SAND2022-14948C

Adaptive Randomized Sketching for Dynamic Nonsmooth Optimization

Robert J. Baraldi[1], Evelyn Herberg[2], Drew P. Kouri[1], and Harbir Antil[3]

[1]Optimization and Uncertainty Quantification
Sandia National Laboratories, Albuquerque, NM 87123

[2]Interdisciplinary Center for Scientific Computing
Heidelberg University, 69120 Heidelberg, Germany

[3]Department of Mathematical Sciences and Center for Mathematics and Artificial Intelligence
George Mason University, Fairfax, VA 22030

ABSTRACT
Dynamic optimization problems arise in many applications including optimal flow control, full waveform inversion, and medical imaging, where they are plagued by significant computational challenges. For example, memory is often a limiting factor on the size of problems one can solve since the evaluation of derivatives requires the entire state trajectory. Additionally, many applications employ nonsmooth regularizers such as the L^1-norm or the total variation as well as auxiliary constraints on the optimization variables. In this paper, we introduce a novel trust-region algorithm for minimizing the sum of a smooth, nonconvex function and a nonsmooth, convex function that addresses these two challenges. Our algorithm employs randomized sketching to store a compressed version of the state trajectory for use in derivative computations. By allowing the trust-region algorithm to adaptively learn the rank of the state sketch, we arrive at a provably convergent method with near optimal memory requirements. We demonstrate the efficacy of our method on a parabolic PDE-constrained optimization problem with measure-valued control variables.

Keywords: Nonsmooth Optimization, Optimal Control, Randomized Sketching, Dynamic Optimization, Compression, PDE-Constrained Optimization

INTRODUCTION

We consider the discrete-time dynamic optimization problem

$$\min_{u_n\in\mathbb{R}^M,\, z_n\in\mathbb{R}^m} \sum_{n=1}^{N} f_n(u_{n-1},u_n,z_n)+\phi_n(z_n) \qquad \text{subject to} \qquad c_n(u_{n-1},u_n,z_n)=0 \quad \text{for} \quad n=1,\dots,N, \tag{1}$$

where $z_n \in \mathbb{R}^m$ is the control variable, $u_n \in \mathbb{R}^M$ is the state variable at the n-th time step for $n = 1,\dots,N$, and $u_0 \in \mathbb{R}^M$ is the prescribed initial system state. Additionally, $f_n : \mathbb{R}^M \times \mathbb{R}^M \times \mathbb{R}^m \to \mathbb{R}$ is the objective function associated with the n-th control and state, $\phi_n : \mathbb{R}^m \to (-\infty,+\infty]$ is a potentially nonsmooth control penalty function, and $c_n : \mathbb{R}^M \times \mathbb{R}^M \times \mathbb{R}^m \to \mathbb{R}^M$ is the dynamic constraint function, which advances the state from u_{n-1} to u_n. Dynamic optimization problems of the form (1) arise in many applications, including turbulent flow control [1], energy system operations [2], vortex control in nuclear reactors and superconductors [3], optimal tomography [4, 5], full waveform inversion [6–8], and airflows in closed environments [9–11]. In addition, nonsmooth penalties are often used to enforce constraints [12–14] or to ensure sparsity in optimal control, parameter estimation and learning [15–23].

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & 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.

The memory required to store the state trajectory $\{u_1,\ldots,u_N\}$ and auxiliary information like Lagrange multipliers presents a significant challenge when solving (1). For example, sequential quadratic programming (SQP) methods require the storage of $N(2M+m)$ floating point numbers. In full waveform inversion, the spatial discretization size often is $M \approx 10^{10}$ and the temporal discretization size is $N \approx 10^5$, requiring the storage of $\mathscr{O}(10^{50})$ floating point numbers [24]. In contrast, if $c_n(u_{n-1},u_n,z_n) = 0$ is uniquely solvable for u_n with fixed u_{n-1} and z_n for each n, then one can reformulate (1) as a minimization problem only in $\{z_1,\ldots,z_N\}$. On the surface, this approach reduces the memory requirement to Nm. However, when solving the reduced problem using derivative-based optimization, the gradient calculation requires the entire state trajectory, again leading to $\mathscr{O}(N(M+m))$ storage. Reducing these storage requirements typically comes at the cost of model fidelity by using, e.g., reduced-order models (ROMs) or low-order discretizations [25–27]. The quality of a fixed ROM can degrade as the optimization routine progresses, leading to adaptive ROM generation [28,29]. Unfortunately, ROMs are generally limited to specific classes of dynamical systems and can be difficult to implement in legacy codes. On the other hand, for the reduced problem one can reduce the memory burden using checkpointing [30–33], which stores judiciously chosen snapshots of the state trajectory for use when computing the gradient. Although this procedure has lower memory requirements, it drastically increases the cost of computing the gradient.

In this paper, we employ adaptive randomized sketching to compress the state trajectory as in [34] to reduce the memory requirement for solving (1). In particular, we generate low-rank approximations of the state trajectory that we use to compute an inexact gradient. In contrast to checkpoint, this approach does not increase the computational burden. We control the gradient error using the trust-region algorithm introduced in [35], resulting in a provably convergent, low-memory algorithm for solving (1). We demonstrate our algorithm's performance on a discretized parabolic PDE-constrained optimization problem with measure-valued controls.

