{B}(\xi_j))_{i',i}] + (\nabla_{\mathbf{u}} \mathcal{R}_{\varepsilon,h})_{i,j}, \tag{2.13}$$

$$(\nabla_{\boldsymbol{\lambda}} \mathcal{L}_h)_{i,j} = w_j \sum_{i'=1}^{\hat{N}} [-(\mathbf{A}(\xi_j))_{i,i'} y_{i',j} + (\mathbf{B}(\xi_j))_{i,i'} u_{i',j} + (\mathbf{f}(\xi_j))_i] \tag{2.14}$$

to be used for the TT method. The expressions for the Monte Carlo method are the same replacing w_j by $1/N$ and $\mathcal{R}_{\varepsilon,h}$ by $\mathcal{R}_{\varepsilon,N}$. To write the Hessian in a more convenient matrix form, let

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{A}(\xi_1) & & \\ & \ddots & \\ & & \mathbf{A}(\xi_N) \end{bmatrix}, \qquad \boldsymbol{B} = \begin{bmatrix} \mathbf{B}(\xi_1) & & \\ & \ddots & \\ & & \mathbf{B}(\xi_N) \end{bmatrix},$$

$$\boldsymbol{W} = \begin{bmatrix} w_1 & & \\ & \ddots & \\ & & w_N \end{bmatrix} \otimes I_{\hat{N}}, \qquad \boldsymbol{f} = \begin{bmatrix} \mathbf{f}(\xi_1) \\ \vdots \\ \mathbf{f}(\xi_N) \end{bmatrix}, \quad \boldsymbol{y}_d = \begin{bmatrix} \mathbf{y}_d(\xi_1) \\ \vdots \\ \mathbf{y}_d(\xi_N) \end{bmatrix},$$

where $I_m \in \mathbb{R}^{m \times m}$ is the identity matrix, and $\otimes$ is the Kronecker product, then

$$\nabla^2 \mathcal{L}_h = \begin{bmatrix} \boldsymbol{W}(I_N \otimes \mathbf{M}_y) & 0 & -\boldsymbol{W}\boldsymbol{A}^\top \\ 0 & \boldsymbol{W}(\alpha I_N \otimes \mathbf{M}_u) + \nabla^2_{\mathbf{uu}} \mathcal{R}_{\varepsilon,h} & \boldsymbol{W}\boldsymbol{B}^\top \\ -\boldsymbol{W}\boldsymbol{A} & \boldsymbol{W}\boldsymbol{B} & 0 \end{bmatrix},$$

with the familiar replacements for the Monte Carlo Hessian $\nabla^2 \mathcal{L}_N$.

2.4. Approximate block-diagonal Hessian and Newton's method. All terms in the Hessian above are block-diagonal with respect to j except $\nabla^2_{\mathbf{uu}} \mathcal{R}_{\varepsilon,h}$. In turn, only the second term in (2.11) breaks this block-diagonality. Moreover, while both terms of (2.11) have the same asymptotic scaling in u ($\mathcal{O}(1/|u|)$ for $|u_{i,j}| \geq |u| \gg \varepsilon$ and $\mathcal{O}(1/\varepsilon)$ for $|u_{i,j}| \ll \varepsilon$), the second term has an additional factor $w_{j'} < 1$, since $w_j > 0$ and $\sum_j w_j = 1$ for the probability normalization. In fact, $w_{j'} = 1/N$ for the Monte Carlo method, and $w_{j'} \leq C/N \ll 1$ for the TT method, since typically $N \gg 1$ for both TT and Monte Carlo. Thus, we omit the second term of (2.11), and approximate $\nabla^2_{\mathbf{uu}} \mathcal{R}_{\varepsilon,h}$ by $\tilde{\nabla}^2_{\mathbf{uu}} \mathcal{R}_{\varepsilon,h}$ whose components are given by

$$(\tilde{\nabla}^2_{\mathbf{uu}} \mathcal{R}_{\varepsilon,h})_{i,j,i',j'} = \frac{\beta \hat{w}_i \delta_{j,j'} \delta_{i,i'}}{\sqrt{\sum_{j''=1}^{N} \left[w_{j''} u^2_{i,j''} \right] + \varepsilon^2}}, \tag{2.15}$$

$$\tilde{\nabla}^2 \mathcal{L}_h = (I_3 \otimes \boldsymbol{W}) \begin{bmatrix} I_N \otimes \mathbf{M}_y & 0 & -\boldsymbol{A}^\top \\ 0 & \alpha I_N \otimes \mathbf{M}_u + \tilde{\nabla}^2_{\mathbf{uu}} \mathcal{R}_{\varepsilon,h} & \boldsymbol{B}^\top \\ -\boldsymbol{A} & \boldsymbol{B} & 0 \end{bmatrix}. \tag{2.16}$$

Note that now all terms contain w_j in front of them, so we can take $\boldsymbol{W}$ out of the Hessian, which will cancel with $\boldsymbol{W}$ in front of the gradient

$$\nabla_{\mathbf{y}}\mathcal{L}_h = \boldsymbol{W}\left[(I_N \otimes \mathbf{M}_y)\mathbf{y} - \boldsymbol{A}^\top\boldsymbol{\lambda} - \boldsymbol{y}_d\right], \tag{2.17}$$

$$\nabla_{\mathbf{u}}\mathcal{L}_h = \boldsymbol{W}\left[(\alpha I_N \otimes \mathbf{M}_u)\mathbf{u} + \boldsymbol{B}^\top\boldsymbol{\lambda} + \tilde{\nabla}_{\mathbf{u}}\mathcal{R}_{\varepsilon,h}\right], \tag{2.18}$$

$$\nabla_{\boldsymbol{\lambda}}\mathcal{L}_h = \boldsymbol{W}\left[-\boldsymbol{A}\mathbf{y} + \boldsymbol{B}\mathbf{u} + \boldsymbol{f}\right], \tag{2.19}$$

where a gradient without w_j reads

$$(\tilde{\nabla}_{\mathbf{u}}\mathcal{R}_{\varepsilon,h})_{i,j} = \beta\hat{w}_i \frac{u_{i,j}}{\sqrt{\sum_{j'=1}^N \left[w_{j'}u_{i,j'}^2\right] + \varepsilon^2}}.$$

This allows us to write the Newton's method as summarized in Algorithm 2.1. Note that we rewrite the update as a linear equation on the next iterate directly. This will be beneficial for the TT computations in Section 3.2. This also uses the step size of 1. For globalization, one can rewrite Steps 3–4 in the usual increment form, followed by a line search, although the TT ranks of the increment $\mathbf{s}^{[k+1]} - \mathbf{s}^{[k]}$ can be larger than the TT ranks of $\mathbf{s}^{[k]}$ or $\mathbf{s}^{[k+1]}$ alone. The linear solution (Step 4) can be implemented in the TT format in a problem-dependent way. One example is shown in the next section.

Algorithm 2.1 Approximate Newton's method for the shared-sparsity optimization

Require: Initial guess $\mathbf{y}^{[0]}, \mathbf{u}^{[0]}, \boldsymbol{\lambda}^{[0]}$, stopping tolerance $\mathrm{tol} > 0$, maximum number of iterations K.
Ensure: Updated iterations $\mathbf{y}^{[k]}, \mathbf{u}^{[k]}, \boldsymbol{\lambda}^{[k]}$.

1: **for** $k = 0, 1, \ldots, K-1$ **do**
2: Let $\mathbf{s}^{[k]} = \left[(\mathbf{y}^{[k]})^\top, (\mathbf{u}^{[k]})^\top, (\boldsymbol{\lambda}^{[k]})^\top\right]^\top$ be the previous solution vector.
3: Assemble $\tilde{\nabla}^2\mathcal{L}_h$ and $\mathbf{b} := \tilde{\nabla}^2\mathcal{L}_h\mathbf{s}^{[k]} - \nabla\mathcal{L}_h$ with $\mathbf{y} = \mathbf{y}^{[k]}$, $\mathbf{u} = \mathbf{u}^{[k]}$ and $\boldsymbol{\lambda} = \boldsymbol{\lambda}^{[k]}$ as shown in (2.16) and (2.17)–(2.19).
4: Solve $\tilde{\nabla}^2\mathcal{L}_h\mathbf{s}^{[k+1]} = \mathbf{b}$.
5: Split the components $\left[(\mathbf{y}^{[k+1]})^\top, (\mathbf{u}^{[k+1]})^\top, (\boldsymbol{\lambda}^{[k+1]})^\top\right]^\top = \mathbf{s}^{[k+1]}$.
6: **if** $\|\mathbf{s}^{[k+1]} - \mathbf{s}^{[k]}\| \le \mathrm{tol} \cdot \|\mathbf{s}^{[k+1]}\|$ **then**
7: **break**.
8: **end if**
9: **end for**

2.5. Reduced space formulation. Assuming that for every $u \in \mathcal{U}_{ad}$ there is a unique solution $y(u; x, \xi) \in \mathcal{Y}$ to (1.1), we can introduce a control-to-state map $\mathcal{U}_{ad} \ni u(x,\xi) \mapsto \Upsilon(u)(x,\xi) := y(u; x, \xi)$, and the reduced version of the cost (1.2):

$$\min_{u\in\mathcal{U}_{ad}} j(u), \quad j(u) := \mathbb{E}[J(\Upsilon(u), \xi)] + \mathbb{E}[S(u)] + \mathcal{R}_\varepsilon(u). \tag{2.20}$$

The reduced cost and its derivatives can be discretized using the basis or Monte Carlo methods from above, and the quadrature weights w_j or $1/N$ can be cancelled in the Hessian and gradient.

Existence of the solution to (2.20) and convergence of the smoothed approximation was established in [30] for linear constraints.

