Quantum-inspired classical algorithms for linear equations and beyond

Changpeng Shao

School of Mathematics, University of Bristol, UK

20 July 2021

arXiv:2103.10309 (joint work with Ashley Montanaro),
arXiv:2105.07736

EPSRC
Engineering and Physical Sciences Research Council

European Research Council
Established by the European Commission

Backgrounds

Solving linear systems is a fundamental problem in machine learning

Backgrounds

Solving linear systems is a fundamental problem in machine learning

In the field of quantum computing, many algorithms are known, eg

- Harrow-Hassidim-Lloyd algorithm for sparse linear systems
- Block-encoding method for general linear systems

Backgrounds

Solving linear systems is a fundamental problem in machine learning

In the field of quantum computing, many algorithms are known, eg

- Harrow-Hassidim-Lloyd algorithm for sparse linear systems
- Block-encoding method for general linear systems

A useful way to construct block-encoding of a matrix is based on the QRAM data structure

Backgrounds

Solving linear systems is a fundamental problem in machine learning

In the field of quantum computing, many algorithms are known, eg

- Harrow-Hassidim-Lloyd algorithm for sparse linear systems
- Block-encoding method for general linear systems

A useful way to construct block-encoding of a matrix is based on the QRAM data structure

But assuming access to QRAM, there exist (quantum-inspired) classical algorithms that can also achieve poly-log dependence on the dimension [Ewin Tang 2018]

Backgrounds

Solving linear systems is a fundamental problem in machine learning

In the field of quantum computing, many algorithms are known, eg

- Harrow-Hassidim-Lloyd algorithm for sparse linear systems
- Block-encoding method for general linear systems

A useful way to construct block-encoding of a matrix is based on the QRAM data structure

But assuming access to QRAM, there exist (quantum-inspired) classical algorithms that can also achieve poly-log dependence on the dimension [Ewin Tang 2018]

As a result, exponential quantum speedups for many machine learning problems reduce to polynomial quantum speedups

Backgrounds

Solving linear systems is a fundamental problem in machine learning

In the field of quantum computing, many algorithms are known, eg

- Harrow-Hassidim-Lloyd algorithm for sparse linear systems
- Block-encoding method for general linear systems

A useful way to construct block-encoding of a matrix is based on the QRAM data structure

But assuming access to QRAM, there exist (quantum-inspired) classical algorithms that can also achieve poly-log dependence on the dimension [Ewin Tang 2018]

As a result, exponential quantum speedups for many machine learning problems reduce to polynomial quantum speedups

Question: How large is this polynomial speedup?

Main results

We focus on the solving of linear systems $A\mathbf{x} = \mathbf{b}$ with QRAM:

Algorithm	Complexity	Ref.	Assumptions
Quantum	$\widetilde{O}(\kappa_F)$	[1]	
Randomized	$\widetilde{O}(s\kappa_F^2)$	[2]	row sparse
classical	$\widetilde{O}(s\mathrm{Tr}(A)\|A^+\|)$	[3]	sparse, SPD
Quantum-inspired	$\widetilde{O}(\kappa_F^6\kappa^6/\epsilon^4)$	[4]	
classical	$\widetilde{O}(\kappa_F^6\kappa^2/\epsilon^2)$	Our	
	$\widetilde{O}(\mathrm{Tr}(A)^3\|A^+\|^3\kappa/\epsilon^2)$	Our	SPD

$\kappa_F = \|A\|_F\|A^+\|$,[1] $\quad \kappa = \|A\|\|A^+\|$, $\quad s =$ row sparsity,
SPD = symmetric positive definite.

[1] Chakraborty, Gilyén, Jeffery 2018 [2] Strohmer, Vershynin 2009
[3] Leventhal, Lewis 2010 [4] Gilyén, Song, Tang 2020

[1] $\|A\|_F^2 = \sum_{i,j} |A_{ij}|^2$, $\quad A^+ =$ pseudoinverse of A

Implication 1: A is row sparse

Classical:
QRAM model $\Rightarrow \widetilde{O}(s\|A\|_F^2\|A^+\|^2)$

Quantum:

- column sparse + sparse access input model
  $\Rightarrow \widetilde{O}(s\|A\|_{\max}\|A^+\|)$ [2]
  $\Rightarrow$ large quantum speedup is possible
- column dense + QRAM model
  $\Rightarrow \widetilde{O}(\|A\|_F\|A^+\|)$
  $\Rightarrow$ quadratic quantum speedup when $s = \widetilde{O}(1)$

Both achieve poly-log dependence on ϵ

[2] $\|A\|_{\max} = \max_{i,j} |A_{ij}|$

Implication 2: A is row sparse

The output:

- **Classical:** a sparse vector
- **Quantum:** a quantum state

If to estimate the norm of the solution (Assume QRAM + column dense):

- **Classical:** $\widetilde{O}(s\|A\|_F^2\|A^+\|^2)$
- **Quantum:** $\widetilde{O}(\|A\|_F\|A^+\|/\epsilon)$

$\Rightarrow$ Classical algorithm is better in terms of the dependence on ϵ

Implication 3: A is row sparse and SPD

Classical:

- $\widetilde{O}(s\mathrm{Tr}(A)\|A^+\|)$
- the output is a sparse vector

Quantum:

- $\widetilde{O}(s\|A\|_{\max}\|A^+\|)$
- the output is a quantum state

$\Rightarrow$ May have no quantum speedup if $\mathrm{Tr}(A) = \widetilde{\Theta}(\|A\|_{\max})$.