DYNAMIC OPTIMIZATION PROBLEM FORMULATION

We consider the reduced form of (1) where u_n is replaced by the unique solution to $c(u_{n-1},u_n,z_n) = 0$ for fixed u_{n-1} and z_n. To formulate the reduced problem, we collect the controls and states into stacked column vectors, denoted by

$$\mathbf{z} = [z_1^\top,\ldots,z_N^\top]^\top \in \mathscr{Z} := \mathbb{R}^{Nm} \qquad \text{and} \qquad \mathbf{u} = [u_1^\top,\ldots,u_N^\top]^\top \in \mathscr{U} := \mathbb{R}^{NM}.$$

We employ the notation $U \in \mathbb{R}^{M\times N}$ to denote the matrix with n-th column u_n for $n = 1,\ldots,N$. Using this notation, we can represent the dynamic constraint and objective functions as

$$c(\mathbf{u},\mathbf{z}) := \begin{bmatrix} c_1(u_0,u_1,z_1)) \\ \vdots \\ c_N(u_{N-1},u_N,z_N) \end{bmatrix}, \qquad f(\mathbf{u},\mathbf{z}) := \sum_{n=1}^{N} f_n(u_n,z_n), \qquad \text{and} \qquad \phi(\mathbf{z}) := \sum_{n=1}^{N} \phi_n(z_n),$$

enabling us to rewrite (1) as

$$\min_{\mathbf{u}\in\mathscr{U},\,\mathbf{z}\in\mathscr{Z}} f(\mathbf{u},\mathbf{z}) + \phi(\mathbf{z}) \qquad \text{subject to} \qquad c(\mathbf{u},\mathbf{z}) = 0. \tag{2}$$

We assume that f and c are continuously differentiable on $\mathscr{U}\times\mathscr{Z}$ and that there exists a control-to-state map $\mathbf{z}\mapsto S(\mathbf{z}) : \mathscr{Z}\to\mathscr{U}$, where $S(\mathbf{z})$ is the unique state trajectory satisfying $c(S(\mathbf{z}),\mathbf{z}) = 0$ for each $\mathbf{z}\in\mathscr{Z}$. In addition, we require that the state Jacobian of c, denoted $\mathsf{d}_{\mathbf{u}}c(\mathbf{u},\mathbf{z})$, has a bounded inverse for all controls $\mathbf{z}\in\mathscr{Z}$. Analogously, we denote the control Jacobian by $\mathsf{d}_{\mathbf{z}}c(\mathbf{u},\mathbf{z})$ and the partial derivatives of f by $\mathsf{d}_{\mathbf{u}}f(\mathbf{u},\mathbf{z})$ and $\mathsf{d}_{\mathbf{z}}f(\mathbf{u},\mathbf{z})$. The control-to-state map has the form

$$S(\mathbf{z}) := \begin{bmatrix} S_1(u_0,z_1) \\ S_2(S_1(u_0,z_1),z_2) \\ \vdots \\ S_N(S_{N-1}(\ldots,z_{N-1}),z_N) \end{bmatrix},$$

where the implicit function theorem [36, Th. 1.41] ensures that S_n and S are continuously differentiable. We can thereby reformulate (2) as the reduced dynamic optimization problem

$$\min_{\mathbf{z}\in\mathscr{Z}} \left\{F(\mathbf{z}) := j(\mathbf{z}) + \phi(\mathbf{z})\right\}, \tag{3}$$

where $j(\mathbf{z}) := f(S(\mathbf{z}),\mathbf{z})$ is the reduced objective function. Under the stated assumptions, j is continuously differentiable and its gradient is given by

$$\nabla j(\mathbf{z}) = \mathsf{d}_{\mathbf{z}}f(S(\mathbf{z}),\mathbf{z}) + (\mathsf{d}_{\mathbf{z}}c(S(\mathbf{z}),\mathbf{z}))^\top \lambda, \tag{4}$$

where $\lambda \in \mathbb{R}^{MN}$ solves the adjoint equation

$$\mathsf{d}_{\mathbf{u}} c(S(\mathbf{z}),\mathbf{z})^\top \lambda = -\mathsf{d}_{\mathbf{u}} f(S(\mathbf{z}),\mathbf{z}). \tag{5}$$

Recall that the adjoint equation (5) is solved backward in time, starting at $n = N$ and requires the entire state trajectory.

LOW-MEMORY MATRIX APPROXIMATION

For many real-world applications, the state trajectory can be so large as to prohibit storage in working memory. To overcome this challenge, we utilize low-rank matrix sketching to compress the state, which collects *sketched* information about the matrix U from which it can be accurately reconstructed on a fixed storage budget. There are many randomized sketching approaches available (cf. [34] and the references therein) that can be interchanged with the method described below.

We produce a sketch of the state matrix $U \in \mathbb{R}^{M\times N}$ with target rank r, denoted U^r, that requires $\mathscr{O}(r(M+N))$ storage [37]. Let the sketch parameters be $s \geq k \geq r$. A common choice for these parameters is $k = 2r+1$ and $s = 2k+1$. The sketch is defined by fixing four random linear dimension reduction maps (DRMs) with i.i.d. standard normal entries:

$$\Upsilon \in \mathbb{R}^{k\times M}, \qquad \Omega \in \mathbb{R}^{k\times N}, \qquad \Phi \in \mathbb{R}^{s\times M}, \qquad \text{and} \qquad \Psi \in \mathbb{R}^{s\times N}.$$

The sketch of U consists of the co-range sketch X, the range sketch Y, and the core sketch Z given by

$$X := \Upsilon U \in \mathbb{R}^{k\times N}, \qquad Y := U\Omega^\top \in \mathbb{R}^{M\times k}, \qquad \text{and} \qquad Z := \Phi U \Psi^\top \in \mathbb{R}^{s\times s}.$$

The range sketch captures the row space (top left singular vectors), the co-range sketch captures the column space (top right singular vectors), and the core sketch captures their interactions (singular values). Linearity of the sketch allows for the online computation of U^r without storing the full state. Since the columns of the state matrix, U, are computed sequentially, we can update sketch components X, Y, and Z in an streaming fashion. For example, the co-range sketch $X = X^{(N)}$ is computed as

$$X^{(0)} = 0 \qquad \text{and} \qquad X^{(n)} = X^{(n-1)} + \Upsilon u_n e_n^\top \qquad \text{for} \qquad n = 1,\dots,N,$$

where e_n is the n-th unit vector. Analogous schemes are used to update Y and Z. The sketching matrices require storing $k(M+N)+s^2$ floating point numbers, and hence for target rank r, the memory requirement is $\mathscr{O}(r(M+N)+r^2)$.

