QUANTUM SPEEDUPS FOR LPs VIA IPMs

Simon Apers
(CNRS & IRIF, Paris)

with Sander Gribling (Tilburg University)

arXiv:2311.03215

ANR QuDATA workshop, Bordeaux, January ’24
LPs and IPMs
Approximate Hessian
Approximate gradient
Quantum LP solver
LPs and IPMs

Approximate Hessian

Approximate gradient

Quantum LP solver
Linear program (LP)

\[
\begin{align*}
\text{min} & \quad c^T x \\
\text{s.t.} & \quad b_i \geq 0 \\
& \quad a_i^T x \leq b_i
\end{align*}
\]

dimensions \ll n

(constrained) convex optimization
Linear program (LP)

\[ \begin{align*}
\min & \quad c^T x \\
\text{s.t.} & \quad Ax \geq b
\end{align*} \]
Linear program (LP)

\[ \min \quad c^T x \]

\[ \text{s.t.} \quad Ax \geq b \]

\( d \) dimensions \( \ll \) \( n \) constraints \((x \in \mathbb{R}^d, A \in \mathbb{R}^{n \times d})\)
Linear program (LP)

\[ \begin{align*}
\text{min} & \quad c^T x \\
\text{s.t.} & \quad Ax \geq b
\end{align*} \]

\( d \) dimensions \( \ll n \) constraints \((x \in \mathbb{R}^d, A \in \mathbb{R}^{n \times d})\)

= (constrained) convex optimization
Interior point method (IPM)

barrier $f$:

$$f(x) \to \infty \text{ when } a_i^T x \to b_i$$
Interior point method (IPM)

barrier $f$:

$$f(x) \to \infty \text{ when } a_i^T x \to b_i$$
Interior point method (IPM)

barrier $f$:
\[ f(x) \to \infty \text{ when } a_i^T x \to b_i \]

e.g., logarithmic barrier: $f(x) = -\sum_i \log(a_i^T x - b_i)$
Interior point method (IPM)

barrier $f$:

$$f(x) \to \infty \text{ when } a_i^T x \to b_i$$

e.g., logarithmic barrier: $f(x) = - \sum_i \log(a_i^T x - b_i)$

$$\min_x f(x) + c^T x$$
Interior point method (IPM)

barrier $f$:

$$f(x) \to \infty \text{ when } a_i^T x \to b_i$$

e.g., logarithmic barrier: $f(x) = - \sum_i \log(a_i^T x - b_i)$

$$\min_x f(x) + c^T x$$

$= \text{unconstrained} \text{ convex optimization}$
Interior point method (IPM)

The central path is defined by:

$$\eta(x) = f(x) + \eta \cdot c^T x$$
Interior point method (IPM)

$$f_\eta(x) = f(x) + \eta \cdot c^T x$$

central path \( \{ z_\eta = \arg\min_x f_\eta(x) \} \eta \geq 0 \)
Path following

1. Increase $\eta$:
   $$\eta' = (1 + \gamma) \eta$$

2. Newton step:
   $$x' = x - H(x)^{-1} g(x)$$
   (Hessian $H(x) = \nabla^2 f_\eta(x)$, gradient $g(x) = \nabla f_\eta(x)$)
Path following

1. increase $\eta$:

$$\eta' = (1 + \gamma) \eta$$
Path following

1. increase $\eta$:
   
   $$\eta' = (1 + \gamma) \eta$$

2. Newton step:
   
   $$x' = x - H(x)^{-1} g(x)$$

   (Hessian $H(x) = \nabla^2 f_\eta(x)$, gradient $g(x) = \nabla f_\eta(x)$)
Path following

e.g., logarithmic barrier:

\[ f(x) = - \sum_i \log(a_i^T x - b_i) \]

slack \( s_i \)
Path following

e.g., logarithmic barrier:

\[ f(x) = -\sum_i \log(a_i^T x - b_i) \]

Hessian

\[ H(x) = \sum_i \frac{1}{s_i^2} a_i a_i^T = B^T B \]

\[ (B^T)_{i} = \frac{1}{s_i} a_i \]
e.g., logarithmic barrier:

\[ f(x) = - \sum_i \log(a_i^T x - b_i) \]

Hessian

\[ H(x) = \sum_i \frac{1}{s_i^2} a_i a_i^T = B^T B \]

\[ (B^T)_i = \frac{1}{s_i} a_i \]

gradient

\[ g(x) = -B^T \mathbf{1} \]
Runtime IPM

number of steps

\sim \text{number of increases } \eta
number of steps

\[ \sim \text{number of increases } \eta \]