Good news: For some SPD linear systems, quantum computers can have quadratically better dependence on the condition number [Davide, Dunjko, arXiv:2101.11868]

Implication 4: A is dense in the QRAM model

Classical: $\widetilde{O}(\|A\|_F^6\|A\|^2\|A^+\|^8/\epsilon^2)$

This improves the previous best result $\widetilde{O}(\|A\|_F^6\|A\|^6\|A^+\|^{12}/\epsilon^4)$ of Gilyén, Song, Tang 2020

The output is a classical analogue of the quantum output

Quantum: $\widetilde{O}(\|A\|_F\|A^+\|)$

$\Rightarrow$ Large polynomial quantum speedup still exists.

The main technique: Kaczmarz method

Background:

- first discovered by the Polish mathematician Stefan Kaczmarz in 1937.

- rediscovered in the field of image reconstruction from projections by Richard Gordon, Robert Bender, and Gabor Herman in 1970.

The main technique: Kaczmarz method

It is an iterative algorithm for solving linear equation systems $A\mathbf{x} = \mathbf{b}$

The main technique: Kaczmarz method

It is an iterative algorithm for solving linear equation systems $A\mathbf{x} = \mathbf{b}$

Assume A is $m \times n$. Let $\mathbf{x}_0$ be an arbitrary initial approximation to the solution. For $k = 0, 1, \ldots$, compute

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{b_{r_k} - \langle A_{r_k *} | \mathbf{x}_k \rangle}{\|A_{r_k *}\|^2} A_{r_k *},$$

where $A_{r_k *}$ is the r_k-th row of A, and r_k is chosen from $\{1, ..., m\}$ randomly with probability proportional to $\|A_{r_k *}\|^2$

The main technique: Kaczmarz method

It is an iterative algorithm for solving linear equation systems $A\mathbf{x} = \mathbf{b}$

Assume A is $m \times n$. Let $\mathbf{x}_0$ be an arbitrary initial approximation to the solution. For $k = 0, 1, \ldots$, compute

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{b_{r_k} - \langle A_{r_k *} | \mathbf{x}_k \rangle}{\|A_{r_k *}\|^2} A_{r_k *},$$

where $A_{r_k *}$ is the r_k-th row of A, and r_k is chosen from $\{1, ..., m\}$ randomly with probability proportional to $\|A_{r_k *}\|^2$

It naturally has a sampling-based data structure
A stochastic gradient descent method: $\min \sum_i \|\langle A_{i*} | \mathbf{x} \rangle - b_i\|^2$

[Strohmer, Vershynin, J Fourier Anal Appl, 2009]

The main technique: Kaczmarz method

The Kaczmarz algorithm has a clear geometric meaning: $\mathbf{x}_{k+1}$ is the orthogonal projection of $\mathbf{x}_k$ onto the hyperplane $A_{r_k *} \cdot \mathbf{x} = b_{r_k}$

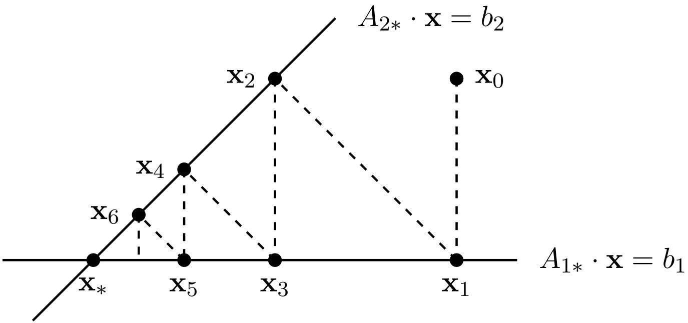


Figure 1: An illustration of Kaczmarz algorithm when $m = 2$.

The main technique: Kaczmarz method

Theorem 1 (Strohmer, Vershynin 2009)

The randomized Kaczmarz algorithm converges to $\mathbf{x}_*$ *in expectation, with the average error*

$$\mathbb{E}[\|\mathbf{x}_T - \mathbf{x}_*\|^2] \leq (1 - \kappa_F^{-2})^T \|\mathbf{x}_0 - \mathbf{x}_*\|^2,$$

where $\kappa_F = \|A\|_F \|A^+\|$.