PROPOSITION 2.1 ([30], Theorem 4 and Corollary 5). *There exists a unique solution to*

$$\min_{u\in\mathcal{U}_{ad}} \int_D \left[\int_\Xi (A^{-1}(Bu + f) - y_d)^2 + \alpha u^2\right] dx\rho(\xi)d\xi + \mathcal{R}_\varepsilon(u), \tag{2.21}$$

if $A : \mathcal{Y} \subset H^1(D) \to H^{-1}(D)$ *is linear and invertible,* $y_d \in \mathcal{Y}$*,* $\alpha > 0$*, for any* $\beta \geq 0$ *and* $\varepsilon \geq 0$*.*

PROPOSITION 2.2 ([30], Lemma 6). *Let* u_ε *be the solution to* (2.21) *with the smoothing parameter* $\varepsilon \geq 0$*, and let* u_0 *be the solution to* (2.21) *with* $\varepsilon = 0$*. If* D *is bounded, then*

$$\|u_\varepsilon - u_0\|^2_{L^2_\rho(\Xi; L^2(D))} \leq \varepsilon\beta\alpha^{-1}|D|.$$

This implies the convergence of the smoothed approximation with the rate of $\mathcal{O}(\varepsilon^{1/2})$. However, in the numerical experiments with the linear elliptic PDE we observe a faster rate of $\mathcal{O}(\varepsilon)$.

3. Functional Tensor Train approximation. In the course of optimization iterations we need to compute expectations of random vectors. Traditional Monte Carlo method may suffer from slow convergence. In this work we propose to approximate random vectors by the Functional Tensor Train depending on ξ.

We start with a Cartesian product discretization. We assume that the parametrizing random variables $\xi_1, \ldots, \xi_d$ are independent, so their probability density function factorizes, $\rho(\xi) = \rho_1(\xi_1) \cdots \rho_d(\xi_d)$. For each ξ_k, $k = 1, \ldots, d$, we introduce a Gauss quadrature weighted with ρ_k, with n nodes $\{\xi_k^j\}_{j=1}^n$ and weights $\{w_k^j\}_{j=1}^n$. The expectation of any scalar continuous function is approximated by the combined quadrature

$$\mathbb{E}\left[f(\xi)\right] \approx \sum_{j_1=1}^{n} \cdots \sum_{j_d=1}^{n} w_1^{j_1} \cdots w_d^{j_d} f(\xi_1^{j_1}, \ldots, \xi_d^{j_d}). \tag{3.1}$$

The convergence rate of this quadrature is bound by the convergence rates of individual quadratures, and, if f is sufficiently smooth, they all can be much faster (e.g. exponential) than the Monte Carlo convergence rate. However, direct application of this quadrature suffers from the *curse of dimensionality*: it involves the computation of n^d evaluations of $f(\xi)$, which quickly becomes infeasible already for a moderate d.

3.1. Tensor Train decomposition. The Tensor Train (TT) decomposition [34] alleviates this problem by approximating the *tensor* of coefficients

$$F_{j_1,\ldots,j_d} = f(\xi_1^{j_1}, \ldots, \xi_d^{j_d}), \quad j_k = 1, \ldots, n, \quad k = 1, \ldots, d,$$

by the following expansion:

$$F_{j_1,\ldots,j_d} \approx \tilde{F}_{j_1,\ldots,j_d} := \sum_{s_0,\ldots,s_d=1}^{r_0,\ldots,r_d} F^{(1)}_{s_0,j_1,s_1} F^{(2)}_{s_1,j_2,s_2} \cdots F^{(d)}_{s_{d-1},j_d,s_d}, \tag{3.2}$$

where the factors $F^{(k)}$ are called *TT cores*, and the ranges r_k of the new summation indices are called *TT ranks*, $r_k(\tilde{F})$ if we want to emphasize that these are TT ranks of the TT decomposition $\tilde{F}$. For convenience, we introduce the maximal TT rank $r := \max_{k=0,\ldots,d} r_k$, and, unless we address individual TT ranks, we will omit the word "maximal", and refer to r as the *TT rank* $r(\tilde{F})$ of a TT decomposition $\tilde{F}$. Note that the TT cores contain $\mathcal{O}(dnr^2)$ elements, which can be much fewer than n^d elements in F, if the TT rank r is small. For example, the probability density function of independent random variables sampled on a Cartesian grid factorizes into a rank-1 TT decomposition. In general, we can always let $r_0 = r_d = 1$. Other TT ranks can be nontrivial. Nonetheless, there exist broad classes of smooth or weakly correlated functions that yield low TT ranks, such as polylogarithmic in the approximation error [36, 38, 22].

Our main usage of the structure (3.2) is the fast computation of the quadrature (3.1). Indeed, we can start with summing up just the last TT core,

$$E^{(d)}_{s_{d-1},s_d} = \sum_{j=1}^{n} w_d^j F^{(d)}_{s_{d-1},j,s_d}, \qquad s_{d-1} = 1, \ldots, r_{d-1}, \quad s_d = 1, \ldots, r_d.$$

Now assume we have a matrix $E^{(k+1)} \in \mathbb{R}^{r_k \times r_d}$. To implement the sums over both i_k and s_k we sum the kth TT core weighted with w_k, and multiply the resulting matrix with $E^{(k+1)}$,

$$E^{(k)}_{s_{k-1},s_d} = \sum_{s_k=1}^{r_k} \left(\sum_{j=1}^{n} w_k^j F^{(k)}_{s_{k-1},j,s_k} \right) E^{(k+1)}_{s_k,s_d}, \qquad s_{k-1} = 1, \ldots, r_{k-1}, \quad s_d = 1, \ldots, r_d.$$

Continuing recursively (and recalling that $r_0 = r_d = 1$), we obtain the approximate expectation

$$\tilde{\mathbb{E}}\,[f] := \sum_{j_1=1}^{n} \cdots \sum_{j_d=1}^{n} w_1^{j_1} \cdots w_d^{j_d} \tilde{F}_{j_1,\ldots,j_d} = E^{(1)}.$$

Counting the summation sizes, we see that the complexity of the expectation computed this way is $\mathcal{O}(dnr^2)$ similarly to the number of unknowns in a TT decomposition.

However, we need to approximate *vector* functions of spatial discretization coefficients such as $\mathbf{u}(\xi)$, or directly the tensor of fully discrete expansion coefficients (2.1), where j is replaced by $j_1, \ldots, j_d$ of the Cartesian quadrature, $u_{i,j} = U_{i,j_1,\ldots,j_d}$. Since we want to keep the spatial discretization and solver black box, we employ the so-called *block TT* decomposition [15], where instead of summing over s_0 we let it enumerate the spatial index i,

$$U_{i,j_1,\ldots,j_d} \approx \tilde{U}_{i,j_1,\ldots,j_d} := \sum_{s_1,\ldots,s_d=1}^{r_1,\ldots,r_d} \hat{U}^{(1)}_{i,j_1,s_1} U^{(2)}_{s_1,j_2,s_2} \cdots U^{(d)}_{s_{d-1},j_d,s_d}. \tag{3.3}$$

In addition to computing expectations, the TT decomposition admits fast interpolation too. Interpolating basis functions in space are assumed to be known from the finite element discretization. In ξ, we introduce Lagrange interpolation polynomials $\phi^k_{j_k}(\xi_k)$ centered at the quadrature nodes $\xi_k^{j_k}$. Now, we just need to sum the basis functions interpolated at each given x and ξ with only one TT core at a time,

$$\hat{u}^{(1)}_{s_1}(x, \xi_1) := \sum_{j_1=1}^{n} \sum_{i=1}^{N} \psi_i(x) \hat{U}^{(1)}_{i,j_1,s_1} \phi^1_{j_1}(\xi_1), \quad u^{(k)}_{s_{k-1},s_k}(\xi_k) := \sum_{j_k=1}^{n} U^{(k)}_{s_{k-1},j_k,s_k} \phi^k_{j_k}(\xi_k),$$

followed by the matrix multiplication

$$u_h(x, \xi) = \hat{u}^{(1)}(x, \xi_1) u^{(2)}(\xi_2) \cdots u^{(d)}(\xi_d) \tag{3.4}$$

calculated by a recursion similar to that used to compute expectations with $\mathcal{O}(dnr^2)$ complexity. The expansion (3.4) generalizing the TT structure to continuous variables was called the Functional Tensor Train decomposition [10, 20]. With a slight abuse of notations we can write $r(u_h)$ and $r_k(u_h)$ for the maximal/individual TT ranks of $\tilde{U}$. Similarly, we can consider a semi-discrete TT decomposition

$$\mathbf{u}(\xi) = \mathbf{u}^{(1)}(\xi_1) u^{(2)}(\xi_2) \cdots u^{(d)}(\xi_d),$$

which is naturally extensible to the solution in Algorithm 2.1,

$$\begin{bmatrix} \mathbf{y}(\xi) \\ \mathbf{u}(\xi) \\ \boldsymbol{\lambda}(\xi) \end{bmatrix} = \begin{bmatrix} \mathbf{y}^{(1)}(\xi_1) \\ \mathbf{u}^{(1)}(\xi_1) \\ \boldsymbol{\lambda}^{(1)}(\xi_1) \end{bmatrix} u^{(2)}(\xi_2) \cdots u^{(d)}(\xi_d). \tag{3.5}$$

3.2. TT-Cross for the approximate Newton solution. The family of TT *Cross* approximation algorithms [35, 37, 13, 14] computes a TT decomposition from $\mathcal{O}(dnr^2)$ evaluations of the function $f(\xi)$ at adaptively chosen points ξ. The first key ingredient is the *Alternating* iteration, similar to the methods of

Alternating Least Squares [24] and Density Matrix Renormalization Group [43, 39]. Instead of computing the elements of all TT cores simultaneously, we fix all but one cores, and restrict the relevant problem (such as Least Squares) to the elements of only one TT core at a time. In the next step, we freeze the TT core we have just computed, and release the next one for the update, and so on. A motivation for this alternation over the subsets of unknowns comes from the fact that the tensor $\tilde{F}$ represented by its TT decomposition (3.2) is *linear* in the elements of one TT core at a time, and hence for example a convex optimization problem on the elements of the whole tensor $\tilde{F}$ remains a convex optimization problem on the elements of one TT core $F^{(k)}$. In contrast, the optimization of all TT cores collectively is usually non-convex even if the initial cost function of $\tilde{F}$ was quadratic.