- logarithmic: \( \sqrt{n} \) [Renegar ’88]
- volumetric: \( \sqrt{nd} \) [Vaidya ’89]
- Lewis weight: \( \sqrt{d} \) [Lee-Sidford ’15]
Runtime IPM

number of steps

\[ \sim \text{number of increases } \eta \]

logarithmic: \( \sqrt{n} \) [Renegar ’88]

volumetric: \( \sqrt{nd} \) [Vaidya ’89]

Lewis weight: \( \sqrt{d} \) [Lee-Sidford ’15]

single step

= computation (inverse) Hessian and gradient of barrier
Runtime IPM

number of steps

\[ \sim \text{number of increases } \eta \]

- logarithmic: \( \sqrt{n} \) \[\text{[Renegar '88]}\]
- volumetric: \( \sqrt{nd} \) \[\text{[Vaidya '89]}\]
- Lewis weight: \( \sqrt{d} \) \[\text{[Lee-Sidford '15]}\]

single step

= computation (inverse) Hessian and gradient of barrier

- logarithmic: matrix inversion
- volumetric: matrix inversion + leverage scores
- Lewis weight: matrix inversion + Lewis weights
SOTA (classical)

Solving Tall Dense Linear Programs in Nearly Linear Time

Jan van den Brand
KTH Royal Institute of Technology, Sweden
janvdb@kth.se

Aaron Sidford
Stanford University, U.S.A.
sidford@stanford.edu

Yin Tat Lee
University of Washington and MSR Redmond, U.S.A.
yintat@uw.edu

Zhao Song
Princeton University and
Institute for Advanced Study, U.S.A.
zhaos@ias.edu
runtime $nd + d^3$

(GOAT for $n \gg d$: linear in input size)
SOTA (classical)

Solving Tall Dense Linear Programs in Nearly Linear Time

Jan van den Brand
KTH Royal Institute of Technology, Sweden
janvdb@kth.se

Aaron Sidford
Stanford University, U.S.A.
sidford@stanford.edu

Yin Tat Lee
University of Washington and MSR Redmond, U.S.A.
yintat@uw.edu

Zhao Song
Princeton University and Institute for Advanced Study, U.S.A.
zhaos@ias.edu

runtime $nd + d^3$
(GOAT for $n \gg d$: linear in input size)

IPM + clever use of dynamic data structures
Prior work (quantum)

quantum speedup for Newton step

\[ x' = x - H(x)^{-1}g(x) \]
Prior work (quantum)

quantum speedup for Newton step

$$x' = x - H(x)^{-1}g(x)$$

quantum linear system solving + tomography

*A Quantum Interior Point Method for LPs and SDPs*

Iordanis Kerenidis and Anupam Prakash, CNRS, IRIF, Université Paris Diderot

* and follow-up works: [Augustino-Nannicini-Terlaky-Zuluaga '21,'23], [Huang-Jiang-Song-Tao-Zhang '22], [Dalzell-Clader-Salton-Berta-Lin-Bader-Stamatopoulos-Schuetz-Brandão-Katzgraber-Zeng '22], . . .

** non-IPM: multiplicative weights [Brandão-Svore '17], [van Apeldoorn-Gilyén-Gribling-de Wolf-Brandão-Kalev-Li-Lin-Svore-Wu '17], simplex method [Nannicini '19], . . .
Prior work (quantum)

quantum speedup for \textbf{Newton step}
\[ x' = x - H(x)^{-1}g(x) \]

quantum linear system solving + tomography

\[ \text{dependence on condition number } \kappa(H) \]
\[ (\rightarrow \infty \text{ as } x \rightarrow \text{OPT}) \]

* and follow-up works: [Augustino-Nannicini-Terlaky-Zuluaga ’21,’23], [Huang-Jiang-Song-Tao-Zhang ’22], [Dalzell-Clader-Salton-Berta-Lin-Bader-Stamatopoulos-Schuetz-Brandão-Katzgraber-Zeng ’22], . . .