The main technique: Kaczmarz method

Theorem 1 (Strohmer, Vershynin 2009)

The randomized Kaczmarz algorithm converges to $\mathbf{x}_*$ in expectation, with the average error

$$\mathbb{E}[\|\mathbf{x}_T - \mathbf{x}_*\|^2] \leq (1 - \kappa_F^{-2})^T \|\mathbf{x}_0 - \mathbf{x}_*\|^2,$$

where $\kappa_F = \|A\|_F \|A^+\|$.

By Markov's inequality, $T = O(\kappa_F^2 \log(1/\epsilon^2))$ iterations is enough to obtain $\mathbf{x}_T$ such that

$$\|\mathbf{x}_T - \mathbf{x}_*\|^2 \leq \epsilon^2 \|\mathbf{x}_0 - \mathbf{x}_*\|^2.$$

The main technique: Kaczmarz method

From the scheme, there exist $y_{k,0}, \ldots, y_{k,k-1}$ such that

$$\mathbf{x}_k = \sum_{j=0}^{k-1} y_{k,j} A_{r_j *},$$

Recall
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{b_{r_k} - \langle A_{r_k *} | \mathbf{x}_k \rangle}{\|A_{r_k *}\|^2} A_{r_k *}$$

that is $\mathbf{x}_k = A^\dagger \mathbf{y}_k$ for a k-sparse vector $\mathbf{y}_k$.

The main technique: Kaczmarz method

From the scheme, there exist $y_{k,0}, \ldots, y_{k,k-1}$ such that

$$\mathbf{x}_k = \sum_{j=0}^{k-1} y_{k,j} A_{r_j *},$$

Recall

$$\boxed{\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{b_{r_k} - \langle A_{r_k *} | \mathbf{x}_k \rangle}{\|A_{r_k *}\|^2} A_{r_k *}}$$

that is $\mathbf{x}_k = A^\dagger \mathbf{y}_k$ for a k-sparse vector $\mathbf{y}_k$.

In [Gilyén,Song,Tang '20], $\mathbf{y}_k$ is called the sparse description of $\mathbf{x}_k$.

The main technique: Kaczmarz method

From the scheme, there exist $y_{k,0}, \ldots, y_{k,k-1}$ such that

$$\mathbf{x}_k = \sum_{j=0}^{k-1} y_{k,j} A_{r_j *},$$

Recall
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{b_{r_k} - \langle A_{r_k *} | \mathbf{x}_k \rangle}{\|A_{r_k *}\|^2} A_{r_k *}$$

that is $\mathbf{x}_k = A^\dagger \mathbf{y}_k$ for a k-sparse vector $\mathbf{y}_k$.

In [Gilyén,Song,Tang '20], $\mathbf{y}_k$ is called the sparse description of $\mathbf{x}_k$.

$\Rightarrow$ an iterative scheme for $\mathbf{y}$'s

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \frac{b_{r_k} - \langle A_{r_k *} | A^\dagger | \mathbf{y}_k \rangle}{\|A_{r_k *}\|^2} \mathbf{e}_{r_k},$$

where $\{\mathbf{e}_1, \ldots, \mathbf{e}_m\}$ is the standard basis of $\mathbb{C}^m$.

The main technique: Kaczmarz method

From the scheme, there exist $y_{k,0}, \ldots, y_{k,k-1}$ such that

$$\mathbf{x}_k = \sum_{j=0}^{k-1} y_{k,j} A_{r_j *},$$

Recall
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{b_{r_k} - \langle A_{r_k *} | \mathbf{x}_k \rangle}{\|A_{r_k *}\|^2} A_{r_k *}$$

that is $\mathbf{x}_k = A^\dagger \mathbf{y}_k$ for a k-sparse vector $\mathbf{y}_k$.

In [Gilyén,Song,Tang '20], $\mathbf{y}_k$ is called the sparse description of $\mathbf{x}_k$.

$\Rightarrow$ an iterative scheme for $\mathbf{y}$'s

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \frac{b_{r_k} - \langle A_{r_k *} | A^\dagger | \mathbf{y}_k \rangle}{\|A_{r_k *}\|^2} \mathbf{e}_{r_k},$$

where $\{\mathbf{e}_1, \ldots, \mathbf{e}_m\}$ is the standard basis of $\mathbb{C}^m$.

Indeed, $\mathbf{y}_k$ converges to the optimal solution of the dual problem of the least square problem $\min \|A\mathbf{x} - \mathbf{b}\|$.