To compute TT cores we can use interpolation rather than optimization:

$$\tilde{F}_{j_1,\ldots,j_d} = F_{j_1,\ldots,j_d} \quad \forall (j_1,\ldots,j_d) \in \mathbf{J}_k := \{(j_1^t,\ldots,j_d^t) \in \mathbb{R}^d : t = 1,\ldots,r_{k-1}nr_k\}, \tag{3.6}$$

$\mathbf{J}_k$ is a set of $r_{k-1}nr_k$ tensor indices. Note that $F^{(k)}$ contains the same number of unknowns. In fact, (3.6) is a linear equation on the elements of $F^{(k)}$. Indeed, let

$$F^{<k}_{t,s_{k-1}} = \sum_{s_0,\ldots,s_{k-2}=1}^{r_0,\ldots,r_{k-2}} F^{(1)}_{s_0,j_1^t,s_1} \cdots F^{(k-1)}_{s_{k-2},j_{k-1}^t,s_{k-1}}, \tag{3.7}$$

$$F^{>k}_{t,s_k} = \sum_{s_{k+1},\ldots,s_d=1}^{r_{k+1},\ldots,r_d} F^{(k+1)}_{s_k,j_{k+1}^t,s_{k+1}} \cdots F^{(d)}_{s_{d-1},j_d^t,s_d}, \tag{3.8}$$

$$F^{\neq k}_{t,s_{k-1}j_k s_k} = F^{<k}_{t,s_{k-1}} \cdot \delta_{j_k,j_k^t} \cdot F^{>k}_{t,s_k}, \tag{3.9}$$

where $\delta_{i,j} = 1$ when $i = j$ and 0 otherwise. Once can check the linear relation

$$\tilde{F}_{j_1^t,\ldots,j_d^t} = \sum_{s_{k-1},j_k,s_k} F^{\neq k}_{t,s_{k-1}j_k s_k} F^{(k)}_{s_{k-1},j_k,s_k}, \tag{3.10}$$

or, stretching $\tilde{F}$ and $F^{(k)}$ into vectors, $\mathrm{vec}(\tilde{F}(\mathbf{J}_k)) = F^{\neq k}\mathrm{vec}(F^{(k)})$.

For efficient computation of (3.7) and (3.8) and index selection in the course of iterations over $k = 1,\ldots,d$, we restrict the structure of $\mathbf{J}_k$ to a Cartesian form

$$\mathbf{J}_k = J_{<k} \times [n] \times J_{>k}, \tag{3.11}$$

where

$$[n] = \{1,\ldots,n\}, \qquad J_{<k} = \{(j_1^t,\ldots,j_{k-1}^t)\}_{t=1}^{r_{k-1}}, \qquad J_{>k} = \{(j_{k+1}^t,\ldots,j_d^t)\}_{t=1}^{r_k}.$$

Now $F^{<k} \in \mathbb{R}^{r_{k-1}\times r_{k-1}}$ and $F^{>k} \in \mathbb{R}^{r_k \times r_k}$ can be constructed only from $J_{<k}$ and $J_{>k}$, respectively. Moreover, we can construct the next sets

$$J_{<k+1} \subset J_{<k} \times [n], \qquad J_{>k-1} \subset [n] \times J_{>k} \tag{3.12}$$

for the next iteration for $F^{(k+1)}$ or $F^{(k-1)}$ by selecting optimal rows from small matrices $\overline{F}^{\leq k} \in \mathbb{R}^{r_{k-1}n\times r_k}$ and $\overline{F}^{\geq k} \in \mathbb{R}^{nr_k \times r_{k-1}}$ with elements

$$\overline{F}^{\leq k}_{s_{k-1}j_k,s_k} = \sum_{t_{k-1}=1}^{r_{k-1}} F^{<k}_{s_{k-1},t_{k-1}} F^{(k)}_{t_{k-1},j_k,s_k} \quad \text{and} \tag{3.13}$$

$$\overline{F}^{\geq k}_{j_k s_k,s_{k-1}} = \sum_{t_k=1}^{r_k} F^{(k)}_{s_{k-1},j_k,t_k} F^{>k}_{t_k,s_k}. \tag{3.14}$$

The set $P \subset \{1, \dots, r_{k-1}n\}$ of indices of rows extracted from $\overline{F}^{\le k}$, or $P \subset \{1, \dots, nr_k\}$ for $\overline{F}^{\ge k}$, is chosen to reduce the condition number of the corresponding submatrix $F^{<k+1} = \overline{F}^{\le k}_{P,:}$ or $F^{>k-1} = (\overline{F}^{\ge k}_{P,:})^\top$, and hence the condition number of (3.9). We use the Maximum Volume (*maxvol*) algorithm [19] for this. Similarly to the submatrices $F^{<k+1}$ and $F^{>k-1}$, the next sets of tensor indices are constructed by taking pth tuples from (3.12) for $p \in P$,

$$J_{<k+1} = (J_{<k} \times [n])_P, \qquad J_{>k-1} = ([n] \times J_{>k})_P. \tag{3.15}$$

The basic TT-Cross algorithm described above generalizes elegantly [13] to the block TT decomposition (3.3) and the Newton iterate in Line 4 of Algorithm 2.1. The block TT decomposition can contain the special index i in any TT core,

$$\begin{aligned}\tilde{U}_{i,j_1,\dots,j_d} = \sum_{s_0,\dots,s_d=1}^{r_0,\dots,r_d} U^{(1)}_{s_0,j_1,s_1} \cdots U^{(k-1)}_{s_{k-2},j_{k-1},s_{k-1}} \cdot U^{(k)}_{s_{k-1},i,j_k,s_k} \\ \cdot U^{(k+1)}_{s_k,j_{k+1},s_{k+1}} \cdots U^{(d)}_{s_{d-1},j_d,s_d}.\end{aligned} \tag{3.16}$$

Note that all TT cores but $\hat{U}^{(k)}$ have the same shapes as generic TT cores in (3.2), and hence we can perform the majority of the TT-Cross computations (3.7)–(3.9), (3.11) and (3.15) for $U^{(k)}$ instead of $F^{(k)}$ verbatim. The linear equation (3.10) looks different but consistent:

$$U_{i,j_1^t,\dots,j_d^t} = \sum_{s_{k-1},j_k,s_k} U^{\ne k}_{t,s_{k-1}j_k s_k} \hat{U}^{(k)}_{s_{k-1},i,j_k,s_k}, \tag{3.17}$$

and solving it for all values of i gives all the elements of the block TT core $\hat{U}^{(k)}$. Now we can use the Principal Component Analysis to reduce $\hat{U}^{(k)}$ to a 3-dimensional TT core:

$$\hat{U}^{(k)}_{s_{k-1},i,j_k,s_k} \approx U^{(k)}_{s_{k-1},j_k,\tilde{s}_k} W_{\tilde{s}_k,s_k i} \qquad \text{for iterating } k \to k+1, \tag{3.18}$$

$$\hat{U}^{(k)}_{s_{k-1},i,j_k,s_k} \approx W_{s_{k-1}i,\tilde{s}_{k-1}} U^{(k)}_{\tilde{s}_{k-1},j_k,s_k} \qquad \text{for iterating } k \to k-1. \tag{3.19}$$

The remaining TT-Cross operations (3.13)–(3.14) can be applied consistently with thus obtained $U^{(k)}$. As a by-product, we can update the TT ranks r_k and r_{k-1} up to a desired truncation threshold in (3.18)–(3.19).

4. Numerical examples.

4.1. Elliptic PDE. In the first test we consider the following benchmark example, an adaptation of that in [23, 17]. We solve (1.2), where:

$$J(y,\xi) := \frac{1}{2}\|y(x,\xi) - y_d(x)\|^2_{L^2(D)}, \tag{4.1}$$

$$S(u) := \frac{\alpha}{2}\|u(x,\xi)\|^2_{L^2(D)}, \tag{4.2}$$

$$\text{s.t.} \quad \nu(\xi)\Delta y(x,\xi) = g(x,\xi) + u(x,\xi), \quad (x,\xi) \in D \otimes [-1,1]^d, \tag{4.3}$$

$$\nu(\xi) = 10^{\xi_1(\omega)-2}, \quad g(x,\xi) = \frac{\xi_2(\omega)}{100}, \tag{4.4}$$

and boundary conditions depending on two options for the physical domain D:

- $D = (0,1)$, in which case $d = 4$, with

$$y|_{x=0} = -1 - \frac{\xi_3(\omega)}{1000}, \qquad y|_{x=1} = -\frac{2+\xi_4(\omega)}{1000}$$

- $D = (0,1)^2$, in which case $d = 6$, and

$$
\begin{aligned}
y|_{x_1=0} &= b_1(\xi)(1-x_2) + b_2(\xi)x_2, & y|_{x_2=1} &= b_2(\xi)(1-x_1) + b_3(\xi)x_1 \\
y|_{x_1=1} &= b_4(\xi)(1-x_2) + b_3(\xi)x_2, & y|_{x_2=0} &= b_1(\xi)(1-x_1) + b_4(\xi)x_1, \\
b_1(\xi) &= -1 - \frac{\xi_3(\omega)}{1000}, & b_2(\xi) &= -\frac{2+\xi_4(\omega)}{1000}, \\
b_3(\xi) &= -1 - \frac{\xi_5(\omega)}{1000}, & b_4(\xi) &= -\frac{2+\xi_6(\omega)}{1000},
\end{aligned}
$$

the random vector $\xi(\omega) = (\xi_1(\omega), \ldots, \xi_d(\omega)) \sim \mathcal{U}(-1,1)^d$ is uniformly distributed (so $\rho(\xi) = 2^{-d}$), and the desired state $y_d(x) = -\sin(50x/\pi)$.