** non-IPM: multiplicative weights [Brandão-Svore ’17], [van Apeldoorn-Gilyén-Gribling-de Wolf-Brandão-Kalev-Li-Lin-Svore-Wu ’17], simplex method [Nannicini ’19], . . .
quantum speedup for **Newton step**

\[ x' = x - H(x)^{-1} g(x) \]
This work

quantum speedup for **Newton step**

\[ x' = x - H(x)^{-1} g(x) \]

(i) approximate Hessian \( H' \approx H \)

(ii) approximate gradient \( g' \approx g \)

(using \( H' \) as preconditioner)
This work

quantum speedup for **Newton step**

\[ x' = x - H(x)^{-1} g(x) \]

(i) approximate Hessian \( H' \approx H \)

(ii) approximate gradient \( g' \approx g \)

(uses \( H' \) as preconditioner)

runtime \( \sqrt{n} \cdot \text{poly}(d) \cdot \log(1/\varepsilon) \)

(no condition number, sublinear for \( n \gg d \))
This work

quantum speedup for **Newton step**

\[ x' = x - H(x)^{-1} g(x) \]

(i) approximate Hessian \( H' \approx H \)
(ii) approximate gradient \( g' \approx g \) (using \( H' \) as preconditioner)

runtime \( \sqrt{n} \cdot \text{poly}(d) \cdot \log(1/\varepsilon) \)
(no condition number, sublinear for \( n \gg d \))

**GOAT:** \( \Omega(\sqrt{nd}) \) row queries
LPs and IPMs

Approximate Hessian

Approximate gradient

Quantum LP solver
Spectral approximation
Spectral approximation via constraint sampling

\[ A = n \quad \tilde{\delta}(d) = \tilde{A} \]
Spectral approximation

approximate Newton step

\[ x' = x - \tilde{H}^{-1}g \]

eeds spectral approximation
Spectral approximation

approximate Newton step

\[ x' = x - \tilde{H}^{-1}g \]

needs spectral approximation

\[ * \quad H \approx \tilde{H} \quad \iff \forall y: \quad y^T H y = (1 \pm 0.1) y^T \tilde{H} y \quad \iff \quad 0.9 \tilde{H} \preceq H \preceq 1.1 \tilde{H} \]
Spectral approximation

sampling via “statistical leverage scores”

\[ \sigma_i = a_i^T (A^T A)^{-1} a_i \]
Spectral approximation

\[ A = \begin{pmatrix} a_1 & \cdots & a_n \end{pmatrix} \]

\[ \tilde{\sigma}(d) = \tilde{A} \]

Sampling via “statistical leverage scores”

\[ \sigma_i = a_i^T (A^T A)^{-1} a_i \]

Sarah-and-egg
Spectral approximation

\[
\sigma_i = a_i^T (A^T A)^{-1} a_i
\]

sampling via “statistical leverage scores”

! chicken-and-egg

use

*uniform* subsampling + bootstrapping scheme

[Cohen-Lee-Musco-Musco-Peng-Sidford ‘14]
Spectral approximation

\[ A = \begin{pmatrix} a_1 & \cdots & a_n \end{pmatrix}, \quad \tilde{\sigma}(u) = \tilde{A} \]

Sampling via “statistical leverage scores”

\[ \sigma_i = a_i^T (A^T A)^{-1} a_i \]

! chicken-and-egg

Use

*uniform* subsampling + bootstrapping scheme

[Cohen-Lee-Musco-Musco-Peng-Sidford ‘14]

+ Grover search
Spectral approximation

quantum algorithm:
– returns $\tilde{A}$ with $\widetilde{O}(d)$ rows,
  $\tilde{A}$ spectral approximation of $A$
– makes $\widetilde{O}(\sqrt{nd})$ row queries
Spectral approximation

quantum algorithm:
- returns $\tilde{A}$ with $\tilde{O}(d)$ rows,
  $\tilde{A}$ spectral approximation of $A$
- makes $\tilde{O}(\sqrt{nd})$ row queries

- generalizes graph sparsification

[Apers-de Wolf '19]
Spectral approximation

quantum algorithm:
- returns $\tilde{A}$ with $\tilde{O}(d)$ rows,
- $\tilde{A}$ spectral approximation of $A$
- makes $\tilde{O}(\sqrt{nd})$ row queries

- generalizes graph sparsification

- applications in regression

[Apers-de Wolf '19]

Revisiting Quantum Algorithms for Linear Regressions: Quadratic Speedups without Data-Dependent Parameters
Zhao Song, Junze Yin, Ruizhe Zhang
Spectral approximation

quantum algorithm:
- returns $\tilde{A}$ with $\tilde{O}(d)$ rows,
  $\tilde{A}$ spectral approximation of $A$
- makes $\tilde{O}(\sqrt{nd})$ row queries