The main technique: Kaczmarz method

Consider the least-norm solution of the linear system

$$\min_{\mathbf{x}\in\mathbb{C}^n} \quad \frac{1}{2}\|\mathbf{x}\|^2, \quad \text{s.t.} \quad A\mathbf{x} = \mathbf{b}. \tag{1}$$

The dual problem is

$$\min_{\mathbf{y}\in\mathbb{C}^m} \quad g(\mathbf{y}) := \frac{1}{2}\|A^\dagger \mathbf{y}\|^2 - \mathbf{b}\cdot\mathbf{y}. \tag{2}$$

The main technique: Kaczmarz method

Consider the least-norm solution of the linear system

$$\min_{\mathbf{x}\in\mathbb{C}^n} \quad \frac{1}{2}\|\mathbf{x}\|^2, \quad \text{s.t.} \quad A\mathbf{x} = \mathbf{b}. \tag{1}$$

The dual problem is

$$\min_{\mathbf{y}\in\mathbb{C}^m} \quad g(\mathbf{y}) := \frac{1}{2}\|A^\dagger \mathbf{y}\|^2 - \mathbf{b}\cdot\mathbf{y}. \tag{2}$$

For (1), apply the stochastic gradient descent method to $\frac{1}{2}(A_{r_k*}\cdot\mathbf{x} - b_{r_k})^2$ with stepsize $1/\|A_{r_k*}\|^2 \Rightarrow$ Kaczmarz method.

The main technique: Kaczmarz method

Consider the least-norm solution of the linear system

$$\min_{\mathbf{x}\in\mathbb{C}^n} \quad \frac{1}{2}\|\mathbf{x}\|^2, \quad \text{s.t.} \quad A\mathbf{x} = \mathbf{b}. \tag{1}$$

The dual problem is

$$\min_{\mathbf{y}\in\mathbb{C}^m} \quad g(\mathbf{y}) := \frac{1}{2}\|A^\dagger \mathbf{y}\|^2 - \mathbf{b}\cdot\mathbf{y}. \tag{2}$$

For (1), apply the stochastic gradient descent method to $\frac{1}{2}(A_{r_k*}\cdot\mathbf{x} - b_{r_k})^2$ with stepsize $1/\|A_{r_k*}\|^2 \Rightarrow$ Kaczmarz method.

For (2), apply the randomized coordinate descent method to the r_k-th component, the gradient is $\nabla_{r_k} g = \langle A_{r_k*}|A^\dagger|\mathbf{y}_k\rangle - b_{r_k}$. Choose the stepsize as $1/\|A_{r_k*}\|^2 \Rightarrow$ the iteration for $\mathbf{y}$.

Main idea of the algorithm when A is row sparse

Assume the QRAM data structure [Gilyén,Song,Tang '20]:

1. Output i with probability $\|A_{i*}\|^2/\|A\|_F^2$
2. For each i, output j with probability $\|A_{ij}\|^2/\|A_{i*}\|^2$
3. Query the (i,j)-th entry
4. For each i, output $\|A_{i*}\|^2$
5. Outputs $\|A\|_F^2$

Main idea of the algorithm when A is row sparse

Assume the QRAM data structure [Gilyén,Song,Tang '20]:

1. Output i with probability $\|A_{i*}\|^2/\|A\|_F^2$
2. For each i, output j with probability $\|A_{ij}\|^2/\|A_{i*}\|^2$
3. Query the (i,j)-th entry
4. For each i, output $\|A_{i*}\|^2$
5. Outputs $\|A\|_F^2$

When A is row sparse, just apply the Kaczmarz iteration:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{b_{r_k} - \langle A_{r_k*}|\mathbf{x}_k\rangle}{\|A_{r_k*}\|^2} A_{r_k*}.$$

Assume A has row sparsity s, then it costs $O(s)$ at each step of iteration. So $\widetilde{O}(s\kappa_F^2)$ in total.

Main idea of the algorithm when A is dense

Compute $\mathbf{y}_T$ first through

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \frac{b_{r_k} - \langle A_{r_k *} | A^\dagger | \mathbf{y}_k \rangle}{\| A_{r_k *} \|^2} \mathbf{e}_{r_k}.$$

When A is dense, using the idea of Monte Carlo to estimate $\langle A_{r_k *} | A^\dagger | \mathbf{y}_k \rangle$

Main idea of the algorithm when A is dense

Compute $\mathbf{y}_T$ first through

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \frac{b_{r_k} - \langle A_{r_k *} | A^\dagger | \mathbf{y}_k \rangle}{\|A_{r_k *}\|^2} \mathbf{e}_{r_k}.$$

When A is dense, using the idea of Monte Carlo to estimate $\langle A_{r_k *} | A^\dagger | \mathbf{y}_k \rangle \Rightarrow$ new iterative scheme

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \frac{1}{\|A_{r_k *}\|} \left(\tilde{b}_{r_k} - \frac{1}{|S|} \sum_{j \in S} \tilde{A}_{r_k, j} \langle A_{*j} | \mathbf{y}_k \rangle \frac{\|A\|_F^2}{\|A_{*j}\|^2} \right) \mathbf{e}_{r_k},$$

where S is a collection of column indices such that each j in put into it with probability proportional to the norm square of j-th column.

