Sandia National Laboratories

Randomized Sketcing for Low-Memory Dynamic Optimization

Drew P. Kouri
Ramchandran Muthukumar, Madeleine Udell

Center for Mathematics and Artificial Intelligence Colloquium
George Mason University, Fairfax, VA

October 2, 2020

AFOSR AIR FORCE OFFICE OF SCIENTIFIC RESEARCH

LDRD

ENERGY NNSA

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.

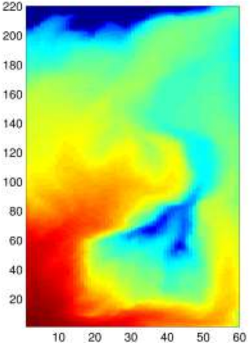
SAND2020-10685 PE

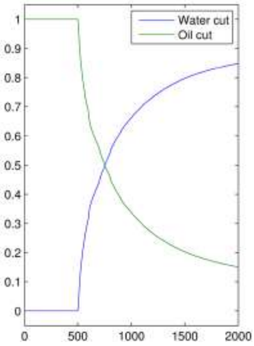
Outline

1. Motivating Applications
2. General Problem Formulation
3. Randomize Sketching for State Compression
4. Fixed Rank Analysis
5. Adaptive Rank Algorithm
6. Numerical Results

Operational Applications

Secondary Oil Recovery





$$\max_{v,p,s,q} \ \mathrm{NPV}(v,p,s,q)$$

subject to

$$v = -\mathbf{K}\lambda(s)(\nabla p - \rho G)$$

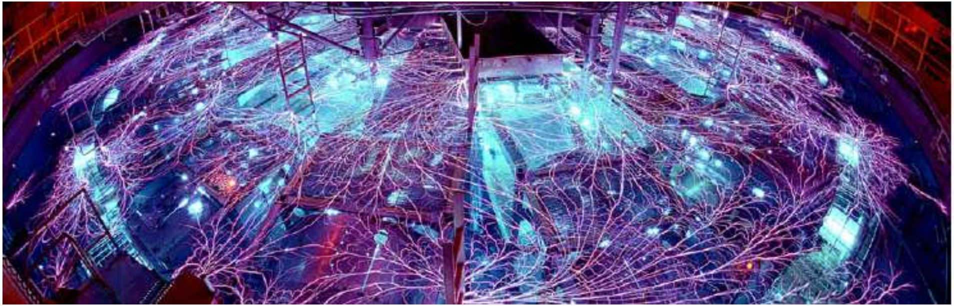
$$\nabla \cdot v = q/\rho$$

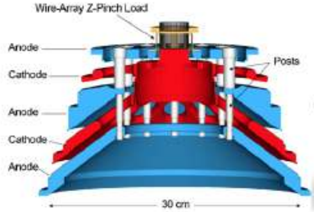
$$\phi\, \partial_t s + \nabla \cdot (f(s)v) = q$$

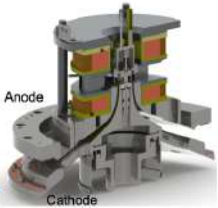
1. **Goal:** Determine injection rates that maximize the net present value of the reservior.
2. **Problem Size:** The reservoir can span 10+ km per side $\Longrightarrow$ 1000km^3
   A regular grid with 5m$\times$5m$\times$5m cells $\Longrightarrow$ 8 billion cells!
3. **Similar Applications:** Gas pipeline and energy network operations, financial portfolio optimization, optimal control of hypersonic vehicles, etc.

Optimal Design

Convolute Design for the Z-Machine (Inertial Confinement Fusion)





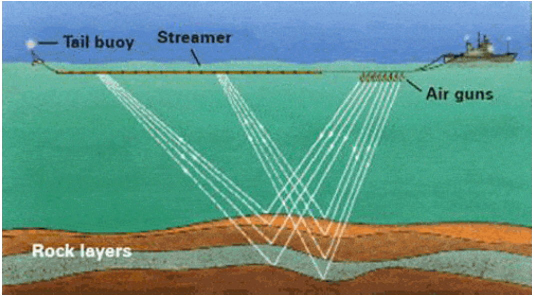


Goal: Minimize power loss – **current design experiences 20% power loss**

Similar Applications: Designing elastic structures with topology optimization, shape optimization for airplane wing design, etc.

Inverse Problems

Seismic Exploration



Courtesy of Schlumberger

$$\min_{u,\theta} \ \frac{1}{2}\|u-d\|^2 + \alpha\wp(\theta)$$

subject to

$$\partial_t\left(\varrho(\theta)\,\partial_t u\right) - \nabla\cdot\left(\mathbf{K}(\theta) : \varepsilon\right) = s$$

$$\varepsilon = \tfrac{1}{2}(\nabla u + \nabla u^\top)$$

1. **Goal:** Determine subsurface rock properties that minimize data misfit.
2. **Problem Size:** **Typical marine surveys require many TBs of memory!**
3. **Similar Applications:** Parameter calibration, data assimilation, supervised learning for neural networks, etc.

Discrete Dynamic Optimization

Full-Space Formulation

$$
\begin{aligned}
\min_{z_n\in\mathbb{R}^m,\ u_n\in\mathbb{R}^M} \quad & \sum_{n=1}^{N} f_n(u_{n-1}, u_n, z_n) \\
\text{subject to} \quad & c_n(u_{n-1}, u_n, z_n) = 0, \quad n = 1, \ldots, N
\end{aligned}
$$

where $f_n : \mathbb{R}^M \times \mathbb{R}^M \times \mathbb{R}^m \to \mathbb{R}$ is the ***cost function at time*** t_n,

$c_n : \mathbb{R}^M \times \mathbb{R}^M \times \mathbb{R}^m \to \mathbb{R}^M$ is the ***dynamic constraint at time*** t_n,

$u_n \in \mathbb{R}^M$ is the ***state/trajectory at time*** t_n, and

$z_n \in \mathbb{R}^m$ is the ***control/design/parameters at time*** t_n.

E.g., continuous dynamic problem discretized in time (and space).

Our methods also work for time-independent control/design/parameters.