- generalizes graph sparsification

[Apers-de Wolf '19]

- applications in regression

[Submitted on 24 Nov 2023]
Revisiting Quantum Algorithms for Linear Regressions: Quadratic Speedups without Data-Dependent Parameters
Zhao Song, Junze Yin, Ruizhe Zhang

- IPMs: need additional work
  (e.g., Lee-Sidford barrier uses “Lewis weights”)
LPs and IPMs

Approximate Hessian

Approximate gradient

Quantum LP solver
Approximate gradient

typical gradient:

\[ g = \left[ a_i \right] A^T \begin{bmatrix} \frac{1}{n} \vdots \frac{1}{n} \end{bmatrix} = n \cdot \mathbb{E}_i \left[ a_i \right] \]
Approximate gradient

typical gradient:

\[
g = \begin{bmatrix} a_i \\ A^T \end{bmatrix} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = n \cdot \mathbb{E}_i \left[ a_i \right]
\]

→ quantum multivariate mean estimation:

[Cornelissen-Hamoudi-Jerbi '22]
Approximate gradient

typical gradient:

\[ g = \left[ \begin{array}{c} a_i \\ \vdots \\ 1 \end{array} \right] A^T \left[ \begin{array}{c} a_i \\ \vdots \\ 1 \end{array} \right] = n \cdot \mathbb{E} \left[ \begin{array}{c} a_i \\ \vdots \\ 1 \end{array} \right] \]

→ quantum multivariate mean estimation:
[Cornelissen-Hamoudi-Jerbi '22]

approximate \( g = \mathbb{E}[X] \) with sample complexity

\[ \sqrt{d \operatorname{Tr}(\Sigma_X)} \]
Preconditioning

gradient $g = \mathbb{E}[X] = \mathbb{E}_i[n a_i]$

covariance matrix $\Sigma_X \preceq \mathbb{E}[XX^T] = n A^T A \Rightarrow \sqrt{dn \text{Tr}(A^T A)}$ samples
Preconditioning

\[
\text{gradient } g = \mathbb{E}[X] = \mathbb{E}_i[na_i] \\
\text{covariance matrix } \Sigma_X \preceq \mathbb{E}[XX^T] = nA^TA \quad \Rightarrow \quad \sqrt{dn\text{Tr}(A^TA)} \text{ samples}
\]

! \text{Tr}(A^TA) \text{ introduces condition number :}(
Preconditioning

gradient \( g = \mathbb{E}[X] = \mathbb{E}_i[n a_i] \)
covariance matrix \( \Sigma_X \preceq \mathbb{E}[XX^T] = nA^T A \)
\[
\{ \Rightarrow \sqrt{dn \text{Tr}(A^T A)} \text{ samples} \}
\]

! Tr\((A^T A)\) introduces condition number :( 

\[
\downarrow
\]

precondition with spectral approximation:
\[
Y = (\tilde{A}^T \tilde{A})^{-1/2} X
\]
gradient $g = \mathbb{E}[X] = \mathbb{E}_i[n a_i]$

covariance matrix $\Sigma_X \preceq \mathbb{E}[XX^T] = nA^TA$

$\Rightarrow \sqrt{d n \text{Tr}(A^TA)}$ samples

! $\text{Tr}(A^TA)$ introduces condition number :

\[ \downarrow \]

precondition with spectral approximation:

\[ Y = (\tilde{A}^T\tilde{A})^{-1/2}X \]

s.t.

\[ g = (\tilde{A}^T\tilde{A})^{1/2} \cdot \mathbb{E}[Y] \quad \text{and} \quad \Sigma_Y \preceq 1.1 n I_d \]
Preconditioning

gradient $g = \mathbb{E}[X] = \mathbb{E}_i[n a_i]$

covariance matrix $\Sigma_X \preceq \mathbb{E}[XX^T] = n A^T A \quad \Rightarrow \quad \sqrt{d n \text{Tr}(A^T A)}$ samples

\[ ! \quad \text{Tr}(A^T A) \text{ introduces condition number :} ( \]

\[ \downarrow \]