To recover the state trajectory from the sketching matrices X, Y, and Z, we first compute QR factorizations of $X^\top$ and Y [34]

$$X^\top = PR_1 \qquad \text{and} \qquad Y = QR_2,$$

where $P \in \mathbb{R}^{N\times k}$ and $Q \in \mathbb{R}^{M\times k}$. We then solve two small least-squares problems to form the matrix

$$C = (\Phi Q)^\dagger Z((\Psi P)^\dagger)^\top \in \mathbb{R}^{k\times k}.$$

The rank-k approximation of U is then given by

$$U \approx QCP^\top.$$

This is truncated to rank r by replacing C with its best rank-r approximation. While solving the dynamic optimization problem (3), we overwrite X and Y with Q and $W := CP^\top$. For more information see [34, Sect. 3] and the references therein.

SKETCHED TRUST-REGION ALGORITHM

We utilize a trust-region method to solve (3), while leveraging inexact gradient computations resulting from sketching. As mentioned, our algorithm is an instance of the trust-region method introduced in [35]. Although the method in [35] is provably convergent in Hilbert space, we restrict our developments to $\mathscr{X} = \mathbb{R}^{mN}$. Following standard convex analysis notation, we denote the subdifferential of a proper, closed and convex function $\psi : \mathscr{X} \to (-\infty,\infty]$ at an arbitrary vector $z \in \mathscr{X}$ by

$$\partial\psi(z) := \{\eta \in \mathscr{X} \mid \psi(y) \geq \psi(z) + \langle \eta, y-z\rangle \ \ \forall\, y \in \mathscr{X}\}$$

and the effective domain of ψ and $\partial\psi$ by $\operatorname{dom}\psi := \{z \in \mathscr{X} \mid \psi(z) < +\infty\}$ and $\operatorname{dom}\partial\psi := \{z \in \mathscr{X} \mid \partial\psi(z) \neq \emptyset\}$, respectively. Furthermore, the proximal mapping of ψ for fixed $t > 0$ is

$$\operatorname{Prox}_{t\psi}(y) := \operatorname*{arg\,min}_{z\in\mathscr{X}} \left\{ \psi(z) + \tfrac{1}{2t}\|z-y\|^2 \right\}. \tag{6}$$

Recall that if $\psi = \iota_{\mathscr{C}}$ is the indicator function of a nonempty, closed and convex set $\mathscr{C} \subset \mathscr{Z}$ (i.e., $\iota_{\mathscr{C}}(z) = 0$ if $z \in \mathscr{C}$ and $+\infty$ otherwise), then $\mathrm{Prox}_{t\psi}$ is the metric projection onto $\mathscr{C}$.

To develop our convergence theory, we make the following assumptions on the components of the objective function F in (3).

Assumption 1 (Problem Data)**.**

1. *The function $\phi : \mathscr{Z} \to (-\infty, +\infty]$ is proper, closed and convex.*
2. *The function $j : \mathscr{Z} \to \mathbb{R}$ is L-smooth on* $\operatorname{dom}\phi$*. That is, j is Fréchet differentiable and its gradient ∇j is Lipschitz continuous with modulus $L > 0$ on an open set $\bar{\mathscr{Z}} \subseteq \mathscr{Z}$ containing* $\operatorname{dom}\phi$.
3. *The objective function $F := j + \phi$ is bounded below on* $\operatorname{dom}\phi$.

At each iteration of our algorithm, we compute a trial iterate $\mathbf{z}_k^+$ that approximately solves the trust-region subproblem

$$\min_{\mathbf{z}\in\mathscr{Z}} \{m_k(\mathbf{z}) := j_k(\mathbf{z}) + \phi(\mathbf{z})\} \qquad \text{subject to} \qquad \|\mathbf{z} - \mathbf{z}_k\| \le \Delta_k, \tag{7}$$

where $\mathbf{z}_k \in \operatorname{dom}\phi$ is the current iterate, j_k is a smooth local model of j around $\mathbf{z}_k$, and $\Delta_k > 0$ is the trust-region radius. We restrict our attention to quadratic models, j_k, with the form

$$j_k(\mathbf{z}) = \tfrac{1}{2}(\mathbf{z} - \mathbf{z}_k)^\top B_k(\mathbf{z} - \mathbf{z}_k) + g_k^\top(\mathbf{z} - \mathbf{z}_k),$$

where $B_k = B_k^\top \in \mathbb{R}^{mN\times mN}$ approximates the Hessian of j at $\mathbf{z}_k$ and g_k approximates the gradient (e.g., via sketching). For are example, we employ the sketched Hessian application described in Algorithms A.5 and A.6 in [34].

To ensure convergence of our trust-region algorithm, we require that the trial iterate $\mathbf{z}_k^+$ satisfies the trust-region constraint and the fraction of Cauchy decrease (FCD) condition:

$$\left\|\mathbf{z}_k^+ - \mathbf{z}_k\right\| \le \kappa_{\mathrm{rad}}\Delta_k \qquad \text{and} \qquad m_k(\mathbf{z}_k) - m_k(\mathbf{z}_k^+) \ge \kappa_{\mathrm{fcd}}\, h_k \min\left\{\frac{h_k}{1+\|B_k\|}, \Delta_k\right\}, \tag{8}$$

where $\kappa_{\mathrm{rad}},\ \kappa_{\mathrm{fcd}} > 0$ are independent of k and for a fixed positive constant $t > 0$,

$$h_k := t^{-1}\left\|\mathrm{Prox}_{t\phi}(\mathbf{z}_k - t g_k) - \mathbf{z}_k\right\|.$$

Commonly, one has $\kappa_{\mathrm{rad}} = 1$. Note that (8) ensures that $\mathbf{z}_k^+ \in \operatorname{dom}\phi$ since the left-hand side of the second inequality would be $-\infty$ otherwise. Given a trial iterate $\mathbf{z}_k^+$ that satisfies (8), the trust-region algorithm decides whether or not to accept $\mathbf{z}_k^+$ based on the ratio of actual and predicted reduction

$$\rho_k := \frac{\mathrm{ared}_k}{\mathrm{pred}_k} = \frac{F(\mathbf{z}_k) - F(\mathbf{z}_k^+)}{m_k(\mathbf{z}_k) - m_k(\mathbf{z}_k^+)}. \tag{9}$$