Recall $\mathbf{x}_k = A^\dagger \mathbf{y}_k$, so we have an iteration scheme for $\mathbf{x}_k$.

Proposition 1

Choose

$$|S| = \frac{4\|A\|_F^2 \kappa_F^2 \log(2/\epsilon^2)}{\epsilon^2 \min_{j\in[n]} \|A_{*j}\|^2},$$

then after $T = O(\kappa_F^2 \log(2/\epsilon^2))$ *steps of iteration, we have*

$$\mathbb{E}[\|\mathbf{x}_T - \mathbf{x}_*\|^2] \leq \epsilon^2 \|\mathbf{x}_*\|^2.$$

Main idea of the algorithm when A is dense

Step 1. Run the new iterative scheme to compute $\mathbf{y}_T$ classically.

Main idea of the algorithm when A is dense

Step 1. Run the new iterative scheme to compute $\mathbf{y}_T$ classically.

$\mathbf{y}_T$ is at most $\widetilde{O}(\kappa_F^2)$-sparse.

Main idea of the algorithm when A is dense

Step 1. Run the new iterative scheme to compute $\mathbf{y}_T$ classically.

$\mathbf{y}_T$ is at most $\widetilde{O}(\kappa_F^2)$-sparse.

Step 2. Build the data structure for $\mathbf{x}_T = A^\dagger \mathbf{y}_T$ (based on some techniques from quantum-inspired algorithms [Chia, Gilyén, Li, Lin, Tang, 2020])

Main idea of the algorithm when A is dense

Step 1. Run the new iterative scheme to compute $\mathbf{y}_T$ classically.

$\mathbf{y}_T$ is at most $\widetilde{O}(\kappa_F^2)$-sparse.

Step 2. Build the data structure for $\mathbf{x}_T = A^\dagger \mathbf{y}_T$ (based on some techniques from quantum-inspired algorithms [Chia, Gilyén, Li, Lin, Tang, 2020])

Step 2 is the most technical part of the algorithm; however, the main cost comes from step 1.

More about Kaczmarz method

There are many generalizations about Kaczmarz method [Gower, Richtárik 2015; Necoara 2019]

More about Kaczmarz method

There are many generalizations about Kaczmarz method [Gower, Richtárik 2015; Necoara 2019]

1. Block projection Kaczmarz algorithm

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k A_{Q_k}^{+}(\mathbf{b}_{Q_k} - A_{Q_k}\mathbf{x}_k),$$

   where A_{Q_k} is the matrix of the rows with index in Q_k.

More about Kaczmarz method

There are many generalizations about Kaczmarz method [Gower, Richtárik 2015; Necoara 2019]

1. Block projection Kaczmarz algorithm

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k A_{Q_k}^{+}(\mathbf{b}_{Q_k} - A_{Q_k}\mathbf{x}_k),$$

   where A_{Q_k} is the matrix of the rows with index in Q_k.

2. Average block Kaczmarz algorithm

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \sum_{i \in Q_k} w_i (b_i - \frac{A_i^T \mathbf{x}_k}{\|A_i\|^2}) A_i,$$

   where the weights $w_i \in [0,1]$ satisfy $\sum_i w_i = 1$ and $\alpha_k \in (0,2)$. With suitable choices of α_k, w_i, the rate of convergence can be $O(\|A\|^2 \|A^{-1}\|^2 \log(1/\epsilon^2))$.

A new way to look at Kaczmarz method

Assume that A is nonsingular and $\mathbf{x}_* = A^{-1}\mathbf{b}$. Recall the original Kaczmarz method

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha\left(b_{i_k} - \frac{A_{i_k}^T \mathbf{x}_k}{\|A_{i_k}\|^2}\right)A_{i_k} = \mathbf{x}_k - \alpha\frac{A_{i_k}A_{i_k}^T}{\|A_{i_k}\|^2}(\mathbf{x}_k - \mathbf{x}_*).$$

A new way to look at Kaczmarz method

Assume that A is nonsingular and $\mathbf{x}_* = A^{-1}\mathbf{b}$. Recall the original Kaczmarz method

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha(b_{i_k} - \frac{A_{i_k}^T \mathbf{x}_k}{\|A_{i_k}\|^2})A_{i_k} = \mathbf{x}_k - \alpha \frac{A_{i_k} A_{i_k}^T}{\|A_{i_k}\|^2}(\mathbf{x}_k - \mathbf{x}_*).$$

Now choose $\alpha = 2$,

$$\mathbf{x}_{k+1} - \mathbf{x}_* = (I - 2\frac{A_{i_k} A_{i_k}^T}{\|A_{i_k}\|^2})(\mathbf{x}_k - \mathbf{x}_*).$$

Apparently, all $\mathbf{x}_k$'s lie on the sphere

$$\mathbb{S} = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x} - \mathbf{x}_*\| = \|\mathbf{x}_0 - \mathbf{x}_*\|\}.$$