Discrete Dynamic Optimization

The **KKT conditions** for the full-space dynamic optimization problem are

$$c_n(u_{n-1}, u_n, z_n) = 0$$

$$(\mathsf{d}_2 c_N(u_{N-1}, u_N, z_N))^\top \lambda_N = -\,\mathsf{d}_2 f_N(u_{N-1}, u_N, z_N)$$

$$(\mathsf{d}_2 c_n(u_{n-1}, u_n, z_n))^\top \lambda_n + (\mathsf{d}_1 c_{n+1}(u_n, u_{n+1}, z_{n+1}))^\top \lambda_{n+1} = -\,\mathsf{d}_2 f_N(u_{n-1}, u_n, z_n) - \mathsf{d}_1 f_{n+1}(u_n, u_{n+1}, z_{n+1})$$

$$\mathsf{d}_3 f_n(u_{n-1}, u_n, z_n) + (\mathsf{d}_3 c_n(u_{n-1}, u_n, z_n))^\top \lambda_n = 0, \quad n = 1, \ldots, N.$$

The new variables $\lambda_1, \ldots, \lambda_N$ are **Lagrange multipliers** for the dynamic constraint.

Most algorithms (e.g., SQP, AL, etc.) require storage of $\{u_n\}_{n=1}^N$, $\{\lambda_n\}_{n=1}^N$ and $\{z_n\}_{n=1}^N$.

Storage Requirement: $\mathcal{O}(N(2M+m))$ floating point numbers

$\Longrightarrow$ Can be **huge** (e.g., TBs of memory)!

Discrete Dynamic Optimization

If $c_n(u_{n-1}, u_n, z_n) = 0$ has a unique solution $u_n = S_n(u_{n-1}, z_n)$ for each $z \in Z_{\mathrm{ad}}$ and fixed initial state $u_0 \in \mathbb{R}^M$, $n = 1, \ldots, N$, then we can equivalently solve

$$\boxed{\min_{Z=\{z_n\}_{n=1}^N \in \mathbb{R}^{mN}} \quad F(Z)}$$

where the ***reduced*** objective function F is given by

$$\boxed{F(Z) := f_1(u_0, S_1(u_0, z_1), z_1) + \sum_{n=2}^{N} f_n(S_{n-1}(u_{n-2}, z_{n-1}), S_n(u_{n-1}, z_n), z_n).}$$

Storage Requirement: $\mathcal{O}(Nm)$ floating point numbers — typically, $m \ll M$.

Discrete Dynamic Optimization

Gradient Computation:

$$(\nabla F(Z))_n = \mathsf{d}_3 f_n(u_{n-1}, u_n, z_n) + (\mathsf{d}_3 c_n(u_{n-1}, u_n, z_n))^\top \lambda_n$$

where $\lambda_n \in \mathbb{R}^M$, $n = 1, \ldots, N$ solves the ***backward-in-time*** **adjoint equaiton**

$$\begin{aligned}
(\mathsf{d}_2 c_N(u_{N-1}, u_N, z_N))^\top \lambda_N &= -\,\mathsf{d}_2 f_N(u_{N-1}, u_N, z_N) \\
(\mathsf{d}_2 c_n(u_{n-1}, u_n, z_n))^\top \lambda_n &= -\,\mathsf{d}_2 f_N(u_{n-1}, u_n, z_n) - \mathsf{d}_1 f_{n+1}(u_n, u_{n+1}, z_{n+1}) \\
&\quad - (\mathsf{d}_1 c_{n+1}(u_n, u_{n+1}, z_{n+1}))^\top \lambda_{n+1}
\end{aligned}$$

Gradient computations requires entire state trajectory to solve adjoint equation!

Discrete Dynamic Optimization

1. **Checkpointing:** Store a handful of state snapshots in memory or to hard disk. Recompute state from snapshots during adjoint computation.

   Griewank and Walther, *Algorithm 799: Revolve: An implementation of checkpointing for scientific computing and optimal control*, 2000.

   Significant increase in computational cost!

Discrete Dynamic Optimization

1. **Checkpointing:** Store a handful of state snapshots in memory or to hard disk. Recompute state from snapshots during adjoint computation.

   Griewank and Walther, *Algorithm 799: Revolve: An implementation of checkpointing for scientific computing and optimal control*, 2000.

   Significant increase in computational cost!

2. **Model Reduction:** Project dynamic equation onto a low-order basis of the solution space (e.g., balanced truncation, POD, reduced basis, etc.).

   Benner, Sachs and Volkwein, *Model order reduction for PDE constrained optimization*, 2014.

   Difficult for nonlinear dynamics and requires significant code modification!

Discrete Dynamic Optimization

1. **Checkpointing:** Store a handful of state snapshots in memory or to hard disk. Recompute state from snapshots during adjoint computation.

   Griewank and Walther, *Algorithm 799: Revolve: An implementation of checkpointing for scientific computing and optimal control*, 2000.

   Significant increase in computational cost!

2. **Model Reduction:** Project dynamic equation onto a low-order basis of the solution space (e.g., balanced truncation, POD, reduced basis, etc.).

   Benner, Sachs and Volkwein, *Model order reduction for PDE constrained optimization*, 2014.

   Difficult for nonlinear dynamics and requires significant code modification!

3. **State Compression:** Compress dynamic trajectory using numerical compression techniques including nested meshes, PCA, Gramm-Schmidt and DFT.

   Cyr, Shadid and Wildey, *Towards efficient backward-in-time adjoint computations using data compression techniques*, 2015.

   Can be physics-dependent and often compresses time/space independently!

Randomized Sketching

We apply **randomized sketching** to compress the $M \times N$ state trajectory matrix

$$U = (u_1 \mid \cdots \mid u_N) \in \mathbb{R}^{M \times N}.$$

Goal: Generate an ***accurate*** rank-r approximation of U using $\mathcal{O}(r(M+N))$ storage.

Matrix Sketching: Given the four **random linear dimension reduction maps**

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

where $k := 2r + 1$ and $s = 2k + 1$, we define the sketch as the following three matrices

$$X := \Upsilon U \in \mathbb{R}^{k \times N} \qquad \text{the co-range sketch;}$$

$$Y := U \Omega^\top \in \mathbb{R}^{M \times k} \qquad \text{the range sketch;}$$

$$Z := \Phi U \Psi^\top \in \mathbb{R}^{s \times s} \qquad \text{the core sketch.}$$

Tropp, Yurtserver, Udell and Cevher, *Streaming low-rank matrix approximation with an application to scientific simulation*, 2019.