Here, ared_k is the reduction of the objective function F achieved by $\mathbf{z}_k^+$ relative to $\mathbf{z}_k$ and pred_k is the reduction of the model m_k. In particular, if $\rho_k \ge \eta_1$ for $\eta_1 \in (0,1)$, we accept $\mathbf{z}_{k+1} = \mathbf{z}_k^+$. Otherwise, we set $\mathbf{z}_{k+1} = \mathbf{z}_k$. The trust-region algorithm then increases the radius Δ_k if $\rho_k \ge \eta_2$ for $\eta_2 \in (\eta_1, 1)$ and reduces Δ_k if $\rho_k < \eta_1$. The algorithmic parameters $0 < \eta_1 < \eta_2 < 1$ are user-specified with common values $\eta_1 = 10^{-4}$ and $\eta_2 = 0.75$.

The computation of the gradient of j requires the solution of the backward-in-time adjoint equation (5), which depends on the state trajectory $S(\mathbf{z})$. Instead of storing the entire state trajectory, we compress $S(\mathbf{z})$ using sketching and then recover each u_n as needed. This procedure introduces errors in the adjoint and hence gradient. Fortunately, trust-region algorithms are able to rigorously handle inexact gradients, while guaranteeing global convergence [38–41]. The following assumption describes the required gradient accuracy and is adapted from [42]. Moreover, this condition is related to the classical conditions used in [43–45].

Assumption 2 (Inexact Gradient)**.** *There exists a constant $\kappa_{\mathrm{grad}} \ge 0$, independent of k, such that the gradient g_k satisfies*

$$\|g_k - \nabla j(\mathbf{z}_k)\| \le \kappa_{\mathrm{grad}} \min\{h_k, \Delta_k\} \quad \forall k. \tag{10}$$

We provide implementation details for the inexactness conditions (10) in Algorithm 2 in the following section. Algorithm 2 is a combination of Algorithm 4 in [35] and the adaptive rank procedure described in Algorithm 4.4 of [34]. We list the nonsmooth trust-region algorithm in Algorithm 1. This algorithm is closely related to the inexact trust-region algorithm described in [46] for smooth unconstrained problems and in [39] for convex-constrained problems.

Algorithm 1 Sketched Nonsmooth Trust-Region Algorithm

```
Require: Initial guess $\mathbf{z}_1 \in \operatorname{dom}\phi$, initial rank parameter $r_1$, initial radius $\Delta_1 > 0$, $0 < \eta_1 < \eta_2 < 1$, and $0 < \gamma_1 \le \gamma_2 < 1 \le \gamma_3$
1: for $k = 1, 2, \ldots$ do
2:     Model Selection: Use Algorithm 2 with rank $r_k$ to compute $g_k$ and choose $B_k$
3:     Step Computation: Compute $\mathbf{z}_k^+ \in \mathscr{Z}$ that satisfies (8)
4:     Step Acceptance and Radius Update: Compute $\rho_k$ as in (9)
5:     if $\rho_k < \eta_1$ then
6:         $\mathbf{z}_{k+1} \leftarrow \mathbf{z}_k$
7:         $\Delta_{k+1} \in [\gamma_1\Delta_k, \gamma_2\Delta_k]$
8:     else
9:         $\mathbf{z}_{k+1} \leftarrow \mathbf{z}_k^+$
10:        if $\rho_k \in [\eta_1, \eta_2)$ then
11:            $\Delta_{k+1} \in [\gamma_2\Delta_k, \Delta_k]$
12:        else
13:            $\Delta_{k+1} \in [\Delta_k, \gamma_3\Delta_k]$
14:        end if
15:    end if
16: end for
```

INEXACT GRADIENT COMPUTATION VIA SKETCHED STATE

In order to describe the adaptive gradient approximation procedure, we first define the adjoint equation residual $G : \mathscr{U} \times \mathscr{U} \times \mathscr{Z} \to \mathscr{U}$ by $G(\lambda, \mathbf{u}, \mathbf{z}) := \mathsf{d}_{\mathbf{u}} f(\mathbf{u}, \mathbf{z}) + (\mathsf{d}_{\mathbf{u}} c(\mathbf{u}, \mathbf{z}))^* \lambda$ and denote by $\Lambda(\mathbf{u}, \mathbf{z}) \in \mathscr{U}$ the solution to the adjoint equation $G(\Lambda(\mathbf{u}, \mathbf{z}), \mathbf{u}, \mathbf{z}) = 0$ for the fixed state $\mathbf{u}$ and control $\mathbf{z}$. We further define the map

$$g(\lambda, \mathbf{u}, \mathbf{z}) := \mathsf{d}_{\mathbf{z}} f(\mathbf{u}, \mathbf{z}) + (\mathsf{d}_{\mathbf{z}} c(\mathbf{u}, \mathbf{z}))^* \lambda.$$

When evaluated at $\mathbf{u} = S(\mathbf{z})$ and $\lambda = \Lambda(\mathbf{u}, \mathbf{z})$, $g(\lambda, \mathbf{u}, \mathbf{z})$ is the gradient of the reduced objective function j as in (4). By evaluating $g(\lambda, \mathbf{u}, \mathbf{z})$ at the sketch state $\mathbf{u}^r = \operatorname{vec}(U^r)$[1] instead of the full state trajectory $\mathbf{u} = S(\mathbf{z})$, we reduce the memory burden for gradient computation. However, the computed value $g^r(\mathbf{z}) = g(\Lambda(\mathbf{u}^r, \mathbf{z}), \mathbf{u}^r, \mathbf{z})$ is only an approximation of true gradient $g(\Lambda(S(\mathbf{z}), \mathbf{z}), S(\mathbf{z}), \mathbf{z})$. Algorithm 2 describes an adaptive procedure for approximating the gradient, using the sketched state $\mathbf{u}^r$.