A new way to look at Kaczmarz method

Assume that A is nonsingular and $\mathbf{x}_* = A^{-1}\mathbf{b}$. Recall the original Kaczmarz method

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha(b_{i_k} - \frac{A_{i_k}^T \mathbf{x}_k}{\|A_{i_k}\|^2})A_{i_k} = \mathbf{x}_k - \alpha \frac{A_{i_k} A_{i_k}^T}{\|A_{i_k}\|^2}(\mathbf{x}_k - \mathbf{x}_*).$$

Now choose $\alpha = 2$,

$$\mathbf{x}_{k+1} - \mathbf{x}_* = (I - 2\frac{A_{i_k} A_{i_k}^T}{\|A_{i_k}\|^2})(\mathbf{x}_k - \mathbf{x}_*).$$

Apparently, all $\mathbf{x}_k$'s lie on the sphere

$$\mathbb{S} = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x} - \mathbf{x}_*\| = \|\mathbf{x}_0 - \mathbf{x}_*\|\}.$$

Question: determine m and $\mathbf{x}_{k_1}, \ldots, \mathbf{x}_{k_m} \in \mathbb{S}$ such that

$$\|\frac{1}{m}\sum_{j=1}^{m} \mathbf{x}_{k_j} - \mathbf{x}_*\| \leq \epsilon \|\mathbf{x}_0 - \mathbf{x}_*\|.$$

Two approaches

1. i_k is chosen in random [Steinerberger, 2020]

$$\mathbb{E}\|\frac{1}{m}\sum_{j=1}^{m}\mathbf{x}_j - \mathbf{x}_*\| \leq \epsilon\|\mathbf{x}_0 - \mathbf{x}_*\|, \quad m = O(\|A\|_F^2\|A^{-1}\|^2/\epsilon^2)$$

2. i_k is chosen in order [our approach]

Two approaches

1. i_k is chosen in random [Steinerberger, 2020]

$$\mathbb{E}\|\frac{1}{m}\sum_{j=1}^{m}\mathbf{x}_j - \mathbf{x}_*\| \le \epsilon\|\mathbf{x}_0 - \mathbf{x}_*\|, \quad m = O(\|A\|_F^2\|A^{-1}\|^2/\epsilon^2)$$

2. i_k is chosen in order [our approach]

(a) i_k is chosen in random (b) i_k is chosen in order

Figure 2: An illustration of the distribution of $\{\mathbf{x}_1, \ldots, \mathbf{x}_{1000}\}$ on $\mathbb{S}$ when $n = 3$.

More details

For simplicity, denote

$$R_i = I - 2\frac{A_i A_i^T}{\|A_i\|^2}, \quad R = R_n \cdots R_2 R_1.$$

If $i_k = (k \mod n) + 1$, then

$$\mathbf{x}_{k+1} - \mathbf{x}_* = R_{i_k}(\mathbf{x}_k - \mathbf{x}_*) = R_{k+1} \cdots R_1(\mathbf{x}_0 - \mathbf{x}_*).$$

More details

For simplicity, denote

$$R_i = I - 2\frac{A_i A_i^T}{\|A_i\|^2}, \quad R = R_n \cdots R_2 R_1.$$

If $i_k = (k \mod n) + 1$, then

$$\mathbf{x}_{k+1} - \mathbf{x}_* = R_{i_k}(\mathbf{x}_k - \mathbf{x}_*) = R_{k+1} \cdots R_1(\mathbf{x}_0 - \mathbf{x}_*).$$

Still consider the case $n = 3$, then we obtain a series of vectors

$$\mathbf{x}_0 \qquad \mathbf{x}_* + R_1(\mathbf{x}_0 - \mathbf{x}_*) \qquad \mathbf{x}_* + R_2 R_1(\mathbf{x}_0 - \mathbf{x}_*)$$

More details

For simplicity, denote

$$R_i = I - 2\frac{A_i A_i^T}{\|A_i\|^2}, \quad R = R_n \cdots R_2 R_1.$$

If $i_k = (k \mod n) + 1$, then

$$\mathbf{x}_{k+1} - \mathbf{x}_* = R_{i_k}(\mathbf{x}_k - \mathbf{x}_*) = R_{k+1} \cdots R_1(\mathbf{x}_0 - \mathbf{x}_*).$$

Still consider the case $n = 3$, then we obtain a series of vectors

$$\begin{array}{lll} \mathbf{x}_0 & \mathbf{x}_* + R_1(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2 R_1(\mathbf{x}_0 - \mathbf{x}_*) \\ \mathbf{x}_* + R(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_1 R(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2 R_1 R(\mathbf{x}_0 - \mathbf{x}_*) \end{array}$$