Randomized Sketching

Choice of Random Matrices: **Standard normal**, scrambled subsampled randomized Fourier transform, sparse sign, . . .

Intuition:

- The range sketch $Y = U\Omega^\top$ consists of columns $U\omega_i$, which are independent random samples from $\mathrm{range}(U)$.

  Roughly speaking, Y captures the action of the ***top*** k ***left singular vectors***.

- Similarly, the co-range sketch $X = \Upsilon U$ consists of rows $\upsilon_i^\top U$, which are independent random samples from $\mathrm{range}(U^\top)$.

  Roughly speaking, X captures the action of the ***top*** k ***right singular vectors***.

- ***Roughly speaking***, the core sketch $Z = \Phi U \Psi^\top$ captures the ***top*** k ***singular values***.

Online State Sketching:

Require: $X = 0$, $Y = 0$, $Z = 0$

1: **for** n=1,...,N **do**
2: Given u_{n-1} and z_n, solve $c_n(u_{n-1}, u_n, z_n) = 0$ for u_n
3: Update $X \leftarrow X + \Upsilon u_n e_n^\top$
4: Update $Y \leftarrow Y + u_n(\Omega e_n)^\top$
5: Update $Z \leftarrow Z + (\Phi u_n)(\Psi e_n)^\top$
6: **end for**

Storage Requirement: Sketching requires $k(M+N) + s^2$ floating point numbers.

Online State Sketching:

Require: $X = 0$, $Y = 0$, $Z = 0$

1: **for** n=1,. . . ,N **do**
2: Given u_{n-1} and z_n, solve $c_n(u_{n-1}, u_n, z_n) = 0$ for u_n
3: Update $X \leftarrow X + \Upsilon u_n e_n^\top$
4: Update $Y \leftarrow Y + u_n(\Omega e_n)^\top$
5: Update $Z \leftarrow Z + (\Phi u_n)(\Psi e_n)^\top$
6: **end for**

Storage Requirement: Sketching requires $k(M+N)+s^2$ floating point numbers.

Recovery:

1: $(Q, R_2) \leftarrow \mathtt{qr}(Y, 0)$ $\qquad Q \in \mathbb{R}^{M\times k},\quad R_2 \in \mathbb{R}^{k\times k}$
2: $(P, R_1) \leftarrow \mathtt{qr}(X^\top, 0)$ $\qquad P \in \mathbb{R}^{N\times k},\quad R_1 \in \mathbb{R}^{k\times k}$
3: $C \leftarrow (\Phi Q)^\dagger Z((\Psi P)^\dagger)^\top$ $\qquad C \in \mathbb{R}^{k\times k}$
4: $W \leftarrow CP^\top$ $\qquad W \in \mathbb{R}^{k\times N}$
5: $u_n \approx QWe_n$

Storage Requirement: Recovery requires $k(M+N)+k^2$ floating point numbers.

Theorem: Sketching Error Bound

Let $\{\{U\}\}_r$ denote the sketch of U associated with the rank parameter r. Then,

$$\mathbb{E}_{\Upsilon,\Omega,\Phi,\Psi}\|U - \{\{U\}\}_r\|_F \leq \sqrt{6}\Bigg(\sum_{i \geq r+1} \sigma_i^2(U)\Bigg)^{\frac{1}{2}}.$$

The error is *expected* to be slightly larger than the best rank-r approximation!

Similar results exist for the probability of large deviation.

Tropp, Yurtserver, Udell and Cevher, *Streaming low-rank matrix approximation with an application to scientific simulation*, 2019.

Solve Adjoint Equation and Compute Gradient:

Require: Sketched state $\{\{U\}\}_r$.

1: Reconstruct $\tilde{u}_{N-1}$ and $\tilde{u}_N$ from $\{\{U\}\}_r$.
2: Compute $\tilde{\lambda}_N$ that solves $(\mathrm{d}_2 c_N(\tilde{u}_{N-1}, \tilde{u}_N, z_N))^\top \tilde{\lambda}_N = -\mathrm{d}_2 f_N(\tilde{u}_{N-1}, \tilde{u}_N, z_N)$.
3: Set $g_N = \mathrm{d}_3 f_N(\tilde{u}_{N-1}, \tilde{u}_N, z_N) + (\mathrm{d}_3 c_N(\tilde{u}_{N-1}, \tilde{u}_N, z_N))^\top \tilde{\lambda}_N$.
4: **for** $n = N-1, \ldots, 1$ **do**
5: Reconstruct $\tilde{u}_{n-1}$ from $\{\{U\}\}_r$.
6: Compute $\tilde{\lambda}_n$ that solves

$$(\mathrm{d}_2 c_n(\tilde{u}_{n-1}, \tilde{u}_n, z_n))^\top \tilde{\lambda}_n = -\,\mathrm{d}_2 f_n(\tilde{u}_{n-1}, \tilde{u}_n, z_n) - \mathrm{d}_1 f_{n+1}(\tilde{u}_n, \tilde{u}_{n+1}, z_{n+1}) \\ - (\mathrm{d}_1 c_{n+1}(\tilde{u}_n, \tilde{u}_{n+1}, z_{n+1}))^\top \tilde{\lambda}_{n+1}.$$

7: Set $g_n = \mathrm{d}_3 f_n(\tilde{u}_{n-1}, \tilde{u}_n, z_n) + (\mathrm{d}_3 c_n(\tilde{u}_{n-1}, \tilde{u}_n, z_n))^\top \tilde{\lambda}_n$.
8: **end for**

Randomized Sketching

Define the combined objective function and constraint by

$$f(U,Z) = \sum_{i=1}^{N} f_i(u_{i-1}, u_i, z_i) \qquad \text{and} \qquad c(U,Z) = \begin{pmatrix} c_1(u_0, u_1, z_1) \\ \vdots \\ c_N(u_{N-1}, u_N, z_N) \end{pmatrix}.$$

Assumptions:

Define the combined objective function and constraint by

$$f(U,Z) = \sum_{i=1}^{N} f_i(u_{i-1},u_i,z_i) \qquad \text{and} \qquad c(U,Z) = \begin{pmatrix} c_1(u_0,u_1,z_1) \\ \vdots \\ c_N(u_{N-1},u_N,z_N) \end{pmatrix}.$$

Assumptions:

1. For any bounded set $\mathcal{Z} \subset \mathbb{R}^{Nm}$, the set $\mathcal{U} = \{U \,|\, c(U,Z) = 0,\ \ Z \in \mathcal{Z}\}$ is bounded;

Randomized Sketching

Define the combined objective function and constraint by

$$f(U,Z) = \sum_{i=1}^{N} f_i(u_{i-1}, u_i, z_i) \qquad \text{and} \qquad c(U,Z) = \begin{pmatrix} c_1(u_0, u_1, z_1) \\ \vdots \\ c_N(u_{N-1}, u_N, z_N) \end{pmatrix}.$$

Assumptions:

1. For any bounded set $\mathcal{Z} \subset \mathbb{R}^{Nm}$, the set $\mathcal{U} = \{U \,|\, c(U,Z) = 0, \;\; Z \in \mathcal{Z}\}$ is bounded;
2. There exists $0 < \sigma_0 \le \sigma_1 < \infty$ such that for all $U \in \mathcal{U}$ and $Z \in \mathcal{Z}$
$$\sigma_0 \le \sigma_{\min}(\mathrm{d}_1 c(U,Z)) \le \sigma_{\max}(\mathrm{d}_1 c(U,Z)) \le \sigma_1;$$

Define the combined objective function and constraint by

$$f(U,Z) = \sum_{i=1}^{N} f_i(u_{i-1}, u_i, z_i) \qquad \text{and} \qquad c(U,Z) = \begin{pmatrix} c_1(u_0, u_1, z_1) \\ \vdots \\ c_N(u_{N-1}, u_N, z_N) \end{pmatrix}.$$

Assumptions:

1. For any bounded set $\mathcal{Z} \subset \mathbb{R}^{Nm}$, the set $\mathcal{U} = \{U \,|\, c(U,Z) = 0, \;\; Z \in \mathcal{Z}\}$ is bounded;
2. There exists $0 < \sigma_0 \le \sigma_1 < \infty$ such that for all $U \in \mathcal{U}$ and $Z \in \mathcal{Z}$
$$\sigma_0 \le \sigma_{\min}(\mathrm{d}_1 c(U,Z)) \le \sigma_{\max}(\mathrm{d}_1 c(U,Z)) \le \sigma_1;$$
3. The functions $\mathrm{d}_1 c(U,Z)$, $\mathrm{d}_2 c(U,Z)$, $\mathrm{d}_1 f(U,Z)$ and $\mathrm{d}_2 f(U,Z)$ are Lipschitz continuous on $\mathcal{U} \times \mathcal{Z}$ with respect to U and the Lipschitz modulii are independent of $Z \in \mathcal{Z}$.

Randomized Sketching

Define the combined objective function and constraint by

$$f(U,Z) = \sum_{i=1}^{N} f_i(u_{i-1},u_i,z_i) \qquad \text{and} \qquad c(U,Z) = \begin{pmatrix} c_1(u_0,u_1,z_1) \\ \vdots \\ c_N(u_{N-1},u_N,z_N) \end{pmatrix}.$$

Assumptions:

1. For any bounded set $\mathcal{Z} \subset \mathbb{R}^{Nm}$, the set $\mathcal{U} = \{U \,|\, c(U,Z) = 0, \;\; Z \in \mathcal{Z}\}$ is bounded;
2. There exists $0 < \sigma_0 \leq \sigma_1 < \infty$ such that for all $U \in \mathcal{U}$ and $Z \in \mathcal{Z}$
$$\sigma_0 \leq \sigma_{\min}(\mathrm{d}_1 c(U,Z)) \leq \sigma_{\max}(\mathrm{d}_1 c(U,Z)) \leq \sigma_1;$$
3. The functions $\mathrm{d}_1 c(U,Z)$, $\mathrm{d}_2 c(U,Z)$, $\mathrm{d}_1 f(U,Z)$ and $\mathrm{d}_2 f(U,Z)$ are Lipschitz continuous on $\mathcal{U} \times \mathcal{Z}$ with respect to U and the Lipschitz moduli are independent of $Z \in \mathcal{Z}$.

Under these assumptions, we can show that

$$\|g - \nabla F(Z)\| \leq \kappa_1 \underbrace{\|c(\{\{U\}\}_r, Z)\|}_{\text{state residual}} \qquad \text{and} \qquad \mathbb{E}\|g - \nabla F(Z)\| \leq \sqrt{6}\kappa \Big(\sum_{i \geq r+1} \sigma_i^2(U) \Big)^{\frac{1}{2}}.$$

Fixed Rank Approach

- **Idea:** Use approximate gradient within a **derivative-based optimization algorithm**. For example, gradient descent, nonlinear CG, Newton–Krylov method, etc.
- If the rank parameter r is sufficiently large, then gradient error **may be negligible**.
- For Newton–Krylov, applying the Hessian to a vector requires **a linearized forward-in-time solve** and **a linearized backward-in-time solve**.
- Can **approximately** apply the Hessian to a vector by **sketching the adjoint as well as the trajectories from the additional solves**.

Fixed Rank Approach

Basic Trust-Region Method

Require: Initial guess $Z^{(0)} \in \mathbb{R}^{Nm}$ and target rank r. Algorithmic parameters: $\Delta^{(0)} > 0$ and $0 < \eta_0 \leq \eta_1 < \eta_2 < 1$.

1: **for** $\ell = 0, 1, 2, \ldots$ **do**

2: Solve state equation and sketch solution $\{\{U^{(\ell)}\}\}_r$.

3: Compute gradient approximation $g^{(\ell)} \approx \nabla F(Z^{(\ell)})$ using sketched state $\{\{U^{(\ell)}\}\}_r$.

4: Choose $B^{(\ell)} \in \mathbb{R}^{Nm \times Nm}$ to be an approximation of the Hessian $\nabla^2 F(Z^{(\ell)})$.