We start with the one-dimensional physical domain, $D = (0,1)$. The physical variable x is discretized with a uniform grid with step size $h = 1/1024$, and each of the random variables ξ_k is discretized with $n_\xi = 17$ Gauss-Legendre grid points. The first regularization parameter $\alpha = 10^{-2}$ is fixed to the value in the original benchmark. However, the shared sparsity regularization parameter β is varied to study its effect on the control. Firstly, we fix $\varepsilon = 10^{-5}$ and the TT approximation threshold 10^{-5}. The control for different β is shown in Figure 1. As we expected, larger β increase the fraction of the domain where both the mean and the variance of the solution are (nearly) zero, i.e. the sparsity is shared. The measure of the set where the absolute mean control is below 10^{-4}, the number of iterations of Algorithm 2.1 required to drive the relative increment between the consecutive iterations of the solution below 10^{-5}, and the misfit of the state are shown in Table 1. Clearly, larger β make the Hessian approximation less accurate, and require

TABLE 1

Elliptic example (1D), measure of sparsity, number of Newton iterations to stopping tolerance 10^{-5}, and state misfit for different β.

β	$\int_0^1 \mathbf{1}[\lvert\mathbb{E}[u_h]\rvert < 10^{-4}]dx$	#iterations	$\mathbb{E}\left[\lVert y_h - y_d\rVert^2_{L^2(D)}\right]$
0	0.000	2	0.0645
10^{-2}	0.108	75	0.0798
10^{-1}	0.575	341	0.1763
1	0.890	1370	0.4246

more iterations of the (inexact) Newton method. Unsurprisingly, $\beta = 0$ corresponding to the linear problem requires only one essential Newton iteration (the second iteration is just checking the convergence). The TT ranks during the TT-Cross algorithm remain between 4 and 6 for all iterations and β in the considered range. The TT ranks also stay in the same range as the TT truncation tolerance is varied between 10^{-3} and 10^{-8}. It is possible to rigorously establish these numerical observations.

PROPOSITION 4.1. *The solution $\left[\mathbf{y}^\top, \mathbf{u}^\top, \boldsymbol{\lambda}^\top\right]^\top$ of this 1D elliptic PDE in the final iteration of TT cross admits an exact TT decomposition of TT ranks not greater than 7.*

Proof. The proof is given in Appendix A. ∎

Note that the elementwise square $u_{i,j}^2$ within $\mathcal{R}_{\varepsilon,h}(\mathbf{u})$ can be computed exactly in the TT format, followed by taking the square root of deterministic coefficients only. In contrast, computing the exact L^1 norm would require to approximate the elementwise modulus $\mathbf{u}_+ = [|u_{i,j}|]$ in the TT format first, followed by the quadrature. However, since the modulus is non-smooth, the TT ranks of $\mathbf{u}_+$ reach 59 in this experiment, in contrast to at most 6 for $\mathbf{u}$.

Now we vary the smoothness parameter ε. We fix $\beta = 0.1$ and compute the optimal control u_ε for ε in the range between 10^{-6} and 10^{-1}, and the TT approximation tolerance of 10^{-8}. In Figure 2 (left), we show the difference between the full control, its mean and variance computed at the given ε compared to those

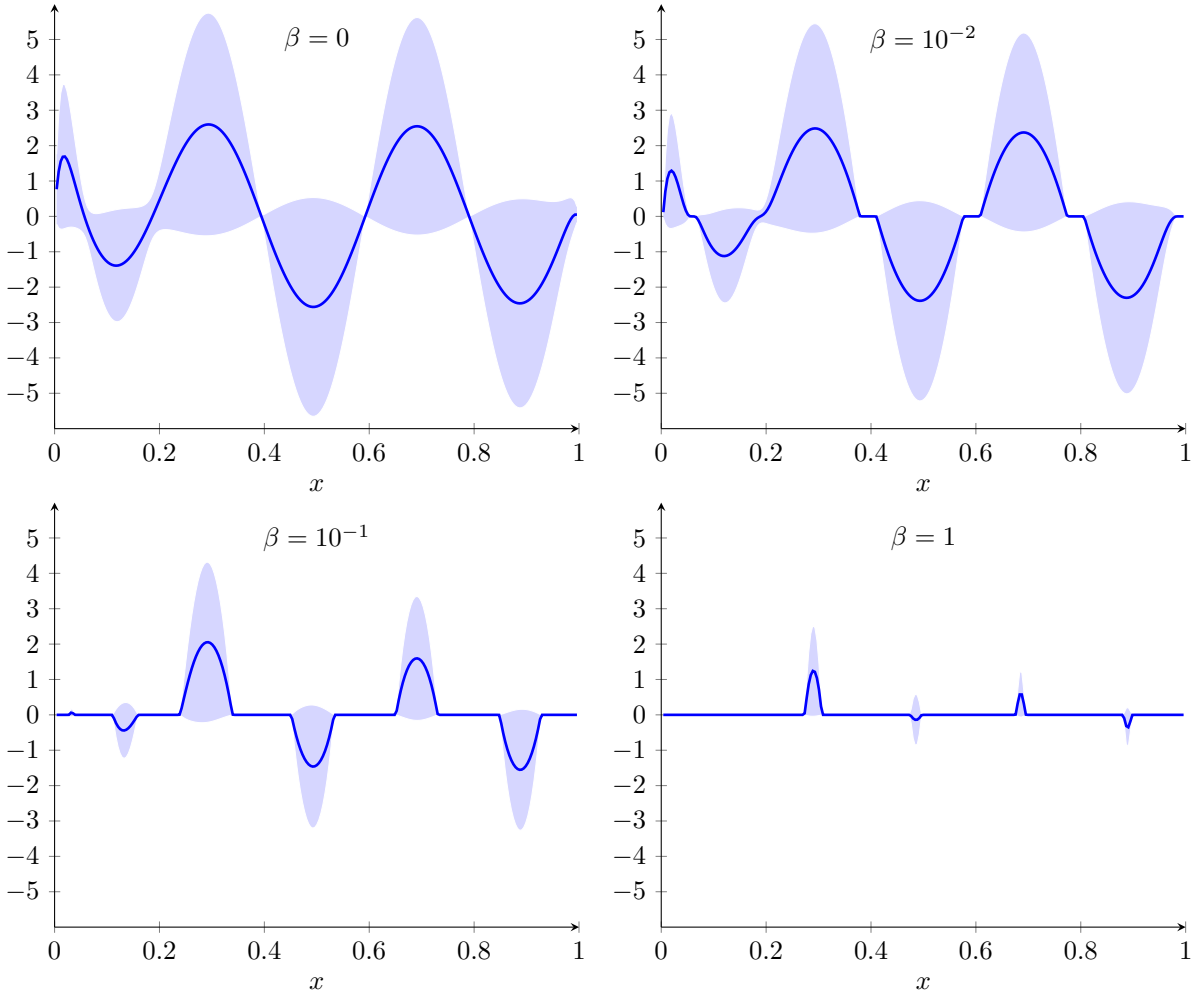


FIG. 1. *Elliptic PDE (1D), mean (solid lines) ± one standard deviation (shaded areas) of the optimal control $u_h(x, \xi)$ for different β.*

computed at $\varepsilon = 10^{-6}$, as well as a similar difference in the total cost (1.2). We see that the convergence is linear in ε for all outputs of the solution. The effect of a high ε is shown in Figure 2 (right). Essentially, ε guides the threshold below which the solution should be seen as zero.

From the left plot of Figure 2 we can also notice that the number of iterations grows asymptotically logarithmically with ε. In Figure 3 (left) we study this further. For each ε, we take the final mean control computed with the TT approximation and stopping tolerance of 10^{-8} as a reference solution $\mathbb{E}_*[u_\varepsilon]$, and use it to estimate the solution error at previous iterations. We see that eventually the method recovers the linear convergence, although the pre-asymptotic regime may take quite a few iterations for small ε.

In addition, in Figure 3 (right) we compare the TT approach with the Monte Carlo method in terms of the computational complexity, measured in the number of deterministic PDE solves. The Monte Carlo method uses the same approximate Newton iteration, but the expectations are approximated by a Monte Carlo quadrature instead of the TT approximation with a Gaussian quadrature. The TT method adapts the TT ranks (and hence the number of PDE solves needed during the TT-Cross algorithm) towards the stopping tolerance, so we just need to vary the latter. However, for the Monte Carlo method, the stopping

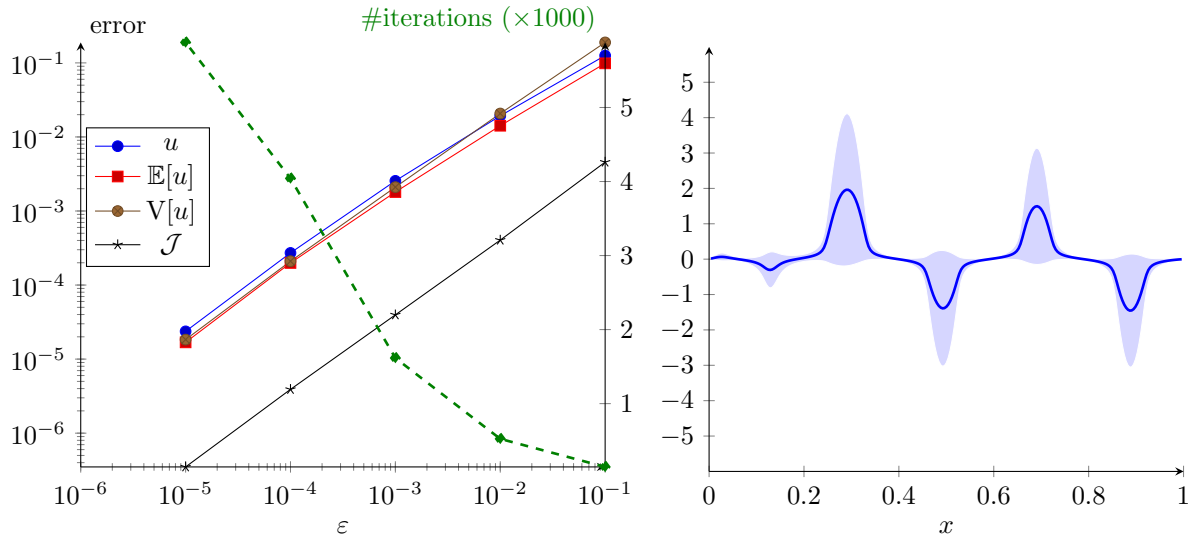


FIG. 2. *Elliptic PDE (1D). Left: errors in the full control* $\|u_\varepsilon - u_{10^{-6}}\|_{L^2_\rho(\Xi;L^2(D))}$, *mean control* $\|\mathbb{E}[u_\varepsilon] - \mathbb{E}[u_{10^{-6}}]\|_{L^2(D)}$, *variance of the control* $\|\mathrm{Var}[u_\varepsilon] - \mathrm{Var}[u_{10^{-6}}]\|_{L^2(D)}$, *and the total cost* $\mathcal{J}_\varepsilon - \mathcal{J}_{10^{-6}}$, *as well as the number of iterations till the stopping tolerance* 10^{-8}. *Right: mean (solid line)* $\pm$ *one standard deviation (shaded area) of the optimal control for* $\varepsilon = 10^{-1}$ *(right).*