More details

For simplicity, denote

$$R_i = I - 2\frac{A_i A_i^T}{\|A_i\|^2}, \quad R = R_n \cdots R_2 R_1.$$

If $i_k = (k \mod n) + 1$, then

$$\mathbf{x}_{k+1} - \mathbf{x}_* = R_{i_k}(\mathbf{x}_k - \mathbf{x}_*) = R_{k+1} \cdots R_1(\mathbf{x}_0 - \mathbf{x}_*).$$

Still consider the case $n = 3$, then we obtain a series of vectors

$$\begin{array}{lll}
\mathbf{x}_0 & \mathbf{x}_* + R_1(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2 R_1(\mathbf{x}_0 - \mathbf{x}_*) \\
\mathbf{x}_* + R(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_1 R(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2 R_1 R(\mathbf{x}_0 - \mathbf{x}_*) \\
\mathbf{x}_* + R^2(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_1 R^2(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2 R_1 R^2(\mathbf{x}_0 - \mathbf{x}_*)
\end{array}$$

More details

For simplicity, denote

$$R_i = I - 2\frac{A_i A_i^T}{\|A_i\|^2}, \quad R = R_n \cdots R_2 R_1.$$

If $i_k = (k \mod n) + 1$, then

$$\mathbf{x}_{k+1} - \mathbf{x}_* = R_{i_k}(\mathbf{x}_k - \mathbf{x}_*) = R_{k+1} \cdots R_1(\mathbf{x}_0 - \mathbf{x}_*).$$

Still consider the case $n = 3$, then we obtain a series of vectors

$$\begin{array}{lll}
\mathbf{x}_0 & \mathbf{x}_* + R_1(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2R_1(\mathbf{x}_0 - \mathbf{x}_*) \\
\mathbf{x}_* + R(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_1R(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2R_1R(\mathbf{x}_0 - \mathbf{x}_*) \\
\mathbf{x}_* + R^2(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_1R^2(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2R_1R^2(\mathbf{x}_0 - \mathbf{x}_*) \\
\mathbf{x}_* + R^3(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_1R^3(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2R_1R^3(\mathbf{x}_0 - \mathbf{x}_*)
\end{array}$$

More details

For simplicity, denote

$$R_i = I - 2\frac{A_i A_i^T}{\|A_i\|^2}, \quad R = R_n \cdots R_2 R_1.$$

If $i_k = (k \mod n) + 1$, then

$$\mathbf{x}_{k+1} - \mathbf{x}_* = R_{i_k}(\mathbf{x}_k - \mathbf{x}_*) = R_{k+1} \cdots R_1(\mathbf{x}_0 - \mathbf{x}_*).$$

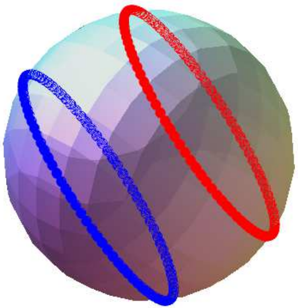
Still consider the case $n = 3$, then we obtain a series of vectors

$$\begin{array}{lll}
\mathbf{x}_0 & \mathbf{x}_* + R_1(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2R_1(\mathbf{x}_0 - \mathbf{x}_*) \\
\mathbf{x}_* + R(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_1R(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2R_1R(\mathbf{x}_0 - \mathbf{x}_*) \\
\mathbf{x}_* + R^2(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_1R^2(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2R_1R^2(\mathbf{x}_0 - \mathbf{x}_*) \\
\mathbf{x}_* + R^3(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_1R^3(\mathbf{x}_0 - \mathbf{x}_*) & \mathbf{x}_* + R_2R_1R^3(\mathbf{x}_0 - \mathbf{x}_*) \\
\cdots & \cdots & \cdots
\end{array}$$

More details

Generally, define

$$\begin{aligned}
\mathbb{S}_0 &:= \{\mathbf{y}_i := \mathbf{x}_* + R^{2i}(\mathbf{x}_0 - \mathbf{x}_*) : i = 0, 1, 2, \ldots\}, \\
\mathbb{S}_1 &:= \{\mathbf{z}_i := \mathbf{x}_* + R^{2i+1}(\mathbf{x}_0 - \mathbf{x}_*) : i = 0, 1, 2, \ldots\}.
\end{aligned}$$



Main results

Theorem 2

For $j = 0, 1$, *let* $\mathbf{c}_j$ *be the center of the minimal sphere supporting* $\mathbb{S}_j$, *then* $\mathbf{c}_0 + \mathbf{c}_1 = 2\mathbf{x}_*$.

Main results

Theorem 2

For $j = 0, 1$, let $\mathbf{c}_j$ be the center of the minimal sphere supporting $\mathbb{S}_j$, then $\mathbf{c}_0 + \mathbf{c}_1 = 2\mathbf{x}_*$.