5: Compute $S^{(\ell)} \in \mathbb{R}^{Nm}$ that approximately solves the trust-region subproblem

$$\min_{S \in \mathbb{R}^{Nm}} \frac{1}{2}\langle S, B^{(\ell)} S\rangle + \langle S, g^{(\ell)}\rangle \quad \text{subject to} \quad \|S\| \leq \Delta^{(\ell)}.$$

6: Compute the ratio of actual and predicted reduction

$$\rho^{(\ell)} = \frac{F(Z^{(\ell)}) - F(Z^{(\ell)} + S^{(\ell)})}{-\frac{1}{2}\langle S^{(\ell)}, B^{(\ell)} S^{(\ell)}\rangle - \langle S^{(\ell)}, g^{(\ell)}\rangle}.$$

7: Set $Z^{(\ell+1)} = \begin{cases} Z^{(\ell)} & \text{if } \rho^{(\ell)} \leq \eta_0 \\ Z^{(\ell)} + S^{(\ell)} & \text{otherwise.} \end{cases}$

8: Choose $\Delta^{(\ell+1)} \in \begin{cases} (0, \Delta^{(\ell)}) & \text{if } \rho^{(\ell)} \leq \eta_1 \\ \{\Delta^{(\ell)}\} & \text{if } \rho^{(\ell)} \in (\eta_1, \eta_2) \\ (\Delta^{(\ell)}, \infty) & \text{otherwise.} \end{cases}$

9: **end for**

Fixed Rank Approach

Require: Initial guess $Z^{(0)} \in \mathbb{R}^{Nm}$ and target rank r. Algorithmic parameters: $\Delta^{(0)} > 0$ and $0 < \eta_0 \le \eta_1 < \eta_2 < 1$.

1: **for** $\ell = 0, 1, 2, \ldots$ **do**

2: Solve state equation and sketch solution $\{\{U^{(\ell)}\}\}_r$.

3: **Compute gradient approximation $g^{(\ell)} \approx \nabla F(Z^{(\ell)})$ using sketched state $\{\{U^{(\ell)}\}\}_r$.**

4: Choose $B^{(\ell)} \in \mathbb{R}^{Nm \times Nm}$ to be an approximation of the Hessian $\nabla^2 F(Z^{(\ell)})$.

5: Compute $S^{(\ell)} \in \mathbb{R}^{Nm}$ that approximately solves the trust-region subproblem

$$\min_{S \in \mathbb{R}^{Nm}} \frac{1}{2}\langle S, B^{(\ell)} S\rangle + \langle S, g^{(\ell)}\rangle \quad \text{subject to} \quad \|S\| \le \Delta^{(\ell)}.$$

6: Compute the ratio of actual and predicted reduction

$$\rho^{(\ell)} = \frac{F(Z^{(\ell)}) - F(Z^{(\ell)} + S^{(\ell)})}{-\frac{1}{2}\langle S^{(\ell)}, B^{(\ell)} S^{(\ell)}\rangle - \langle S^{(\ell)}, g^{(\ell)}\rangle}.$$

7: Set $Z^{(\ell+1)} = \begin{cases} Z^{(\ell)} & \text{if } \rho^{(\ell)} \le \eta_0 \\ Z^{(\ell)} + S^{(\ell)} & \text{otherwise.} \end{cases}$

8: Choose $\Delta^{(\ell+1)} \in \begin{cases} (0, \Delta^{(\ell)}) & \text{if } \rho^{(\ell)} \le \eta_1 \\ \{\Delta^{(\ell)}\} & \text{if } \rho^{(\ell)} \in (\eta_1, \eta_2) \\ (\Delta^{(\ell)}, \infty) & \text{otherwise.} \end{cases}$

9: **end for**

Algorithm may not converge becuase gradient is inexact!

Adaptive Rank Algorithm

1. Fixed-rank algorithm may not converge unless the rank r is chosen sufficiently large.
   A priori choice of r is often difficult for practical applications!

Adaptive Rank Algorithm

1. Fixed-rank algorithm may not converge unless the rank r is chosen sufficiently large.
   A priori choice of r is often difficult for practical applications!

2. To ensure convergence of the TR method, gradient inexactness must be controlled

$$\|g^{(\ell)} - \nabla F(Z^{(\ell)})\| \le \kappa_0 \min\{\|g^{(\ell)}\|, \Delta^{(\ell)}\}.$$

Heinkenschloss and Vicente, *Analysis of incexact SQP algorithms*, 2001.

Adaptive Rank Algorithm

1. Fixed-rank algorithm may not converge unless the rank r is chosen sufficiently large.
   A priori choice of r is often difficult for practical applications!

2. To ensure convergence of the TR method, gradient inexactness must be controlled

$$\|g^{(\ell)} - \nabla F(Z^{(\ell)})\| \leq \kappa_0 \min\{\|g^{(\ell)}\|, \Delta^{(\ell)}\}.$$

   Heinkenschloss and Vicente, *Analysis of incexact SQP algorithms*, 2001.

3. Satisfy gradient inexactness condition by monitoring state residual

$$\|g^{(\ell)} - \nabla F(Z^{(\ell)})\| \leq \kappa_1 \|c(\tilde{U}^{(\ell)}, Z^{(\ell)})\| \quad \Longrightarrow \quad \|c(\tilde{U}^{(\ell)}, Z^{(\ell)})\| \leq \kappa_0' \min\{\|g^{(\ell)}\|, \Delta^{(\ell)}\}.$$

Adaptive Rank Algorithm

1. Fixed-rank algorithm may not converge unless the rank r is chosen sufficiently large.
   A priori choice of r is often difficult for practical applications!

2. To ensure convergence of the TR method, gradient inexactness must be controlled

$$\|g^{(\ell)} - \nabla F(Z^{(\ell)})\| \le \kappa_0 \min\{\|g^{(\ell)}\|, \Delta^{(\ell)}\}.$$

   Heinkenschloss and Vicente, *Analysis of incexact SQP algorithms*, 2001.

3. Satisfy gradient inexactness condition by monitoring state residual

$$\|g^{(\ell)} - \nabla F(Z^{(\ell)})\| \le \kappa_1 \|c(\tilde{U}^{(\ell)}, Z^{(\ell)})\| \quad \Longrightarrow \quad \|c(\tilde{U}^{(\ell)}, Z^{(\ell)})\| \le \kappa_0' \min\{\|g^{(\ell)}\|, \Delta^{(\ell)}\}.$$

4. Evaluate state residual while solving adjoint equation.
   If bound is violated, increase r, re-solve state equation and re-sketch state!

Adaptive Rank Algorithm

1. Fixed-rank algorithm may not converge unless the rank r is chosen sufficiently large.
   A priori choice of r is often difficult for practical applications!