Algorithm 2 Inexact Gradient Computation with Adaptive Rank

```
Require: Control iterate $\mathbf{z}_k \in \mathbb{R}^{mN}$, initial rank parameter $r$, sketch object for state $\mathbf{u}_k^r$, trust-region radius $\Delta_k > 0$, positive constant $\kappa_{\text{scale}} > 0$, and tolerance $\mu_{\text{grad}} > 1$.
1: Set $\tau_k^- \leftarrow \kappa_{\text{scale}}\Delta_k$
2: Compute $g_k \leftarrow g(\Lambda(\mathbf{u}_k^r, \mathbf{z}_k), \mathbf{u}_k^r, \mathbf{z}_k)$ and $h_k \leftarrow t^{-1}\left\|\operatorname{Prox}_{r\phi}(\mathbf{z}_k - t g_k) - \mathbf{z}_k\right\|$
3: Set $\tau_k^+ \leftarrow \kappa_{\text{scale}} \min\{h_k, \Delta_k\}$
4: while $\tau_k^- > \mu_{\text{grad}}\tau_k^+$ do
5:     while $r < \min\{M, N\}$ do
6:         Compute norm of the constraint residual rnorm $\leftarrow \left\|c(\mathbf{u}_k^r, \mathbf{z}_k)\right\|$
7:         if rnorm $< \tau_k^+$ then
8:             Compute gradient $g_k^r \leftarrow g(\Lambda(\mathbf{u}_k^r, \mathbf{z}_k), \mathbf{u}_k^r, \mathbf{z}_k)$
9:             break
10:        end if
11:        Increase Rank parameter $r \leftarrow 2r$
12:        Solve the state equation at $\mathbf{z}_k$ and resketch to produce $\mathbf{u}_k^r$
13:    end while
14:    Set $g_k \leftarrow g_k^r$ and compute $h_k \leftarrow t^{-1}\left\|\operatorname{Prox}_{r\phi}(\mathbf{z}_k - t g_k) - \mathbf{z}_k\right\|$
15:    Set $\tau_k^- \leftarrow \tau_k^+$ and $\tau_k^+ \leftarrow \kappa_{\text{scale}} \min\{h_k, \Delta_k\}$
16: end while
17: return Approximate gradient $g_k \approx \nabla f(\mathbf{z}_k)$ using $\mathscr{O}(r(M+N) + mN)$ storage for $r \le \min\{M, N\}$.
```

[1]The notation $\operatorname{vec}(U)$ denotes the vector obtained by stacking the columns of U.

To ensure that Algorithm 2 satisfies the required accuracy (10) with finitely many rank updates, we make the following regularity assumptions on the problem data in (1).

Assumption 3 (Regularity Properties for (1))**.** *The following conditions hold for the data in* (1)*:*

1. *The set of states corresponding to controls in any open and bounded set $\mathscr{Z}_0 \subseteq \mathscr{Z}$ is bounded: there exists $\mathscr{U}_0 \subset \mathscr{U}$ open and bounded such that $\{\mathbf{u} \in \mathscr{U} | \exists \mathbf{z} \in \mathscr{Z}_0, c(\mathbf{u},\mathbf{z}) = 0\} \subseteq \mathscr{U}_0$.*
2. *There exists singular value thresholds $0 < \sigma_0 \leq \sigma_1 < +\infty$ such that for any $\mathbf{u} \in \mathscr{U}_0$ and $\mathbf{z} \in \mathscr{Z}_0$, the state Jacobian matrix $\mathsf{d}_{\mathbf{u}}c(\mathbf{u},\mathbf{z})$ satisfies $\sigma_0 \leq \sigma_{\min}(\mathsf{d}_{\mathbf{u}}c(\mathbf{u},\mathbf{z})) \leq \sigma_{\max}(\mathsf{d}_{\mathbf{u}}c(\mathbf{u},\mathbf{z})) \leq \sigma_1$.*
3. *The following functions are Lipschitz continuous on $\mathscr{U}_0 \times \mathscr{Z}_0$ with respect to their first arguments, and their respective Lipschitz moduli are independent of $\mathbf{z} \in \mathscr{Z}_0$:*
   (a) *the state Jacobian of the constraint $\mathsf{d}_{\mathbf{u}}c(\mathbf{u},\mathbf{z})$;*
   (b) *the control Jacobian of the constraint $\mathsf{d}_{\mathbf{u}}c(\mathbf{u},\mathbf{z})$;*
   (c) *the state gradient of the smooth objective term $\mathsf{d}_{\mathbf{u}}f(\mathbf{u},\mathbf{z})$;*
   (d) *the control gradient of the smooth objective term $\mathsf{d}_{\mathbf{z}}f(\mathbf{u},\mathbf{z})$.*

Using Assumption 3, we can bound the state, adjoint and gradient errors as in [34, Prop. 4.1] and ultimately show that Algorithm 2 produces a gradient approximation that satisfies (10).

