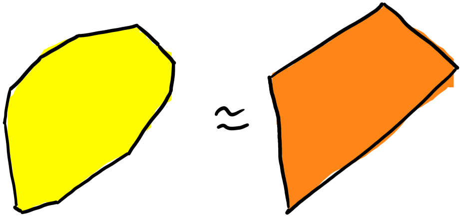
Quantum speedups for LPs via IPMs



Simon Apers
(CNRS & IRIF, Paris)

with **Sander Gribling** (Tilburg University)

arXiv:2311.03215

QIP, Taipei (Taiwan), January ’24

LPs and IPMs

Approximate Hessian

Approximate gradient

Quantum LP solver

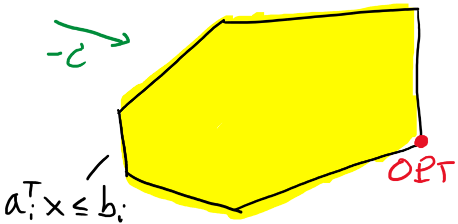
LPs and IPMs

Approximate Hessian

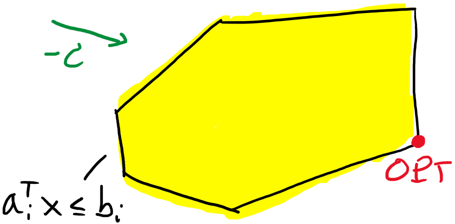
Approximate gradient

Quantum LP solver

Linear program (LP)

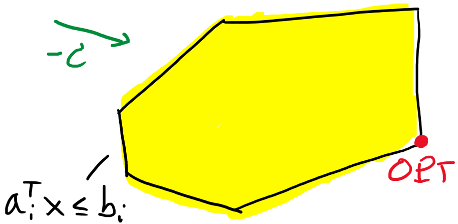


Linear program (LP)



$$\min \quad c^T x$$
$$\text{s.t.} \quad Ax \geq b$$

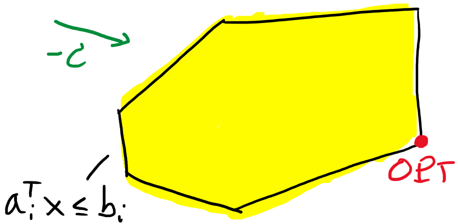
Linear program (LP)



$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \ge b \end{aligned}$$

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

Linear program (LP)



$$\min \quad c^T x$$
$$\text{s.t.} \quad Ax \geq b$$

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

= (constrained) convex optimization

Interior point method (IPM)

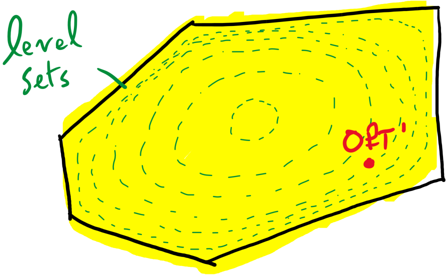
barrier f:

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

Interior point method (IPM)

barrier f:

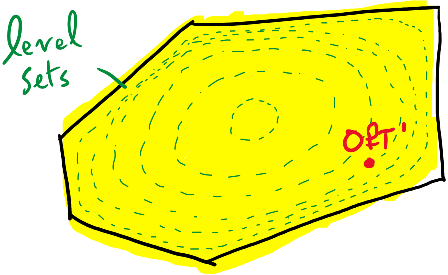
$f(x) \to \infty$ when $a_i^T x \to b_i$



Interior point method (IPM)

barrier f:

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

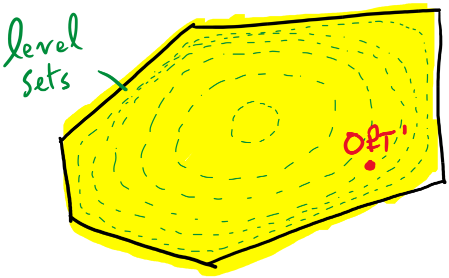


e.g., logarithmic barrier: $f(x) = -\sum_i \log(a_i^T x - b_i)$

Interior point method (IPM)

barrier f:

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

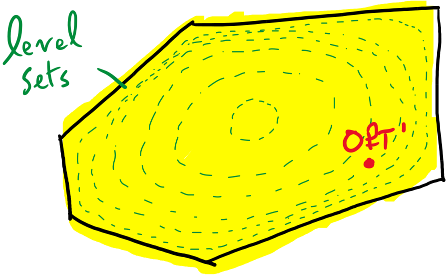


e.g., logarithmic barrier: $f(x) = -\sum_i \log(a_i^T x - b_i)$

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

Interior point method (IPM)

barrier f:
$f(x) \to \infty$ when $a_i^T x \to b_i$

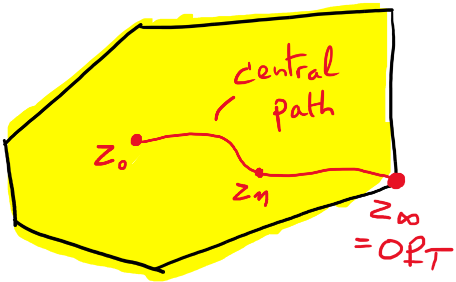


e.g., logarithmic barrier: $f(x) = -\sum_i \log(a_i^T x - b_i)$

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

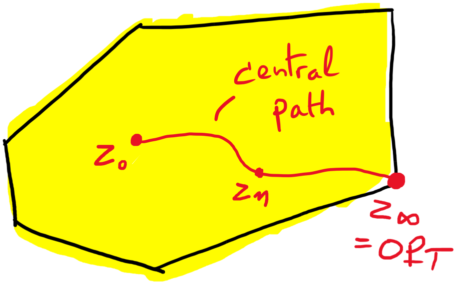
= *unconstrained* convex optimization

Interior point method (IPM)



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

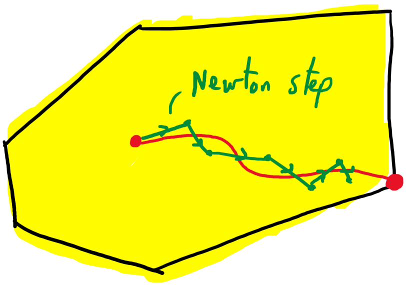
Interior point method (IPM)



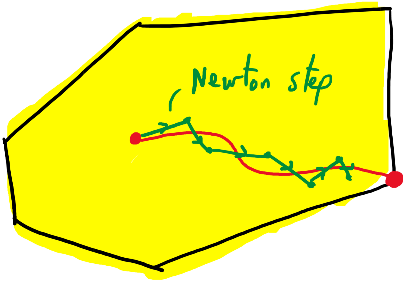
$$f_\eta(x) = f(x) + \eta \cdot c^T x$$

central path $\{z_\eta = \operatorname{argmin}_x f_\eta(x)\}_{\eta \geq 0}$

Path following



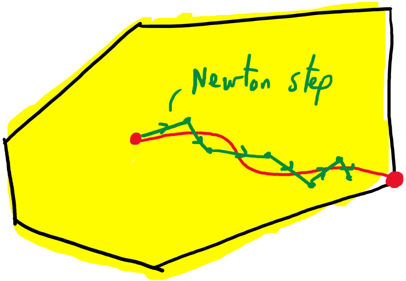
Path following



1. increase η:

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

Path following



1. increase η:

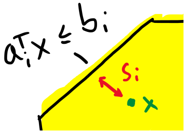
$$\eta' = (1+\gamma)\eta$$

2. Newton step:

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

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

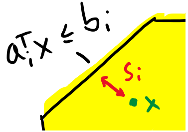
Path following



e.g., logarithmic barrier:

$$f(x) = -\sum_i \log(\underbrace{a_i^T x - b_i}_{\text{slack } s_i})$$

Path following



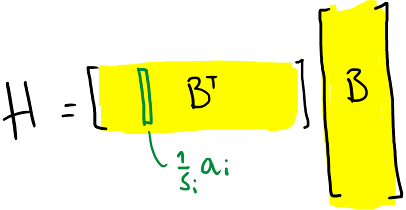
e.g., logarithmic barrier:

$$f(x) = -\sum_i \log(\underbrace{a_i^T x - b_i}_{\text{slack } s_i})$$

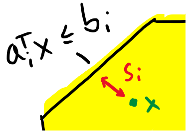
Hessian

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

$$(B^T)_{\cdot i} = \frac{1}{s_i} a_i$$

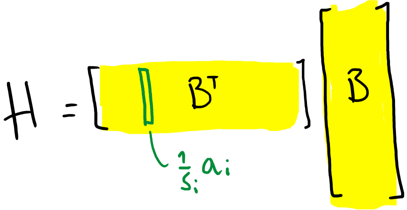


Path following



e.g., logarithmic barrier:

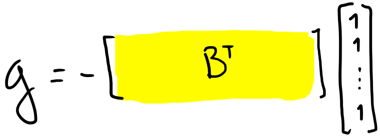
$$f(x) = -\sum_i \log(\underbrace{a_i^T x - b_i}_{\text{slack } s_i})$$



Hessian

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

$$(B^T)_{\cdot i} = \frac{1}{s_i} a_i$$



gradient

$$g(x) = -B^T \vec{1}$$

Runtime IPM

number of steps

$\sim$ number of increases η

Runtime IPM

number of steps

$\sim$ number of increases η

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

Runtime IPM

number of steps

$\sim$ number of increases η

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

$\sim$ number of increases η

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

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 '20

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

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

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

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

SOTA (classical)

Solving Tall Dense Linear Programs in Nearly Linear Time '20

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

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

Aaron Sidford
Stanford University, U.S.A.
sidford@stanford.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 '20

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

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

Aaron Sidford
Stanford University, U.S.A.
sidford@stanford.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**
$$x' = x - H(x)^{-1} g(x)$$

A Quantum Interior Point Method for LPs and SDPs*
'18

IORDANIS KERENIDIS and ANUPAM PRAKASH, CNRS, IRIF, Université Paris Diderot

quantum linear system solving + tomography

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

A Quantum Interior Point Method for LPs and SDPs* '18

IORDANIS KERENIDIS and ANUPAM PRAKASH, CNRS, IRIF, Université Paris Diderot

quantum linear system solving + tomography

! dependence on condition number $\kappa(H)$
($\to \infty$ as $x \to \mathrm{OPT}$)

* 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], . . .

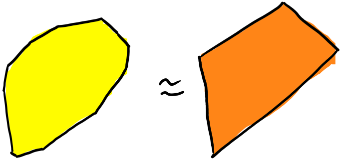
** non-IPM: multiplicative weights [Brandão-Svore ’17], [van Apeldoorn-Gilyén-Gribling-de Wolf-Brandão-Kalev-Li-Lin-Svore-Wu ’17], simplex method [Nannicini ’19], . . .

This work

quantum speedup
for **Newton step**
$x' = x - H(x)^{-1} g(x)$

This work

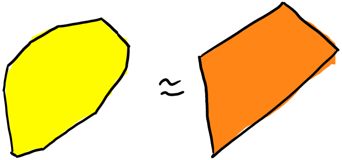
quantum speedup
for **Newton step**
$x' = x - H(x)^{-1} g(x)$



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

This work

quantum speedup
for **Newton step**
$x' = x - H(x)^{-1} g(x)$

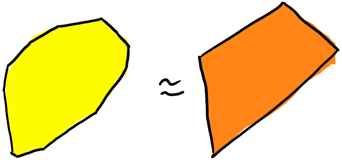


(i) approximate Hessian $H' \approx H$
(ii) approximate gradient $g' \approx g$
(using H' as preconditioner)