2. To ensure convergence of the TR method, gradient inexactness must be controlled

$$\|g^{(\ell)} - \nabla F(Z^{(\ell)})\| \leq \kappa_0 \min\{\|g^{(\ell)}\|, \Delta^{(\ell)}\}.$$

   Heinkenschloss and Vicente, *Analysis of incexact SQP algorithms*, 2001.

3. Satisfy gradient inexactness condition by monitoring state residual

$$\|g^{(\ell)} - \nabla F(Z^{(\ell)})\| \leq \kappa_1 \|c(\tilde{U}^{(\ell)}, Z^{(\ell)})\| \quad \Longrightarrow \quad \|c(\tilde{U}^{(\ell)}, Z^{(\ell)})\| \leq \kappa_0' \min\{\|g^{(\ell)}\|, \Delta^{(\ell)}\}.$$

4. Evaluate state residual while solving adjoint equation.
   If bound is violated, increase r, re-solve state equation and re-sketch state!

5. Hopefully sufficiently large r is determined after a handful of updates.
   May run out of memory if rank of state is too large!

Theorem: Convergence of Adaptive-Rank Trust-Region Method

Suppose there exists a bounded, open, convex set $\mathcal{Z} \subset \mathbb{R}^{Nm}$ such that $Z^{(\ell)} \in \mathcal{Z}$ for all ℓ. Suppose F is bounded below and twice continuously differentiable on $\mathcal{Z}$. Moreover, assume that $\nabla^2 F(\cdot)$ is uniformly bounded on $\mathcal{Z}$ and that $B^{(\ell)}$ is bounded independently of ℓ. Then,

$$\liminf_{\ell\to\infty} \|g^{(\ell)}\| = \liminf_{\ell\to\infty} \|\nabla F(Z^{(\ell)})\| = 0.$$

For detailed analysis of inexact trust-region methods see:

Carter, *On the global convergence of trust region algorithms using inexact gradient information*, 1991.

Conn, Gould and Toint, *Trust Region Methods*, 2000.

Heinkenschloss and Vicente, *Analysis of incexact SQP algorithms*, 2001.

Kouri and Ridzal, *Inexact trust-region methods for PDE-constrained optimization*, 2018.

Adaptive Rank Algorithm

"Simple" Control Constraints: Constraints with ***easy-to-compute*** projections such as

$$a_0 \le CZ \le a_1 \qquad \text{and} \qquad DZ = b.$$

Only need to modify trust-region subproblem to account for constraints in the set Z_{ad}, i.e.,

$$\min_{S \in \mathbb{R}^{Nm}} \ \frac{1}{2}\langle S, B^{(\ell)} S\rangle + \langle S, g^{(\ell)}\rangle \quad \text{subject to} \quad Z^{(\ell)} + S \in Z_{\mathrm{ad}}, \quad \|S\| \le \Delta^{(\ell)}.$$

Garreis and Ulbrich, *An inexact trust-region algorith for constrained problems in Hilbert space and its application to the adaptive solution of optimal control problems with PDEs*, 2019.

Adaptive Rank Algorithm

"Simple" Control Constraints: Constraints with ***easy-to-compute*** projections such as

$$a_0 \le CZ \le a_1 \qquad \text{and} \qquad DZ = b.$$

Only need to modify trust-region subproblem to account for constraints in the set Z_{ad}, i.e.,

$$\min_{S \in \mathbb{R}^{Nm}} \ \tfrac{1}{2}\langle S, B^{(\ell)} S\rangle + \langle S, g^{(\ell)}\rangle \quad \text{subject to} \quad Z^{(\ell)} + S \in Z_{\mathrm{ad}}, \quad \|S\| \le \Delta^{(\ell)}.$$

Garreis and Ulbrich, *An inexact trust-region algorith for constrained problems in Hilbert space and its application to the adaptive solution of optimal control problems with PDEs*, 2019.

General Constraints: Use augmented Lagrangian, log barrier, etc. to penalize general nonlinear constraints such as

$$g(U,Z) = 0 \qquad \text{and} \qquad h(U,Z) \le 0.$$

Apply the adaptive rank algorithm to solve the penalized subproblems

$$\min_{Z \in \mathbb{R}^{Nm}} \ F(Z) + \wp^{(\ell)}(Z).$$

Example Application

Let $\mathcal{L}$ be a uniformly bounded and coercive linear operator, B a bounded linear operator and β a continuous linear functional. Consider the optimal control problem

$$\begin{aligned} \min_{u,\, z} \; & \frac{1}{2}\int_0^T \|u - w\|^2 \,\mathrm{d}t + \frac{\alpha}{2}\int_0^T \|z\|^2 \,\mathrm{d}t \\ \text{subject to} \quad & \partial_t u + \mathcal{L}(t)u = \beta(t) + Bz \\ & u(0) = u_0. \end{aligned}$$

Discretization

1. Continuous finite elements in space and implicit Euler in time.
2. Similar results hold for trapezoidal (Crank-Nicolson) and explicit Euler.

Stability Estimates: C_n^1, C_n^2 are positive constants that depend on the time step δt_n

$$\|\tilde{u}_n - u_n\| \le C_n^1 \sum_{i=1}^n \|c_i(\tilde{u}_{i-1}, \tilde{u}_i, z_i)\| \quad \text{and} \quad \|\tilde{\lambda}_n - \lambda_n\| \le C_n^2 \sum_{i=1}^n \|c_i(\tilde{u}_{i-1}, \tilde{u}_i, z_i)\|.$$

Using these estimates, we can prove convergence of adaptive rank algorithm!