Proposition 1 (Proposition 4.1 in [34])**.** *Suppose Assumption 3 holds for a bounded control set $\mathscr{Z}_0$. Then there exists $\kappa_0, \kappa_1 > 0$ such that the error in the state satisfies*

$$\kappa_0 \|\mathbf{u} - S(\mathbf{z})\| \leq \|c(\mathbf{u},\mathbf{z})\| \leq \kappa_1 \|\mathbf{u} - S(\mathbf{z})\|, \forall \mathbf{u} \in \mathscr{U}_0, \mathbf{z} \in \mathscr{Z}_0$$

where $\mathscr{U}_0 \subseteq \mathscr{U}$ is defined in condition 1 of Assumption 3. Additionally, the error in the adjoint is controlled by the adjoint residual together with the state residual: for some $\kappa_2, \kappa_3 > 0$

$$\|\lambda - \Lambda(S(\mathbf{z}),\mathbf{z})\| \leq \kappa_2 \|c(\mathbf{u},\mathbf{z})\| + \kappa_3 \|G(\lambda,\mathbf{u},\mathbf{z})\|, \forall \mathbf{u}, \lambda \in \mathscr{U}_0, \forall \mathbf{z} \in \mathscr{Z}_0.$$

Therefore, the error in the gradient approximation $g(\lambda,\mathbf{u},\mathbf{z})$ is controlled by the adjoint and state residuals: for some $\kappa_4, \kappa_5 > 0$

$$\|g(\lambda,\mathbf{u},\mathbf{z}) - g(\Lambda(S(\mathbf{z}),\mathbf{z}),S(\mathbf{z}),\mathbf{z})\| = \|g(\lambda,\mathbf{u},\mathbf{z}) - \nabla f(\mathbf{z})\| \leq \kappa_4 \|c(\mathbf{u},\mathbf{z})\| + \kappa_5 \|G(\lambda,\mathbf{u},\mathbf{z})\|.$$

Recall that both the state and adjoint are intermediate variables used to compute the gradient $\nabla j(\mathbf{z})$ and require MN storage each. The control $\mathbf{z}$ only requires mN storage, which is often much smaller in practical applications where $m \ll M$. All constants $\kappa_i > 0$ for $i = 0,\ldots,5$ in Proposition 1 depend only on the finite quantities defined in Assumption 3. We can now prove that Algorithm 2 produces an approximate gradient that satisfies (10) in finitely many iterations.

Lemma 1 (Adaptive Rank Gradient Approximation)**.** *If Assumption 3 holds, then Algorithm 2 produces a gradient approximation $g_k = g(\Lambda(\mathbf{u}_k^r,\mathbf{z}_k),\mathbf{u}_k^r,\mathbf{z}_k)$, in finitely many iterations, that satisfies the gradient error bound Assumption 2 with $\kappa_{\text{grad}} = \kappa_4 \kappa_{\text{scale}} \mu_{\text{grad}}$.*

One can prove Lemma 1 using [34, Th. 4.4] and the discussion in Appendix B in [35]. A consequence of Lemma 1 is that Algorithm 1 is guaranteed to converge as demonstrated in the following result.

Theorem 1 (Convergence of Algorithm 1)**.** *Let $\{\mathbf{z}_k\}$ be the sequence of iterates generated by Algorithm 1 and assume that Assumptions 1 and 3 hold. In addition, suppose that there exists an open bounded set $\mathscr{Z}_0 \subset \mathscr{Z}$ with $\{z_k\} \subseteq \mathscr{Z}_0$ and that the model Hessians B_k satisfy*

$$\sum_{k=1}^{\infty} \frac{1}{b_k} = +\infty, \qquad \textit{where} \qquad b_k := 1 + \max_{i=1,\ldots,k} \|B_i\|.$$

Then

$$\liminf_{k\to+\infty} h_k = 0.$$

Proof. The problem data satisfies Assumption 3 and therefore Lemma 1 ensures that Assumption 2 holds. The result then follows from [35, Th. 3]. □

NUMERICAL RESULTS

In this section, we apply Algorithm 1 to a discretization of the parabolic PDE-constrained optimization problem

$$\begin{aligned}&\min_{z,\,u}\ \frac{1}{2}\|u-u_d\|^2_{L^2(Q)}+\iota_C(z)\\&\text{subject to}\quad\begin{cases}\partial_t u-\Delta u=0 & \text{in } Q:=\Omega\times(0,T)\\ \nabla u\cdot n=0 & \text{on } \Sigma:=\partial\Omega\times(0,T)\ .\\ u(0)=z & \text{in } \Omega\end{cases}\end{aligned}\tag{11}$$

Here, $\Omega=(0,1)^2$ and $u_d(x)=|(\sin(2\pi x_1)\sin(2\pi x_2))|^{10}$ for $x=(x_1,x_2)\in\Omega$ and all $t\in[0,T]$. In this application, the control variable z is a nonnegative, regular Borel measure representing the initial state and we enforce the constraints

$$C:=\{z\in M(\Omega)\mid \|z\|_{M(\Omega)}\le\alpha,\quad z(B)\ge 0\quad\forall\,\text{Borel subsets } B\subseteq\Omega\},$$

where $\alpha=0.1$ and $M(\Omega)$ denotes the Banach space of regular Borel measures on Ω endowed with the total variation norm. We discretize the state variable u in space using continuous piecewise linear finite elements on a uniform triangular mesh ($M=4225$) and employ a variational discretization for the controls [47]. We further discretize in time using implicit Euler with $N=501$ timesteps for $T=2$ to arrive at a problem with the form (1). After discretization, the control is represented as a linear combination of point masses located at the mesh vertices and the nonsmooth term ϕ is the indicator function of the feasible set

$$\mathscr{C}=\left\{\mathbf{z}=(z_1,\ldots,z_m)^\top\in\mathbb{R}^m\ \middle|\ \sum_{i=1}^m z_i\le\alpha,\quad z_i\ge 0\quad\text{for}\quad i=1,\ldots,m\right\},$$

where $m=M$ is the number of mesh vertices. Although the control is time independent, Algorithm 1 is still applicable. We quantify the memory savings of Algorithm 1 using the compression ratio

$$\zeta:=\frac{\text{full storage}}{\text{reduced storage}}=\frac{4225\times 501}{k(4225+501)+s^2}.$$

We solved the discretized problem using Algorithm 1 with $\kappa_{\text{scale}}=10^{-4}$. For comparison, we also solved it using Algorithm 1 in [35] with fixed-rank sketching and with no sketching. To compute the trial step in line 3 of Algorithm 1, we use the spectral proximal gradient method described in [35, Alg. 5] with a maximum of 50 iterations. We further set the maximum number of trust-region iterations to 100. We terminate Algorithm 1 if either

$$h_k\le 10^{-4}h_1\qquad\text{or}\qquad\left\|\mathbf{z}_k^+-\mathbf{z}_k\right\|\le 10^{-6}h_1.$$

Table 1 compares the performance of Algorithm 1 with the fixed-rank ($r\in\{1,\ldots,5\}$) and full-storage approaches. When using rank-1 sketching, the algorithm stopped because the norm of the trial step size was smaller than the prescribed tolerance. Overall, we see a decreasing trend in the number of iterations, resulting from fewer rejected steps as the fixed rank increases. In comparison, Algorithm 1 finished with the final rank of $r=8$. The performance of the full-storage, adaptive sketching, and fixed-rank with $r=4,5$ approaches are comparable, suggesting that Algorithm 1 is a memory-efficient, application-agnostic approach to solving dynamic optimization problems with the form (3).