**precondition** with spectral approximation:

$$Y = (\tilde{A}^T \tilde{A})^{-1/2} X$$

\[ \text{s.t.} \]

$$g = (\tilde{A}^T \tilde{A})^{1/2} \cdot \mathbb{E}[Y] \quad \text{and} \quad \Sigma_Y \preceq 1.1 n I_d \quad \Rightarrow \quad O(d \sqrt{n}) \text{ samples}$$
LPs and IPMs

Approximate Hessian

Approximate gradient

Quantum LP solver
Quantum LP solver

explicitly returns $\tilde{x}$ satisfying

$$c^T \tilde{x} \leq c^T \text{OPT} + \varepsilon$$

and

$$A \tilde{x} \leq b$$
Quantum LP solver

explicitly returns \( \tilde{x} \) satisfying

\[
c^T \tilde{x} \leq c^T \text{OPT} + \varepsilon
\]

and \( A\tilde{x} \leq b \)

\[
\begin{align*}
\# \text{ row queries}^* &:= (\# \text{ steps}) \times (\# \text{ cost Newton step}) \\
&= \sqrt{d} \log(1/\varepsilon) \times \sqrt{nd^{2.5}} \in \tilde{O}(\sqrt{nd^3})
\end{align*}
\]

time complexity: \( \sqrt{n} \log(1/\varepsilon) \text{poly}(d) \)

* using Lewis weight barrier.

log-barrier: \( \sqrt{n} \log(1/\varepsilon) \times \sqrt{nd} \), volumetric barrier: \( (nd)^{1/4} \log(1/\varepsilon) \times \sqrt{nd^2} \)
Summary and open questions

- quantum IPM for solving LPs (without condition numbers)
- runtime \( (n \gg d) \sqrt{n} \cdot \text{poly}(d) \cdot \log(\frac{1}{\epsilon}) \) versus \( n \cdot d \cdot \log(\frac{1}{\epsilon}) \)

- new tools
  - spectral approximation (Grover)
  - approximate matrix-vector (mean estimation)

- main open question: be the GOAT; match \( \Omega(\sqrt{nd}) \) row queries LB

Thanks!
Summary and open questions

- quantum IPM for solving LPs
  (without condition numbers)

\[ a^T x \leq b_i \]

\[ A = \begin{bmatrix} \tilde{A}^T \end{bmatrix} = n \cdot \tilde{\Phi} \left( \tilde{A}^T \right) \]
Summary and open questions

- **quantum IPM for solving LPs**
  (without condition numbers)

- **runtime** \((n \gg d)\)

\[
\sqrt{n} \cdot \text{poly}(d) \cdot \log(1/\varepsilon)
\]

versus \(n \cdot d \cdot \log(1/\varepsilon)\) classical
Summary and open questions

- quantum IPM for solving LPs (without condition numbers)
- runtime \((n \gg d)\)
  \[
  \sqrt{n} \cdot \text{poly}(d) \cdot \log(1/\varepsilon)
  \]
  versus \(n \cdot d \cdot \log(1/\varepsilon)\) classical

- new tools
  spectral approximation (Grover)
  approximate matrix-vector (mean estimation)
Summary and open questions

- **quantum IPM for solving LPs**
  (without condition numbers)

- **runtime** \((n \gg d)\)
  \[
  \sqrt{n} \cdot \text{poly}(d) \cdot \log(1/\varepsilon)
  \]

versus \(n \cdot d \cdot \log(1/\varepsilon)\) classical

- **new tools**
  spectral approximation (Grover)
  approximate matrix-vector (mean estimation)

- **main open question**
  be the GOAT: match \(\Omega(\sqrt{nd})\) row queries LB
Summary and open questions

- **quantum IPM for solving LPs**
  (without condition numbers)

- **runtime** \((n \gg d)\)
  \[
  \sqrt{n} \cdot \text{poly}(d) \cdot \log(1/\varepsilon)
  \]
  versus \(n \cdot d \cdot \log(1/\varepsilon)\) classical

- **new tools**
  spectral approximation (Grover)
  approximate matrix-vector (mean estimation)

- **main open question**
  be the GOAT: match \(\Omega(\sqrt{nd})\) row queries LB

- **thanks!**
Extra slide: cutting plane & lower bound

- separation query:
given $x$, return violated constraint (if any)

- fastest cutting plane method [Lee-Sidford-Wong ’15]:
solve LP with $d$ separation queries

  - using Grover:
    answer separation query with $\sqrt{n}$ row queries
    solve LP with $d \cdot \sqrt{n}$ row queries

- quantum lower bound:
  $\sqrt{d \cdot n}$ row queries