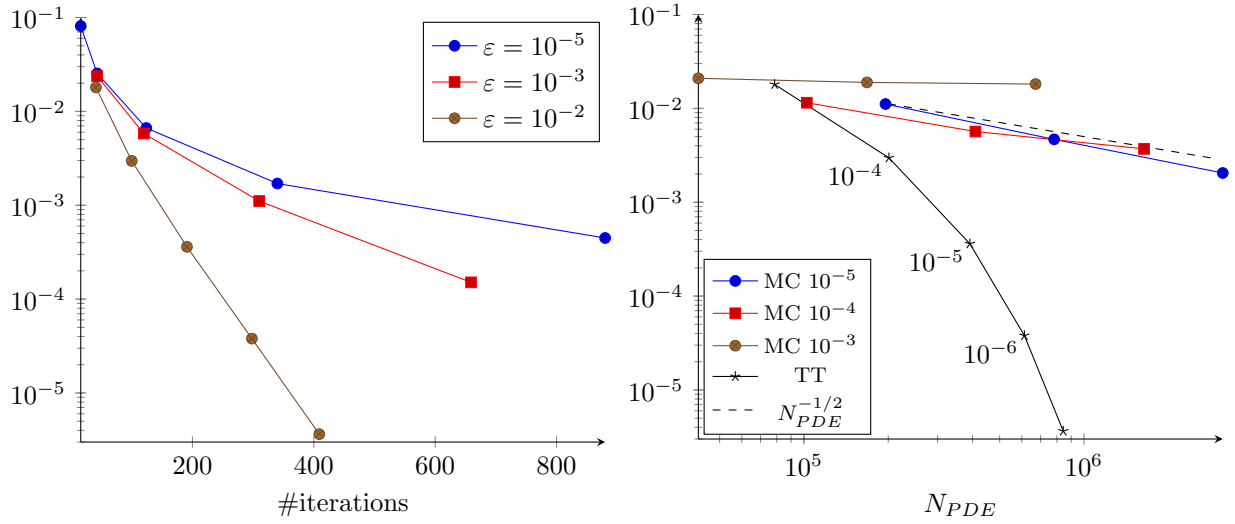


FIG. 3. *Elliptic PDE (1D), left: mean control error* $\|\tilde{\mathbb{E}}[u_\varepsilon] - \mathbb{E}_*[u_\varepsilon]\|_{L^2(D)}$ *in the TT method with Newton iterations for different smoothing parameters* ε. *Right:* $\|\tilde{\mathbb{E}}[u_\varepsilon] - \mathbb{E}_*[u_\varepsilon]\|_{L^2(D)}$ *in the Monte Carlo and TT methods with respect to the number of deterministic PDE solutions for* $\varepsilon = 10^{-2}$. *Numbers below points in the TT plot and in the legend of MC plots show the stopping tolerance. The reference solution* $\mathbb{E}_*[u_\varepsilon]$ *is computed with the TT method and tolerance* 10^{-8}.

tolerance and the number of Monte Carlo samples are independent parameters, so we vary them both. We see that the Monte Carlo method exhibits the usual $1/\sqrt{N}$ rate of convergence, much slower than the linear convergence of the TT method.

Figure 4 shows the results for the case when $D = (0,1)^2$, i.e., a 2-dimensional physical domain. A similar

behavior as in the 1-dimensional setting is observed.

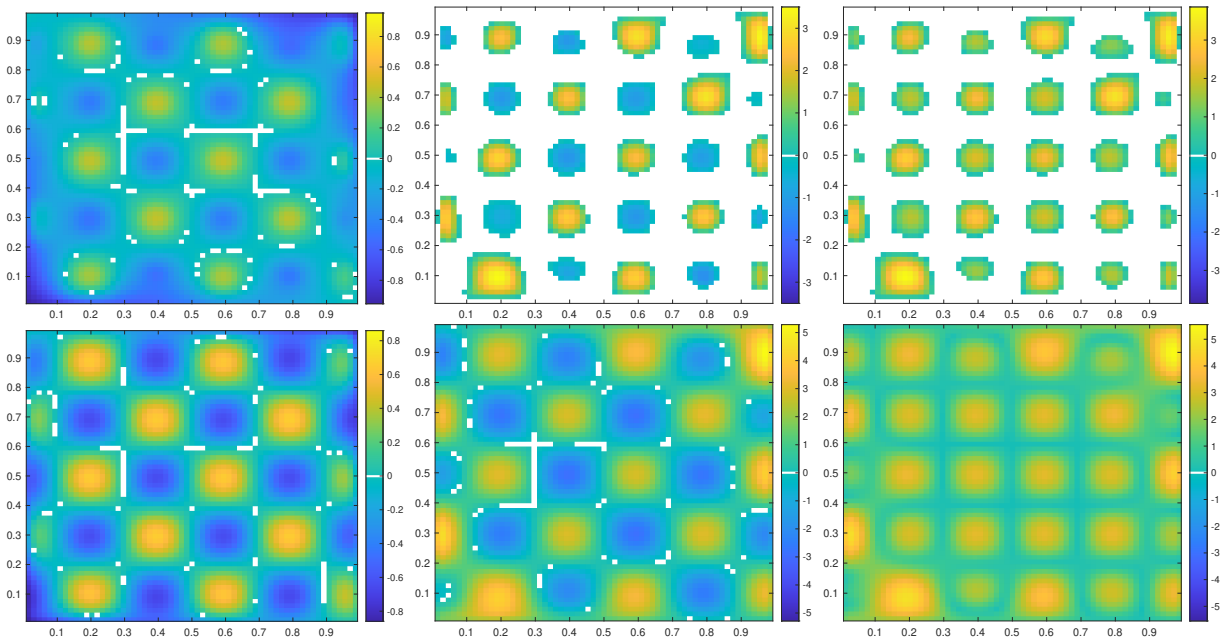


FIG. 4. *Elliptic PDE (2D) with* $h = 1/64$. *Left:* $\mathbb{E}[y_h]$. *Middle:* $\mathbb{E}[u_h]$. *Right:* $\mathrm{Std}[u_h]$. *Top:* $\beta = 0.1$ *and* $\varepsilon = 10^{-5}$, $\mathbb{E}\left[\|y_h - y_d\|^2_{L^2(D)}\right] = 0.1843$, $\int \mathbf{1}[|\mathbb{E}[u_h]| < 10^{-4}]dx = 0.681$. *Bottom:* $\beta = 0$, $\mathbb{E}\left[\|y_h - y_d\|^2_{L^2(D)}\right] = 0.0925$, $\int \mathbf{1}[|\mathbb{E}[u_h]| < 10^{-4}]dx = 0$.

4.2. Topology optimization. In this section we consider the topology optimization problem under uncertain Young's modulus. This problem aims to find an optimal distribution of the material of given relative volume $\bar{V}$ over the given domain D, which minimizes the compliance of the structure. The material distribution is encoded by the function of volume fraction $u(x, \xi) : D \times \Xi \to [0, 1]$, where 0 corresponds to the absence of the material, and 1 corresponds to the presence of the material (fractional values aid optimization and may also be seen as a reduced thickness of the material). After discretization of the physical domain D, the compliance optimization problem can be written as

$$\min_{\mathbf{u}} \ \mathbb{E}[C(\mathbf{u};\xi)], \ C(\mathbf{u};\xi) := \mathbf{f}^\top \mathbf{y}(\xi), \tag{4.5}$$

$$\text{s.t. } K(\mathbf{u};\xi)\mathbf{y}(\xi) = \mathbf{f},$$

$$\int_D u_h(x,\xi)dx = \bar{V} \cdot |D|, \qquad \text{a.s.} \tag{4.6}$$

$$0 \le u_h(x,\xi) \le 1, \qquad \text{a.s.} \tag{4.7}$$

where $\mathbf{u} \in \mathbb{R}^{\hat{N}N}$ is the vector of nodal values of the material volume fraction $u(x, \xi)$, $K(\mathbf{u};\xi) \in \mathbb{R}^{\hat{N}\times\hat{N}}$ is the stiffness matrix of the linear elasticity model

$$-\nabla \cdot (E(u;x,\xi)\nabla y(x,\xi)) = f(x)$$

subject to appropriate boundary conditions, $\mathbf{f} \in \mathbb{R}^{\hat{N}}$ is the force vector, $\mathbf{y}(\xi) \in \mathbb{R}^{\hat{N}}$ is the displacement vector, and $C(\mathbf{u};\xi) \in \mathbb{R}$ is the compliance. The net Young's modulus $E(u;x,\xi)$ depends on the material

distribution, and a random Young's modulus of the pure material [26],

$$E(u;x,\xi) = E_{\min} + \tilde{u}^3 \left(E_0 - E_{\min} + 0.02 \sum_{k=1}^{d} \sqrt{\lambda_k}\phi_k(x)\xi_k \right),$$

where E_0 is the mean Young's modulus of the pure material, perturbed by a truncated Karhunen-Loeve expansion (KLE) of a random field with uniformly distributed $\xi_k \sim \mathcal{U}(-1,1)$ and a Gaussian covariance function with the correlation length equal to the smaller side of the domain D, with eigenvalues λ_k sorted descending and corresponding eigenfunctions $\phi_k(x)$. Specifically, truncating the KLE at the total variance error of 1%, we obtain $d = 8$ random variables. $E_{\min} = 10^{-9}$ is the Young's modulus of the void (taken nonzero to avoid the singularity of the stiffness matrix), and

$$\tilde{u} = (I - \delta\Delta)^{-1} u$$

is the filtered material distribution, with Δ being the Laplace operator with natural boundary conditions, and $\delta > 0$ is a small parameter. We use $\delta = 0.1$ throughout the experiments. Similarly to [26, 41], we consider the MBB beam optimization problem where $D = [0, n_x] \times [0, n_y]$ with $n_x/n_y = 4$, such that $\hat{N} = n_x n_y = 4n_y^2$, whereas the mesh size is fixed to 1.