Numerical Results

Let $D = (0, 0.5) \times (0, 0.2)$ with boundary ∂D and consider the optimal control problem

$$\begin{aligned} &\min_{u,\,z} \frac{1}{2}\int_0^T \int_D |u-1|^2 \,\mathrm{d}x\mathrm{d}t + \frac{\alpha}{2}\int_0^T \int_D |z|^2 \,\mathrm{d}x\mathrm{d}t \\ &\text{subject to} \quad \partial_t u - \gamma \Delta u + \mathbf{v}\cdot\nabla u + u = \beta + z \qquad \text{in } (0,T)\times D \\ &\qquad\qquad\qquad\qquad \gamma \nabla u \cdot n = 0 \qquad \text{on } (0,T)\times \partial D \\ &\qquad\qquad\qquad\qquad u(0) = 0 \qquad \text{in } D, \end{aligned}$$

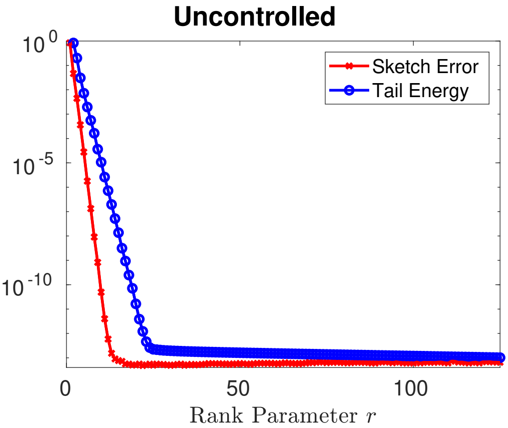
where $\alpha = 10^{-4}$, $\gamma = 0.1$, $\mathbf{v}(x) = (7.5 - 2.5x_1, 2.5x_2)^\top$, and

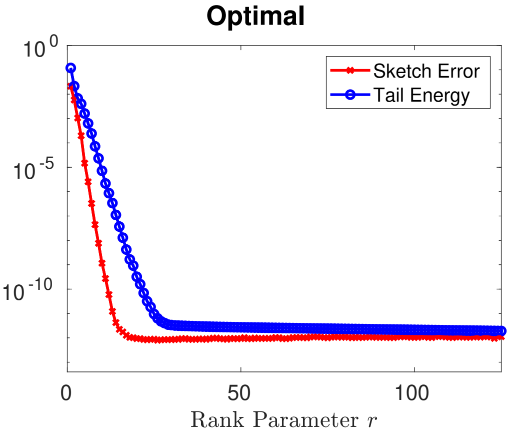
$$\beta(x) = \begin{cases} 1 & \text{if } \|x - (0.1, 0.1)^\top\| \le 0.07 \\ 0 & \text{otherwise.} \end{cases}$$

Spatial Discretization: Q1 finite elements on a uniform mesh of 60×20 quadrilaterals.

Temporal Discretization: Implicit Euler with 500 equal time steps.

Numerical Results





Sketching error averaged over 20 realizations and **tail energy** for the uncontrolled state (left) and the optimal state (right). Recall that the rank of the sketch is $k = 2r + 1$.

Numerical Results

Termination Criterion: $\|g^{(\ell)}\| \le 10^{-7}$ or $\ell > 20$.

Rank Increase Function: $r \leftarrow \max\{r+2, \lceil (b - \log\tau)/a \rceil\}$ with $a = 2.6125$, $b = 2.4841$.

rank	objective	iter	nstate	nadjoint	iterCG	compression
*1	5.544040e-4	20	21	11	196	118.79
2	5.528490e-4	5	6	6	151	70.96
3	5.528490e-4	4	5	5	78	50.46
4	5.528490e-4	4	5	5	67	38.08
5	5.528490e-4	4	5	5	59	31.83
Adaptive	5.528490e-4	4	8	5	65	23.14
Full	5.528490e-4	4	5	5	53	1.00

*The rank 1 experiment exceeded the maximum number of iterations.

Numerical Results

Linear Parabolic Control

	iter	value	gnorm	snorm	delta	iterCG	rank	rnorm
Adaptive	0	5.446e-2	5.990e-3	---	1.000e+1	---	1	2.707e-4
	1	1.375e-2	2.205e-3	1.000e+1	2.500e+1	1	1	1.990e-4
	2	1.475e-3	1.408e-4	2.499e+1	6.250e+1	5	5	6.700e-7
	3	5.531e-4	6.431e-7	4.077e+1	1.563e+1	27	7	3.893e-9
	4	5.528e-4	5.039e-9	1.059e+0	3.906e+2	32	7	1.508e-9
Full	0	5.446e-2	5.989e-3	---	1.000e+1	---	---	---
	1	1.375e-2	2.201e-3	1.000e+1	2.500e+1	1	---	---
	2	1.472e-3	1.401e-4	2.500e+1	6.250e+1	5	---	---
	3	5.538e-4	1.361e-6	4.051e+1	1.563e+1	19	---	---
	4	5.528e-4	8.416e-9	2.178e+0	3.906e+2	28	---	---

Adpative-rank algorithm reduced memory requirement 23.14-fold!

We consider the optimal flow control problem

$$\min_z \int_0^T \left\{ \int_{\partial C} \left(\frac{1}{\mathrm{Re}} \frac{\partial \mathbf{v}}{\partial n} - pn \right) \cdot (z\tau - \mathbf{v}_\infty)\,\mathrm{d}x + \frac{\alpha}{2} z(t)^2 \right\} \mathrm{d}t,$$

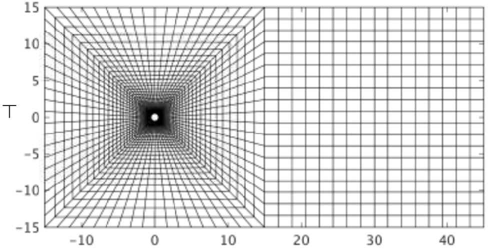
where the velocity/pressure $(\mathbf{v}, p) : [0,T] \times D \to \mathbb{R}^2 \times \mathbb{R}$ solves the Navier-Stokes equations

$$\begin{aligned}
\frac{\partial \mathbf{v}}{\partial t} - \frac{1}{\mathrm{Re}} \Delta \mathbf{v} + (\mathbf{v} \cdot \nabla)\mathbf{v} + \nabla p &= 0 && \text{in } (0,T) \times D \setminus C \\
\nabla \cdot \mathbf{v} &= 0 && \text{in } (0,T) \times D \setminus C \\
\frac{1}{\mathrm{Re}} \frac{\partial \mathbf{v}}{\partial n} - pn &= 0 && \text{on } (0,T) \times \Gamma_{\mathrm{out}} \\
\mathbf{v} &= \mathbf{v}_\infty && \text{on } (0,T) \times \partial D \setminus \Gamma_{\mathrm{out}} \\
\mathbf{v} &= z\tau && \text{on } (0,T) \times \partial C
\end{aligned}$$

Goal: Minimize the **power required to overcome the drag** on C by rotating C.

$$\mathbf{v}_2 = 0$$

$\mathbf{v} = (0,1)^\top$



Γ_{out}

$$\mathbf{v}_2 = 0$$

Problem Data: $D = (-15, 45) \times (-15, 15)$, $C = B_{1/2}$, $\mathrm{Re} = 200$, $T = 20$, and tangent vector

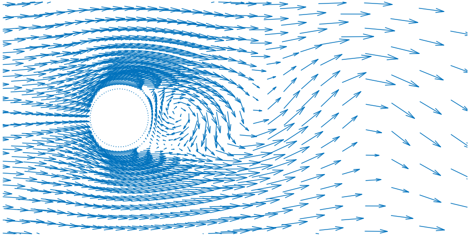
$$\tau(x) = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} (x - x_0), \quad x \in \partial C.$$

Spatial Discretization: Q2–Q1 finite elements.