CONCLUSION

In this work, we describe a low memory, application-agnostic approach for solving a class of nonsmooth dynamic optimization problems without the need to store or recompute the entire state trajectory. Our method uses randomized matrix sketching to compress the state trajectory for use when solving the adjoint equation and inexactly evaluating the gradient. We employ a trust-region algorithm to control the gradient approximation by adaptively learning the state sketch rank $r\ll\min\{M,N\}$, where N is the number of time steps and M is the size of each state. In contrast to traditional approaches that require $\mathscr{O}(N(M+m))$ memory (m being the control dimension) or significant recomputation of the state trajectory, our approach greatly reduces the storage to $\mathscr{O}(r(M+N)+mN)$ with no additional computational cost, enabling the solution of large-scale dynamic problems.

rank	objective	niter	nobjs	ngrad	nhess	nobjn	nprox	ζ
^{*1}	2.680962e-02	16	17	9	521	1036	1726	148.78
2	2.680946e-02	37	38	38	1948	4597	3873	89.12
3	2.680946e-02	31	32	32	1635	3759	3248	63.55
4	2.680946e-02	22	23	23	1163	2654	2308	49.35
5	2.680946e-02	22	23	23	1160	2640	2305	40.31
Adaptive	2.680946e-02	22	23	25	1162	2530	2309	25.95
Full	2.680946e-02	23	24	24	1212	2793	2409	---

Table 1: Algorithmic performance summary using fixed rank, adaptive rank, and full storage. The table displays the final function value (`objective`), the number of iterations (`niter`), the number of smooth objective evaluations (`nobj`), the number of gradient evaluations (`ngrad`), the number of hessian evaluations (`nhess`), the number of nonsmooth objective evaluations (`nobjn`), the number of proximal operator evaluations (`nprox`), and the compression factor (ζ).

ACKNOWLEDGEMENTS

RJB and DPK were partially supported by the U.S. Air Force Office of Scientific Research Program. HA and EH were partially supported by National Science Foundation (NSF) grants DMS-2110263, DMS-1913004 and the Air Force Office of Scientific Research under Award NO: FA9550-22-1-0248.

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] M. D. Gunzberger. *Perspectives in Flow Control and Optimization*. SIAM, Philadelphia, 2003.

[2] D. Dentcheva and W. Römisch. "Optimal power generation under uncertainty via stochastic programming". In K. Marti and P. Kall, editors, *Stochastic Programming Methods and Technical Applications*, pages 22–56, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[3] K. Harada, T. Matsuda, J. Bonevich, M. Igarashi, S. Kondo, G. Pozzi, U. Kawabe, and A. Tonomura. "Real-time observation of vortex lattices in a superconductor by electron microscope". *Nature*, 360(6399):51, 1992.

[4] S. R. Arridge and J. C. Schotland. "Optical tomography: Forward and inverse problems". *Inverse Problems*, 25:123010, 2009.

[5] A. D. Klose and A. H. Hielscher. "Optical tomography using the time-independent equation of radiative transfer–part 2: Inverse model". *Journal of Quantitative Spectroscopy and Radiative Transfer*, 72:715–732, 2002.

[6] J. R. Krebs, J. E. Anderson, D. Hinkley, R. Neelamani, S. Lee, A. Baumstein, and M.-D. Lacasse. "Fast full-wavefield seismic inversion using encoded sources". *Geophysics*, 74:WCC177–WCC188, 2009.

[7] A. Tarantola. "Linearized inversion of seismic reflection data". *Geophysical Prospecting*, 32:998–1015, 1984.

[8] M. Warner and L. Gausch. "Adaptive waveform inversion: Theory". *Geophysics*, 81:R429–R445, 2016.

[9] R. Löhner and H. Antil. "High fidelity modeling of aerosol pathogen propagation in built environments with moving pedestrians". *International Journal of Numerical Methods in Biomedical Engineering*, 37(3):3428–3434, 2021.

[10] R. Löhner, H. Antil, S. Idelsohn, and Oñate. "Detailed simulation of viral propagation in the built environment". *Computational Mechanics*, 66(5):1093–1107, 2020.

[11] R. Löhner, H. Antil, A. Srinivasan, S. Idelsohn, and E. Oñate. "High-fidelity simulation of pathogen propagation, transmission, and mitigation in the built environment". *Archives Computational Methods in Engineering*, pages 1–26, 2021.

[12] D. P. Kouri. “A matrix-free trust-region Newton algorithm for convex-constrained optimization”. *Optimization Letters*, pages 1–15, 2021.

[13] M. Schmidt, E. Berg, M. Friedlander, and K. Murphy. “Optimizing costly functions with simple constraints: A limited-memory projected quasi-Newton algorithm”. In D. van Dyk and M. Welling, editors, *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 456–463, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 16–18 Apr 2009. PMLR.

[14] P. L. Toint. “Global Convergence of a Class of Trust-Region Methods for Nonconvex Minimization in Hilbert Space”. *IMA Journal of Numerical Analysis*, 8(2):231–252, 04 1988.

[15] A. Beck and M. Teboulle. “A fast iterative shrinkage-thresholding algorithm for linear inverse problems”. *SIAM J. Imaging Sci.*, 2(1):183–202, 2009.