Optimization of (4.5) can be carried out with a projected gradient descent for each $\xi \in \Xi$. However, this would result in different optimal material distributions, which are difficult to implement in practice. To make the material distribution more stable with respect to random perturbations, we consider two penalties. Firstly, as in [41], we penalize the standard deviation of the compliance, $\mathrm{Std}[C(\mathbf{u};\xi)]$. Secondly, we consider the smoothed L^1-norm of the material distribution, $\mathcal{R}_{\varepsilon,h}(\mathbf{u})$, enforcing the sparsity (that is, absence) of the material shared across the random realizations. The smoothing parameter is fixed to $\varepsilon = 10^{-3}$. We introduce two regularization parameters $\kappa \geq 0$ and $\beta \geq 0$, controlling the weight of each penalty, so instead of (4.5) we consider the following problem

$$\min_{\mathbf{u}} \ \mathbb{E}[C(\mathbf{u};\xi)] + \kappa \cdot \mathrm{Std}[C(\mathbf{u};\xi)] + \mathcal{R}_{\varepsilon,h}(\mathbf{u}) \,. \tag{4.8}$$

The implementation is based on the `top88` Matlab code [3]. Specifically, we seek a TT approximation of the discretization coefficients of the filtered solution $\tilde{\mathbf{u}}$. Since the structure of the problem (4.5)–(4.7) is challenging for Newton's method, we resort to the simpler projected gradient descent method with a fixed step size $\tau > 0$. In its i-th iteration we use the block TT-Cross outlined in Section 3.2 to approximate directly the next iterate,

$$\begin{aligned} \mathbf{u}^{(i)} &= (I - \delta\Delta_h)\tilde{\mathbf{u}}^{(i)}, \\ \tilde{\mathbf{u}}^{(i+1)} &= (I - \delta\Delta_h)^{-1}\mathcal{P}\left(\mathbf{u}^{(i)} - \tau\nabla_{\mathbf{u}} \left[\mathbb{E}[C(\mathbf{u}^{(i)};\xi)] + \kappa \cdot \mathrm{Std}[C(\mathbf{u}^{(i)};\xi)] + \mathcal{R}_{\varepsilon,h}(\mathbf{u}^{(i)}) \right] \right), \end{aligned}$$

where Δ_h is the finite element discretization of the Laplace operator in the filter, and $\mathcal{P}$ is the projection onto the constraints (4.6)–(4.7). In each gradient descent iteration, we carry out 1 TT-Cross iteration initialized with the previous gradient descent iterate, truncating singular values of the block TT cores to the relative error threshold 10^{-2}. To ensure convergence we also do continuation in κ, increasing it gradually, by starting with $\kappa = 0$, and incrementing κ by $2 \cdot 10^{-3}$ each iteration until it reaches the desired value.

Numerical tests. For faster computations, we consider first a coarser grid of size 100×25. For this grid, we choose the gradient step size $\tau = 3 \cdot 10^{-3}$, which is about a good balance between speed and stability, and carry out 5000 iterations. For performance metrics, we consider the mean TT rank over iterations and cores,

$$\langle r \rangle = \frac{1}{5000} \sum_{i=1}^{5000} \frac{1}{d-1} \sum_{k=1}^{d-1} r_k(\tilde{\mathbf{u}}^{(i)}),$$

and the standard deviation of the thresholded material distribution $u_h(x,\xi) > 1/2$ and its spatial average,

$$\mathcal{S} := \int_D \mathrm{Std}[\mathbb{I}_{u_h(x,\xi)>1/2}]dx.$$

In Figures 5 and 6 we compare all four combinations of penalties, composed from two options for the standard deviation penalty ($\kappa = 0$ or $\kappa = 3$) and the shared sparsity penalty ($\beta = 0$ or $\beta = 1$). We see that the standard deviation penalty leads actually to a more complicated structure. In Figure 7 we show the evolution of TT ranks of the solution $\tilde{\mathbf{u}}$ as the iteration progresses. All penalties reduce the TT ranks due to reduced uncertainty, and the best performing methods are those with $\beta = 1$.

FIG. 5. *Mean optimized topology with different penalties. Left top: $\beta = \kappa = 0$ (no penalty). Right top: $\beta = 1$, $\kappa = 0$. Left bottom: $\beta = 0$, $\kappa = 3$. Right bottom: $\beta = 1$, $\kappa = 3$.*

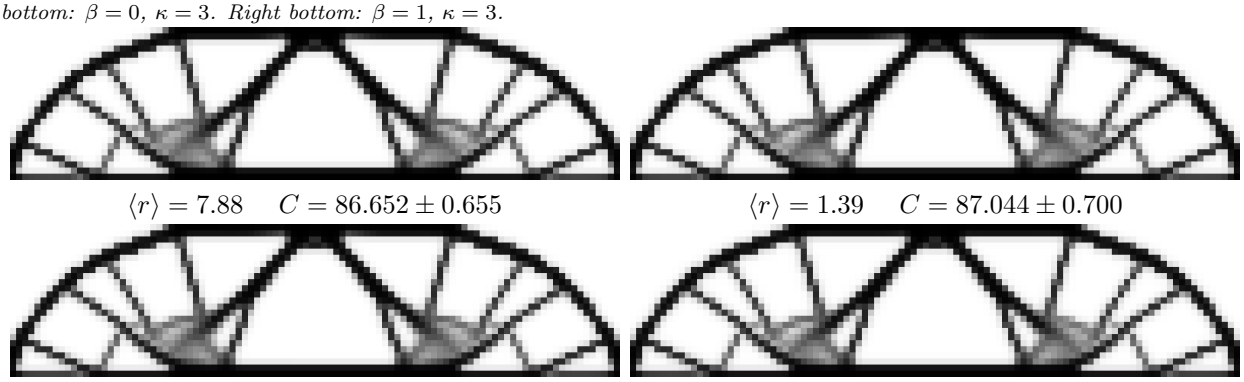
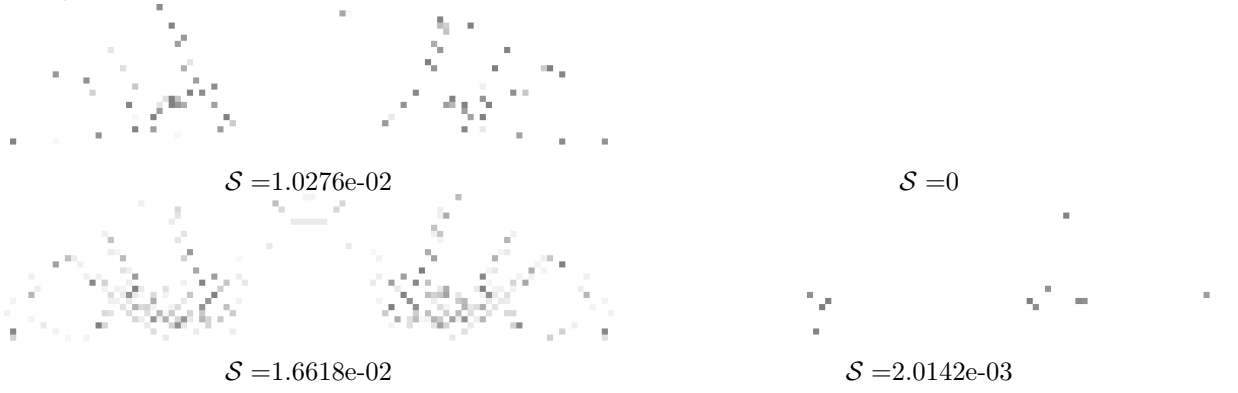
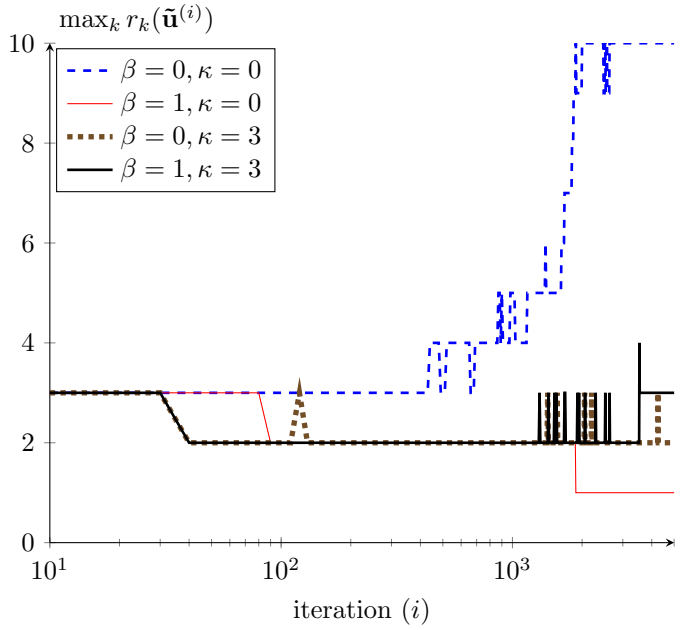