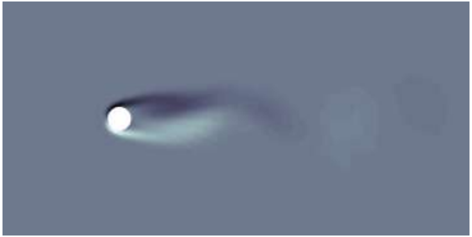
Temporal Discretization: Implicit Euler with 800 equal time steps.

Initial Vorticity ($t = 0$)

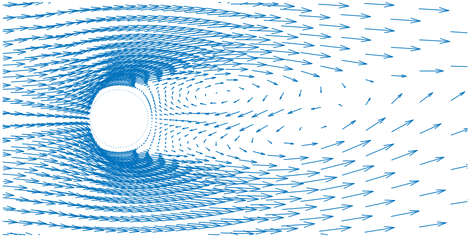
Initial Velocity ($t = 0$)



Controlled Vorticity ($t = 20$)

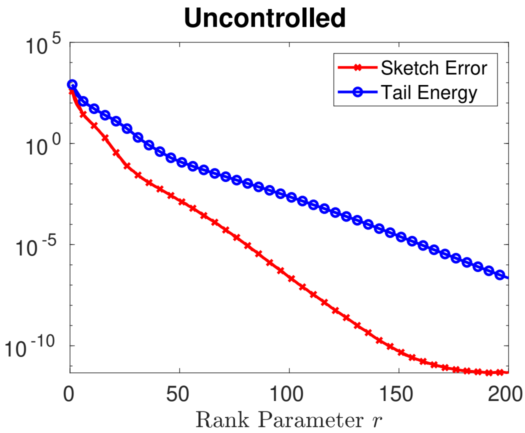


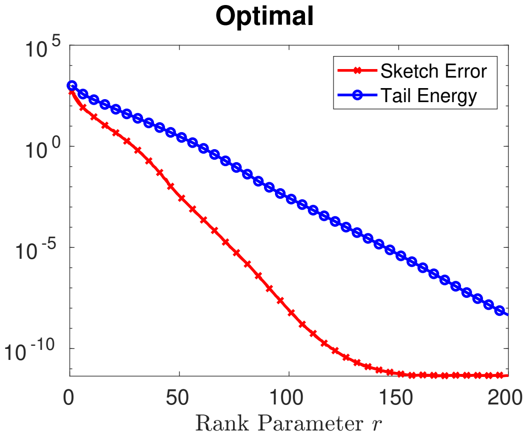
Controlled Velocity ($t = 20$)



Numerical Results

Flow Control





Sketching error averaged over 20 realizations and **tail energy** for the uncontrolled state (left) and the optimal state (right). Recall that the rank of the sketch is $k = 2r + 1$.

Numerical Results

Termination Criterion: $\|g^{(\ell)}\| \le 10^{-5}$ or $\ell > 40$.

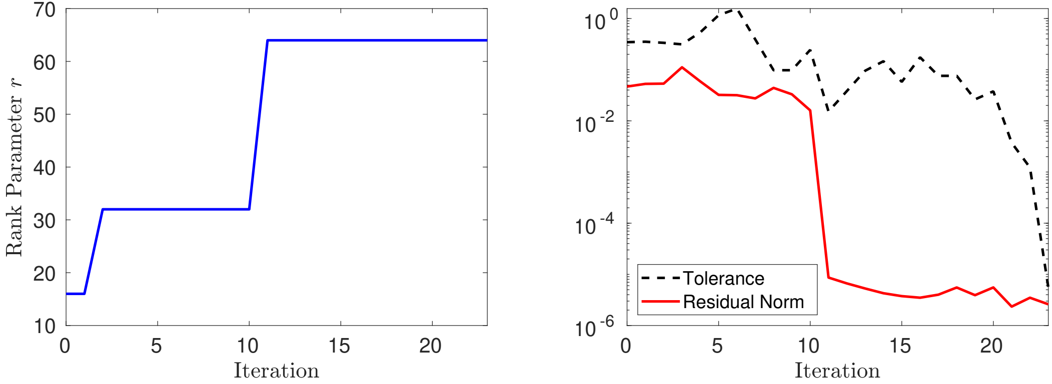
Rank Increase Function: $r \leftarrow 2r$.

rank	objective	iter	nstate	nadjoint	iterCG	compression
* 8	18.35919	40	41	15	136	45.44
*16	18.20003	40	41	33	897	23.35
*32	18.19779	40	41	31	236	11.80
64	18.19779	29	41	34	110	5.88
Adaptive	18.19779	23	27	24	121	5.88
Full	18.19779	29	30	24	107	---

*The rank 8, 16, and 32 experiments exceeded the maximum number of iterations.

Numerical Results

Flow Control



Left: **Sketch rank** as a function of iteration of adaptive-rank algorithm. **Right:** Required **gradient inexactness tolerance** and computed **residual norm** as functions of iteration.

Adpative-rank algorithm reduced memory requirement 5.88-fold!

Conclusions

- **Numerical solution** of dynamic optimization is **expensive** and requires **careful memory management**.
- We **compress** the dynamic state using **randomized matrix sketching**.
- State compression using sketching is **lossy** and results in **inexact gradients**.
- Introduced provably convergent algorithm to handle gradient inexactness. Algorithm adatively learns the rank of the optimal state — **May run out of memory for non-low-rank states!**
- Numerical examples suggest $\sim$**5–70-fold reduction** in memory requirement.

R. Muthukumar, D. P. Kouri and M. Udell, *Randomized sketching Algorithms for low-memory dynamic optimization*, 2019. `http://www.optimization-online.org/DB_FILE/2019/11/7475.pdf`