[16] E. van den Berg and M. P. Friedlander. “Probing the Pareto frontier for basis pursuit solutions”. *SIAM J. Sci. Comput.*, 31(2):890–912, November 2008.

[17] P. L. Combettes and J.-C. Pesquet. “Proximal splitting methods in signal processing”. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer, New York, NY, 2011.

[18] R. Herzog, J. Obermeier, and G. Wachsmuth. “Annular and sectorial sparsity in optimal control of elliptic equations”. *Computational Optimization and Applications*, 62(1):157–180, 2015.

[19] R. Herzog, G. Stadler, and G. Wachsmuth. “Directional sparsity in optimal control of partial differential equations”. *SIAM Journal on Control and Optimization*, 50(2):943–963, 2012.

[20] M. Porcelli, V. Simoncini, and M. Stoll. “Preconditioning PDE-constrained optimization with l1-sparsity and control constraints”. *Computers & Mathematics with Applications*, 74(5):1059–1075, 2017. SI: SDS2016 – Methods for PDEs.

[21] G. Stadler. “Elliptic optimal control problems with L^1-control cost and applications for the placement of control devices”. *Computational Optimization and Applications*, 44(2):159, 2009.

[22] R. Tibshirani. “Regression shrinkage and selection via the lasso”. *J. Roy. Statist. Soc. Ser. B*, 58(1):267–288, 1996.

[23] J. Yun, P. Zheng, E. Yang, A. Lozano, and A. Aravkin. “Trimming the L1 regularizer: Statistical analysis, optimization, and applications to deep learning”. In *International Conference on Machine Learning*, pages 7242–7251. PMLR, 2019.

[24] H. Antil, D. P. Kouri, M.-D. Lacasse, and D. Ridzal. *Frontiers in PDE-constrained Optimization*, volume 163. Springer, New York, NY, 2018.

[25] A. A. Jalali, C. S. Sims, and P. Famouri. “Reduced order systems”. In *Lecture Notes in Control and Information Sciences*, volume 343, Berlin, 2006. Springer-Verlag.

[26] A. C. Antoulas. *Approximation of Large-Scale Dynamical Systems*. Society for Industrial and Applied Mathematics, 2005.

[27] L. Dedé. “Reduced basis method and a psoteriori error estimation for parameterized linear-quadratic optimal control problems”. *SIAM Journal of Scientific Computation*, 32:997–1019, 2010.

[28] M. J. Zahr, K. T. Carlberg, and D. P. Kouri. “An efficient, globally convergent method for optimization under uncertainty using adaptive model reduction and sparse grids”. *SIAM/ASA Journal on Uncertainty Quantification*, 7(3):877–912, 2019.

[29] M. Fahl and E. Sachs. “Reduced order modelling approaches to PDE-constrained optimization based on proper orthogonal decomposition”. In L. T. Beigler, O. Ghattas, M. Heinkenschloss, and B. van Bloemen Waanders, editors, *Large-Scale PDE-Constrained Optimization*, volume 30 of *Lecture Notes in Computational Science and Engineering*, Berlin, 2003. Springer-Verlag.

[30] A. Griewank and A. Walther. “Algorithm 799: Revolve: An implementation of checkpointing for the reverse or adjoint mode of computational differentiation”. *ACM Transactions on Mathematical Software*, 26:19–45, 2000.

[31] G. Aupy, J. Herrmann, P. Hovland, and Y. Robert. “Optimal multistage algorithm for adjoint computation”. *SIAM Journal of Scientific Computation*, 38:C232–255, 2016.

[32] P. Stumm and A. Walther. “New algorithms for optimal online checkpointing”. *SIAM Journal on Scientific Computing*, 32:836–854, 2010.

[33] Q. Wang, P. Moin, and G. Iaccarino. “Minimal repetition dynamic checkpointing algorithm for unsteady adjoint calculation”. *SIAM Journal of Scientific Computing*, 31:2549–2567, 2009.

[34] R. Muthukumar, D. P. Kouri, and M. Udell. “Randomized sketching algorithms for low-memory dynamic optimization”. *SIAM Journal on Optimization*, 31(2):1242–1275, 2021.

[35] R. J. Baraldi and D. P. Kouri. “A proximal trust-region method for nonsmooth optimization with inexact function and gradient evaluations”. *Mathematical Programming - submitted*, 2022.

[36] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich. *Optimization with PDE Constraints*, volume 23. Springer, New York, NY, 2009.

[37] J. A. Tropp, A. Yurtsever, M. Udell, and V. Cevher. “Streaming low-rank matrix approximation with an application to scientific simulations”. *SIAM Journal on Scientific Computing*, 41:A2430 – A2463, 2019.

[38] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust region methods*. SIAM, Philadelphia, PA, 2000.

[39] S. Garreis and M. Ulbrich. “An inexact trust-region algorithm for constrained problems in Hilbert space and its application to the adaptive solution of optimal control problems with PDEs”. *Preprint, submitted, Technical University of Munich*, 2019.

[40] 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.

[41] D. P. Kouri and D. Ridzal. “Inexact trust-region methods for PDE-constrained optimization”. In *Frontiers in PDE-Constrained Optimization*, pages 83–121. Springer, New York, NY, 2018.

[42] M. Heinkenschloss and L. N. Vicente. “Analysis of inexact trust–region SQP algorithms”. *SIAM J. Optimization*, 12:283–302, 2001.

[43] R. G. Carter. “Numerical optimization in Hilbert space using inexact function and gradient evaluations”. Technical Report 89-45, ICASE, Langley, VA, 1989.

[44] R. G. Carter. “On the global convergence of trust region algorithms using inexact gradient information”. *SIAM J. Numer. Anal.*, 28:251–265, 1991.

[45] 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.

[46] 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.

[47] E. Herberg and M. Hinze. “Variational discretization approach applied to an optimal control problem with bounded measure controls”. *Optimization and Control for Partial Differential Equations: Uncertainty quantification, open and closed-loop control, and shape optimization*, 29:113, 2022.