FIG. 6. *Standard deviation of the material distribution above 1/2 with different penalties. Left top: $\beta = \kappa = 0$ (no penalty). Right top: $\beta = 1$, $\kappa = 0$. Left bottom: $\beta = 0$, $\kappa = 3$. Right bottom: $\beta = 1$, $\kappa = 3$.*



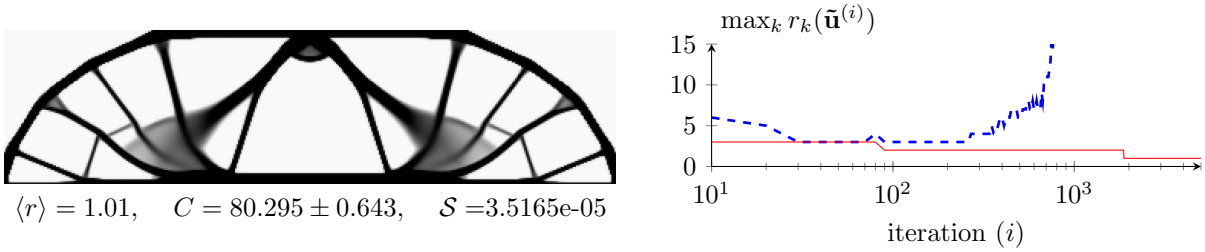
Finally, we carry out the topology optimization experiment on a fine spatial grid of size 400×100. We increase the gradient step size to $\tau = 5 \cdot 10^{-2}$ to achieve the solution increments similar to those in the previous coarse-grid experiment, which provides a well-converged solution after the same number of iterations of 5000. Since $\kappa > 0$ did not improve sparsity or performance, we test only $\kappa = 0$ here. For $\beta = 0$ (no penalty) the solution could not be completed due to the TT ranks exceeding 40 after 893 iterations, and the Matlab process crashing due to running out of 64Gb of memory. For $\beta = 1$ (enforcing shared sparsity), we managed to carry out 5000 iterations, reaching the compliance 80.295 ± 0.643 at the end of iterations and maximal TT rank 6 in the first few iterations, as shown in Figure 8 (right). The mean material distribution

FIG. 7. *Maximal TT ranks as a function of the iteration number*



is shown in Figure 8 (left). The standard deviation of the thresholded solution averages to 3.5165e-05, which would give again an almost blank figure similarly to Fig. 6 (top right). Again the penalty makes the material distribution more reproducible with respect to randomness.

FIG. 8. *Optimized topology on a fine* 400×100 *grid. Left:* $\mathbb{E}[\tilde{\mathbf{u}}(\xi)]$ *with* $\beta = 1$. *Right: maximal TT rank as a function of the iteration number (dashed:* $\beta = 0$*, solid:* $\beta = 1$*).*



5. Conclusion. We have developed both first and approximate second order methods for the PDE-constrained optimization with a smoothed shared sparsity penalty. For a nonzero smoothing parameter we obtain a linear convergence. The error depends linearly on the smoothing parameter as well. Smooth function approximations also converge sub-exponentially in the number of quadrature points in random parameters and TT ranks (exponentially if the function is analytic, and faster than any algebraic degree if the function is infinitely differentiable). This makes the overall rate of convergence sub-exponential in the total computational cost. This can be much faster than the algebraic rate of convergence of low-order and Monte Carlo methods, as we demonstrated in the benchmark elliptic PDE example. Moreover, a more structured solution with shared sparsity may actually exhibit lower TT ranks compared to the unconstrained solution. This opens the way for the shared sparsity optimization in real-life applications, such as topology.

At the moment, we observe a linear convergence in ε, which is faster than the theoretically predicted

rate of $\varepsilon^{-1/2}$. It remains to a future research to obtain a sharp estimate of the convergence rate.

Appendix A. TT decomposition of the solution of the 1D elliptic PDE. We are looking for the solution in the block TT format (3.5),

$$\begin{bmatrix}\mathbf{y}(\xi)\\ \mathbf{u}(\xi)\\ \boldsymbol{\lambda}(\xi)\end{bmatrix} = \begin{bmatrix}\mathbf{y}^{(1)}(\xi_1)\\ \mathbf{u}^{(1)}(\xi_1)\\ \boldsymbol{\lambda}^{(1)}(\xi_1)\end{bmatrix} u^{(2)}(\xi_2)u^{(3)}(\xi_3)u^{(4)}(\xi_4),$$

as it is returned from the block TT cross. The Gauss-Newton system corresponding to the elliptic PDE reads

$$\begin{bmatrix}\mathbf{M}_y & 0 & \mathbf{A}(\xi_1)\\ 0 & \mathbf{M}_u & -\mathbf{B}^\top\\ \mathbf{A}(\xi_1) & -\mathbf{B} & 0\end{bmatrix}\begin{bmatrix}\mathbf{y}(\xi)\\ \mathbf{u}(\xi)\\ \boldsymbol{\lambda}(\xi)\end{bmatrix} = \begin{bmatrix}\mathbf{M}_y\mathbf{y}_d\\ 0\\ -\mathbf{g}(\xi_2)-\mathbf{b}_3(\xi_3)-\mathbf{b}_4(\xi_4)\end{bmatrix}, \tag{A.1}$$

where $\mathbf{M}_y$ is the mass matrix in state, independent of ξ, $\mathbf{A}(\xi_1)$ is the symmetric stiffness matrix of $\nu(\xi_1)\Delta$ which depends only on ξ_1, $\mathbf{M}_u$ is the control part of the approximate Hessian of the Lagrangian (independent of ξ), $\mathbf{B}$ is the actuator matrix independent of ξ, so is the desired state $\mathbf{y}_d$, and $\mathbf{b}_3(\xi_3)$ and $\mathbf{b}_4(\xi_4)$ are the right hand sides accommodating left and right boundary conditions, respectively, each of which depends on either ξ_3 or ξ_4. We split the state into 4 components: $\mathbf{y}(\xi) = \mathbf{y}_u(\xi) + \mathbf{y}_g(\xi) + \mathbf{y}_3(\xi) + \mathbf{y}_4(\xi)$, which satisfy the following equations:

$$\mathbf{A}(\xi_1)\mathbf{y}_u(\xi) = \mathbf{B}\mathbf{u}(\xi), \qquad \mathbf{A}(\xi_1)\mathbf{y}_g(\xi) = -\mathbf{g}(\xi_2) \tag{A.2}$$

$$\mathbf{A}(\xi_1)\mathbf{y}_3(\xi) = -\mathbf{b}_3(\xi_3), \qquad \mathbf{A}(\xi_1)\mathbf{y}_4(\xi) = -\mathbf{b}_4(\xi_4). \tag{A.3}$$

Since $\mathbf{A}(\xi_1)^{-1}$ acts linearly on a function independent of ξ_1, we obtain rank-1 TT decompositions

$$\mathbf{y}_g(\xi) = \mathbf{y}_g^{(1)}(\xi_1)y_g^{(2)}(\xi_2), \qquad \mathbf{y}_3(\xi) = \mathbf{y}_3^{(1)}(\xi_1)y_3^{(3)}(\xi_3), \qquad \mathbf{y}_4(\xi) = \mathbf{y}_4^{(1)}(\xi_1)y_4^{(4)}(\xi_4).$$

From the first equation of (A.1) and (A.2) we get

$$\boldsymbol{\lambda}(\xi) = \mathbf{A}(\xi_1)^{-1}\left(\mathbf{M}_y\mathbf{y}_d - \mathbf{M}_y\mathbf{A}(\xi_1)^{-1}\mathbf{B}\mathbf{u}(\xi) - \mathbf{M}_y\mathbf{y}_g(\xi) - \mathbf{M}_y\mathbf{y}_3(\xi) - \mathbf{M}_y\mathbf{y}_4(\xi)\right),$$

whereas from the second equation

$$\mathbf{u}(\xi) = \mathbf{M}_u^{-1}\mathbf{B}^\top\boldsymbol{\lambda}(\xi),$$

which gives us the Schur complement depending only on ξ_1,

$$\begin{aligned}\underbrace{\left(\mathbf{I} + \mathbf{A}(\xi_1)^{-1}\mathbf{M}_y A(\xi_1)^{-1}\mathbf{B}\mathbf{M}_u^{-1}\mathbf{B}^\top\right)}_{\mathbf{S}(\xi_1)}\boldsymbol{\lambda}(\xi) &= \mathbf{A}(\xi_1)^{-1}\mathbf{M}_y\left(\mathbf{y}_d - \mathbf{y}_g(\xi) - \mathbf{y}_3(\xi) - \mathbf{y}_4(\xi)\right)\\ \boldsymbol{\lambda}(\xi) &= \mathbf{Q}(\xi_1)\left(\mathbf{y}_d - \mathbf{y}_g(\xi) - \mathbf{y}_3(\xi) - \mathbf{y}_4(\xi)\right),\\ \mathbf{Q}(\xi_1) &= \mathbf{S}(\xi_1)^{-1}\mathbf{A}(\xi_1)^{-1}\mathbf{M}_y.\end{aligned}$$

Each summand in the right hand side of $\boldsymbol{\lambda}(\xi)$ is a rank-1 TT decomposition, so $\boldsymbol{\lambda}(\xi)$ is a TT decomposition of ranks not greater than 4,

$$\boldsymbol{\lambda}(\xi) = \boldsymbol{\lambda}^{(1)}(\xi_1)\lambda^{(2)}(\xi_2)\lambda^{(3)}(\xi_3)\lambda^{(4)}(\xi_4).$$

So are $\mathbf{u}(\xi)$ and $\mathbf{y}_u(\xi)$, which moreover differ from $\boldsymbol{\lambda}(\xi)$ only in the first TT cores. Finally, adding rank-1

$\mathbf{y}_g(\xi)$, $\mathbf{y}_3(\xi)$ and $\mathbf{y}_4(\xi)$, we obtain that $\mathbf{y}(\xi)$ is a TT decomposition of ranks not greater than 7. Specifically,

$$\begin{bmatrix}\mathbf{y}(\xi)\\ \mathbf{u}(\xi)\\ \boldsymbol{\lambda}(\xi)\end{bmatrix} = \begin{bmatrix}\mathbf{A}(\xi_1)^{-1}\mathbf{B}\mathbf{M}_u^{-1}\mathbf{B}^\top\boldsymbol{\lambda}^{(1)}(\xi_1)\\ \mathbf{M}_u^{-1}\mathbf{B}^\top\boldsymbol{\lambda}^{(1)}(\xi_1)\\ \boldsymbol{\lambda}^{(1)}(\xi_1)\end{bmatrix}\lambda^{(2)}(\xi_2)\lambda^{(3)}(\xi_3)\lambda^{(4)}(\xi_4)$$
$$+\begin{bmatrix}\mathbf{y}_g^{(1)}(\xi_1)\\ 0\\ 0\end{bmatrix}y_g^{(2)}(\xi_2)+\begin{bmatrix}\mathbf{y}_3^{(1)}(\xi_1)\\ 0\\ 0\end{bmatrix}y_3^{(3)}(\xi_3)+\begin{bmatrix}\mathbf{y}_4^{(1)}(\xi_1)\\ 0\\ 0\end{bmatrix}y_4^{(4)}(\xi_4).$$

REFERENCES

[1] F. Airaudo, H. Antil, R. Löhner, and U. Rakhimov. On the use of risk measures in digital twins to identify weaknesses in structures. In *AIAA SCITECH 2024 Forum*, page 2622, 2024.