Theorem 3

Let $\{e^{i\theta_l} : 1 \leq l \leq n, |\theta_l| \leq \pi\}$ be the eigenvalues of R, then $\min_l |\theta_l| > 0$ and

$$\|\frac{1}{2m}\sum_{i=0}^{m-1}(\mathbf{y}_i + \mathbf{z}_i) - \mathbf{x}_*\| \leq \epsilon\|\mathbf{x}_*\|,$$

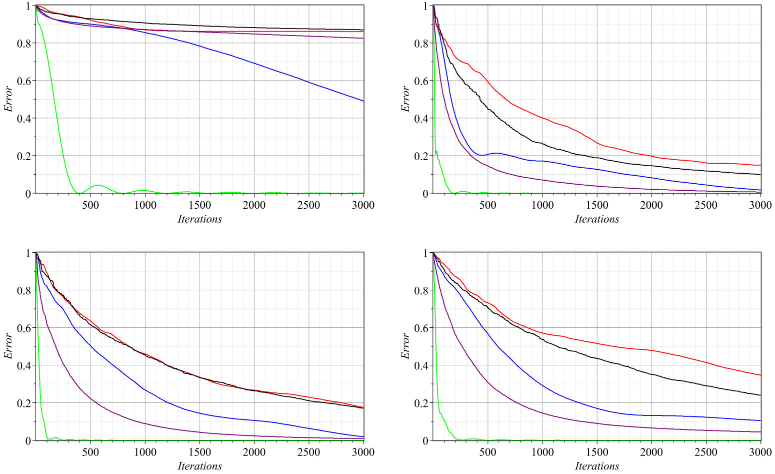
where

$$m = O\left(\frac{1}{\epsilon^2} + \frac{1}{\epsilon \min_l |\theta_l|}\right).$$

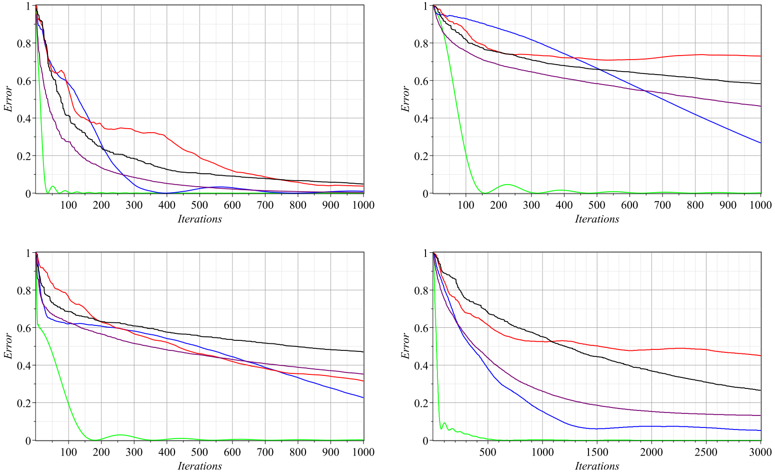
Comparison

Algorithm	Total operations	Type
Kaczmarz	$O(n\|A\|_F^2\|A^{-1}\|^2\log(1/\epsilon^2))$	Randomized
Steinerberger	$O(n\|A\|_F^2\|A^{-1}\|^2/\epsilon^2)$	Randomized
Our	$O(n^2/\epsilon^2 + n^2/\epsilon\min_l\|\theta_l\|)$	Deterministic

Experiments



Experiments



Conclusions and outlooks

Conclusions:

- We reduced the gap between quantum and quantum-inspired classical algorithm for linear equations from $\kappa_F : \kappa_F^6\kappa^6/\epsilon^4$ to $\kappa_F : \kappa_F^6\kappa^2/\epsilon^2$.
- In the row sparse case, the quantum speedup is quadratic if assuming access to QRAM.
- We proposed a deterministic Kaczmarz method with fast rate of convergence in practice.

Conclusions and outlooks

Conclusions:

- We reduced the gap between quantum and quantum-inspired classical algorithm for linear equations from $\kappa_F : \kappa_F^6 \kappa^6/\epsilon^4$ to $\kappa_F : \kappa_F^6 \kappa^2/\epsilon^2$.
- In the row sparse case, the quantum speedup is quadratic if assuming access to QRAM.
- We proposed a deterministic Kaczmarz method with fast rate of convergence in practice.

Outlooks:

- Reduce the dependence of quantum-inspired classical algorithm on $\|A\|_F$.

Conclusions and outlooks

Conclusions:

- We reduced the gap between quantum and quantum-inspired classical algorithm for linear equations from $\kappa_F : \kappa_F^6 \kappa^6/\epsilon^4$ to $\kappa_F : \kappa_F^6 \kappa^2/\epsilon^2$.
- In the row sparse case, the quantum speedup is quadratic if assuming access to QRAM.
- We proposed a deterministic Kaczmarz method with fast rate of convergence in practice.

Outlooks:

- Reduce the dependence of quantum-inspired classical algorithm on $\|A\|_F$.

Thank you!