[2] A. Ali A, E. Ullmann, and M. Hinze. Multilevel Monte Carlo analysis for optimal control of elliptic PDEs with random coefficients. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):466–492, 2017.

[3] E. Andreassen, A. Clausen, M. Schevenels, B. S. Lazarov, and O. Sigmund. Efficient topology optimization in matlab using 88 lines of code. *Structural and Multidisciplinary Optimization*, 43(1):1–16, 2011.

[4] H. Antil. Mathematical opportunities in digital twins (MATH-DT). *arXiv preprint arXiv:2402.10326*, 2024.

[5] H. Antil, S. Dolgov, and A. Onwunta. Smoothed Moreau-Yosida Tensor Train approximation of state-constrained optimization problems under uncertainty. *arXiv:2301.08684*, 2023.

[6] H. Antil, S. Dolgov, and A. Onwunta. TTRISK: tensor train decomposition algorithm for risk averse optimization. *Numerical Linear Algebra with Applications*, 30(3):Paper No. e2481, 29, 2023.

[7] M Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on signal processing*, 50(2):174–188, 2002.

[8] C. Audouze, A. Klein, A. Butscher, N. Morris, P. Nair, and M. Yano. Robust level-set-based topology optimization under uncertainties using anchored ANOVA Petrov–Galerkin method. *SIAM/ASA Journal on Uncertainty Quantification*, 11(3):877–905, 2023.

[9] P. Benner, A. Onwunta, and M. Stoll. Block-diagonal preconditioning for optimal control problems constrained by PDEs with uncertain inputs. *SIAM Journal on Matrix Analysis and Applications*, 37(2):491–518, 2016.

[10] D. Bigoni, A. P. Engsig-Karup, and Y. M. Marzouk. Spectral tensor-train decomposition. *SIAM J. Sci. Comput.*, 38(4):A2405–A2439, 2016.

[11] P. Chen, M. R. Haberman, and O. Ghattas. Optimal design of acoustic metamaterial cloaks under uncertainty. *Journal of Computational Physics*, 431:Paper No. 110114, 23, 2021.

[12] P. Chen and A. Quarteroni. Weighted reduced basis method for stochastic optimal control problems with elliptic PDE constraint. *SIAM/ASA Journal on Uncertainty Quantification*, 2(1):364–396, 2014.

[13] S. Dolgov, B. N. Khoromskij, A. Litvinenko, and H. G. Matthies. Polynomial Chaos Expansion of random coefficients and the solution of stochastic partial differential equations in the Tensor Train format. *SIAM J. Uncertainty Quantification*, 3(1):1109–1135, 2015.

[14] S. Dolgov and D. Savostyanov. Parallel cross interpolation for high–precision calculation of high–dimensional integrals. *Comput. Phys. Commun.*, 246:106869, 2020.

[15] S. V. Dolgov, B. N. Khoromskij, I. V. Oseledets, and D. V. Savostyanov. Computation of extreme eigenvalues in higher dimensions using block tensor train format. *Comput. Phys. Commun.*, 185(4):1207–1216, 2014.

[16] T. Duswald, B. Keith, B. Lazarov, S. Petrides, and B. Wohlmuth. Finite elements for matérn-type random fields: Uncertainty in computational mechanics and design optimization. *Computer Methods in Applied Mechanics and Engineering*, 429:117146, 2024.

[17] D.B. Gahururu, M. Hintermüller, and T.M. Surowiec. Risk-neutral pde-constrained generalized nash equilibrium problems. *Mathematical Programming*, 2022.

[18] S. Garreis and M. Ulbrich. Constrained optimization with low-rank tensors and applications to parametric problems with PDEs. *SIAM Journal on Scientific Computing*, 39(1):A25–A54, 2017.

[19] S. A. Goreinov, I. V. Oseledets, D. V. Savostyanov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. How to find a good submatrix. In V. Olshevsky and E. Tyrtyshnikov, editors, *Matrix Methods: Theory, Algorithms, Applications*, pages 247–256. World Scientific, Hackensack, NY, 2010.

[20] A. Gorodetsky, S. Karaman, and Y. Marzouk. A continuous analogue of the tensor-train decomposition. *Comput. Methods Appl. Mech. Engrg.*, 347:59–84, 2019.

[21] W. Hackbusch. *Tensor Spaces And Numerical Tensor Calculus*. Springer–Verlag, Berlin, 2012.

[22] W. Hackbusch and B. N. Khoromskij. Low-rank Kronecker-product approximation to multi-dimensional nonlocal operators. I. Separable approximation of multi-variate functions. *Computing*, 76(3-4):177–202, 2006.

[23] M. Hoffhues, W. Römisch, and T. M. Surowiec. On quantitative stability in infinite-dimensional optimization under uncertainty. *Optimization Letters*, 15(8):2733–2756, 2021.

[24] S. Holtz, T. Rohwedder, and R. Schneider. The alternating linear scheme for tensor optimization in the tensor train format. *SIAM J. Sci. Comput.*, 34(2):A683–A713, 2012.
[25] A. H. Jazwinski. *Stochastic processes and filtering theory*. Dover Publications Inc., NY, 2007.
[26] V. Keshavarzzadeh, R. M. Kirby, and A. Narayan. Robust topology optimization with low rank approximation using artificial neural networks. *Computational Mechanics*, 68(6):1297–1323, 2021.
[27] 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.
[28] D. P. Kouri and A. Shaprio. Optimization of PDEs with uncertain inputs. In H. Antil, D. P. Kouri, M.-D. Lacasse, and D. Ridzal, editors, *Frontiers in PDE-Constrained Optimization*, volume 163, pages 41 – 81, Berlin, Heidelberg, New-York, 2018. Springer Verlag.
[29] D. P. Kouri and T. M. Surowiec. Risk-averse PDE-constrained optimization using the conditional value-at-risk. *SIAM J. Optim.*, 26(1):365–396, 2016.
[30] C. Li and G. Stadler. Sparse solutions in optimal control of PDEs with uncertain parameters: the linear case. *SIAM Journal on Control and Optimization*, 57(1):633–658, 2019.
[31] M. Martin and F. Nobile. PDE-constrained optimal control problems with uncertain parameters using SAGA. *SIAM/ASA Journal on Uncertainty Quantification*, 9(3):979–1012, 2021.
[32] J. Milz. Reliable error estimates for optimal control of linear elliptic PDEs with random inputs. *SIAM/ASA Journal on Uncertainty Quantification*, 11(4):1139–1163, 2023.
[33] F. Negri, G. Rozza, A. Manzoni, and A. Quarteroni. Reduced basis method for parametrized elliptic optimal control problems. *SIAM Journal on Scientific Computing*, 35(5):A2316–A2340, 2013.
[34] I. V. Oseledets. Tensor train decomposition. *SIAM J. Sci. Comp.*, 33(5):2295 – 2317, 2011.
[35] I. V. Oseledets and E. E. Tyrtyshnikov. TT-cross approximation for multidimensional arrays. *Linear Algebra Appl.*, 432(1):70–88, 2010.
[36] P. B. Rohrbach, S. Dolgov, L. Grasedyck, and R. Scheichl. Rank bounds for approximating Gaussian densities in the Tensor-Train format. *SIAM/ASA Journal on Uncertainty Quantification*, 10(3):1191–1224, 2022.
[37] D. V. Savostyanov and I. V. Oseledets. Fast adaptive interpolation of multi-dimensional arrays in tensor train format. In *Proceedings of 7th International Workshop on Multidimensional Systems (nDS)*. IEEE, 2011.
[38] R. Schneider and A. Uschmajew. Approximation rates for the hierarchical tensor format in periodic Sobolev spaces. *J. Complexity*, 2013.
[39] U. Schollwöck. The density matrix renormalization group. *Rev. Mod. Phys.*, 77(1):259–315, 2005.
[40] H. Tiesler, R. M. Kirby, D. Xiu, and T. Preusser. Stochastic collocation for optimal control problems with stochastic PDE constraints. *SIAM Journal on Control and Optimization*, 50(5):2659–2682, 2012.
[41] A. P. Torres, J. E. Warner, M. A. Aguiló, and J. K. Guest. Robust topology optimization under loading uncertainties via stochastic reduced order models. *International Journal for Numerical Methods in Engineering*, 122(20):5718–5743, 2021.
[42] C. R. Vogel. *Computational Methods for Inverse Problems*. Society for Industrial and Applied Mathematics, 2002.
[43] S. R. White. Density matrix algorithms for quantum renormalization groups. *Phys. Rev. B*, 48(14):10345–10356, 1993.