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

GOAT: $\Omega(\sqrt{nd})$ row queries

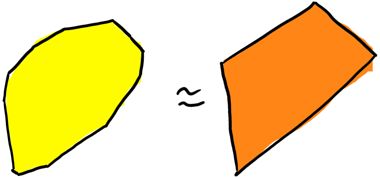
LPs and IPMs

Approximate Hessian

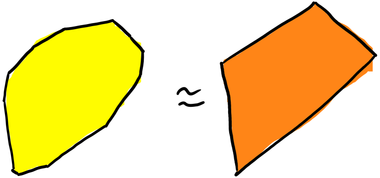
Approximate gradient

Quantum LP solver

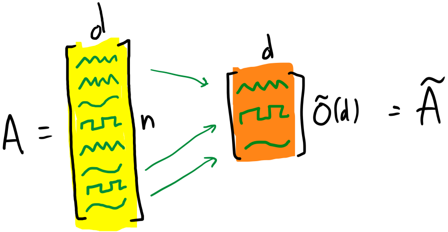
Spectral approximation



Spectral approximation



via constraint sampling

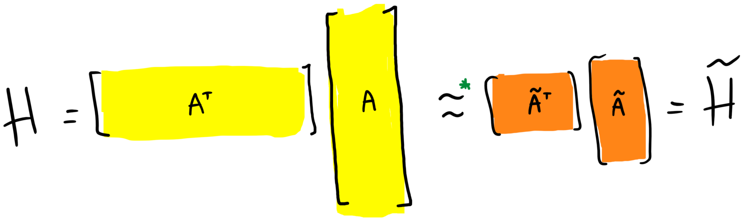


Spectral approximation

approximate Newton step

$$x' = x - \widetilde{H}^{-1} g$$

needs *spectral approximation*

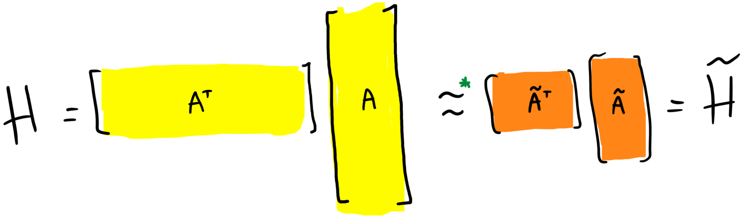


Spectral approximation

approximate Newton step

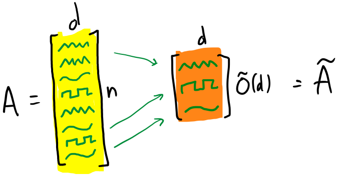
$$x' = x - \widetilde{H}^{-1} g$$

needs *spectral approximation*



* $H \approx \tilde{H} \quad \Leftrightarrow \quad \forall y\colon \; y^T H y = (1 \pm 0.1) y^T \tilde{H} y \quad \Leftrightarrow \quad 0.9\tilde{H} \preceq H \preceq 1.1\tilde{H}$

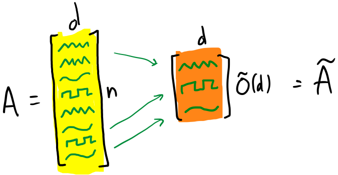
Spectral approximation



sampling via
"statistical leverage scores"

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

Spectral approximation

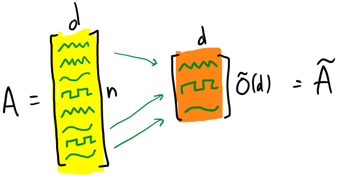


sampling via
"statistical leverage scores"

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

! chicken-and-egg

Spectral approximation



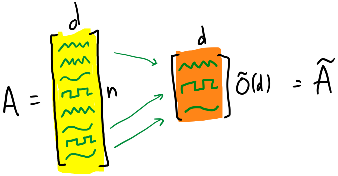
sampling via
"statistical leverage scores"

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

! chicken-and-egg

use
uniform subsampling + bootstrapping scheme
[Cohen-Lee-Musco-Musco-Peng-Sidford '14]

Spectral approximation

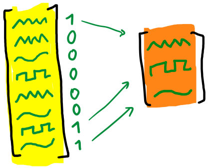


sampling via
"statistical leverage scores"

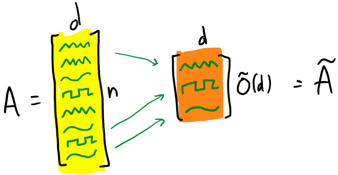
$$\sigma_i = a_i^T (A^T A)^{-1} a_i$$

! chicken-and-egg

use
uniform subsampling + bootstrapping scheme
[Cohen-Lee-Musco-Musco-Peng-Sidford '14]
+
Grover search



Spectral approximation

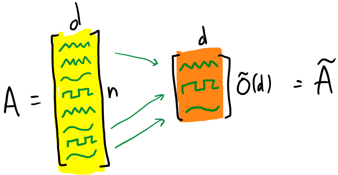


quantum algorithm:

– returns $\widetilde{A}$ with $\widetilde{O}(d)$ rows,
$\widetilde{A}$ spectral approximation of A

– makes $\widetilde{O}(\sqrt{nd})$ row queries

Spectral approximation

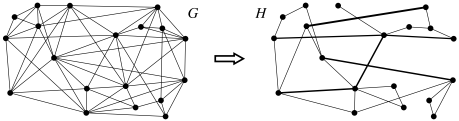


quantum algorithm:

– returns $\widetilde{A}$ with $\widetilde{O}(d)$ rows,
  $\widetilde{A}$ spectral approximation of A

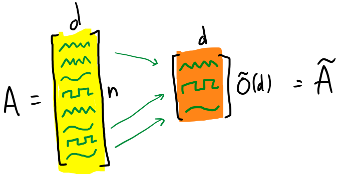
– makes $\widetilde{O}(\sqrt{nd})$ row queries

– generalizes graph sparsification



[Apers-de Wolf '19]

Spectral approximation

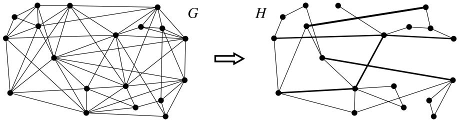


quantum algorithm:

– returns $\widetilde{A}$ with $\widetilde{O}(d)$ rows,
$\widetilde{A}$ spectral approximation of A

– makes $\widetilde{O}(\sqrt{nd})$ row queries

– generalizes graph sparsification



[Apers-de Wolf '19]

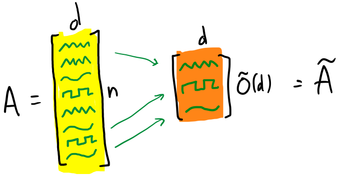
– applications in regression

[Submitted on 24 Nov 2023]

Revisiting Quantum Algorithms for Linear Regressions: Quadratic Speedups without Data-Dependent Parameters

Zhao Song, Junze Yin, Ruizhe Zhang

Spectral approximation

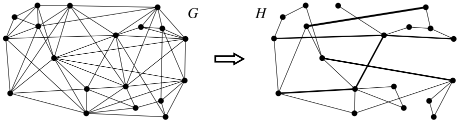


quantum algorithm:

– returns $\widetilde{A}$ with $\widetilde{O}(d)$ rows,
$\widetilde{A}$ spectral approximation of A

– makes $\widetilde{O}(\sqrt{nd})$ 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:

$$g = \begin{bmatrix} \; a_i \; & A^T \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = n \cdot \mathbb{E}_i \begin{bmatrix} a_i \end{bmatrix}$$

Approximate gradient

typical gradient:

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

→ quantum multivariate mean estimation:

[Cornelissen-Hamoudi-Jerbi ’22]

Approximate gradient

typical gradient:

$$g = \begin{bmatrix} & a_i & A^T & \end{bmatrix} \begin{bmatrix}1\\1\\1\\\vdots\\1\end{bmatrix} = n \cdot \mathbb{E}_i \begin{bmatrix} a_i \end{bmatrix}$$

$\rightarrow$ quantum multivariate mean estimation:

[Cornelissen-Hamoudi-Jerbi ’22]

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

$$\sqrt{d \, \mathrm{Tr}(\Sigma_X)}$$

Preconditioning

$$\left.\begin{array}{l}\text{gradient } g = \mathbb{E}[X] = \mathbb{E}_i[n\, a_i] \\ \text{covariance matrix } \Sigma_X \preceq \mathbb{E}[XX^T] = n\, A^T A\end{array}\right\} \Rightarrow \sqrt{dn\mathrm{Tr}(A^T A)} \text{ samples}$$

Preconditioning

$$\left.\begin{array}{l}\text{gradient } g = \mathbb{E}[X] = \mathbb{E}_i[n\, a_i] \\ \text{covariance matrix } \Sigma_X \preceq \mathbb{E}[XX^T] = n\, A^T A\end{array}\right\} \Rightarrow \sqrt{dn\mathrm{Tr}(A^T A)} \text{ samples}$$

! $\mathrm{Tr}(A^T A)$ introduces condition number :(

Preconditioning

gradient $g = \mathbb{E}[X] = \mathbb{E}_i[n\, a_i]$
covariance matrix $\Sigma_X \preceq \mathbb{E}[XX^T] = n\, A^T A$

$$\left.\begin{array}{l} g = \mathbb{E}[X] = \mathbb{E}_i[n\, a_i] \\ \Sigma_X \preceq \mathbb{E}[XX^T] = n\, A^T A \end{array}\right\} \Rightarrow \sqrt{dn\mathrm{Tr}(A^T A)} \text{ samples}$$

! $\mathrm{Tr}(A^T A)$ introduces condition number :(

$\downarrow$

precondition with spectral approximation:

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

Preconditioning

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

covariance matrix $\Sigma_X \preceq \mathbb{E}[XX^T] = n\, A^T A$

$\Bigg\} \Rightarrow \sqrt{dn\mathrm{Tr}(A^T A)}$ samples

! $\mathrm{Tr}(A^T A)$ introduces condition number :(

$\downarrow$

precondition with spectral approximation:

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

s.t.

$g = (\widetilde{A}^T \widetilde{A})^{1/2} \cdot \mathbb{E}[Y]$ and $\Sigma_Y \preceq 1.1\, n\, I_d$

Preconditioning

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

covariance matrix $\Sigma_X \preceq \mathbb{E}[XX^T] = n\, A^T A$

$\Bigg\} \Rightarrow \sqrt{dn\mathrm{Tr}(A^T A)}$ samples

! $\mathrm{Tr}(A^T A)$ introduces condition number :(

$$\downarrow$$

precondition with spectral approximation:

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

s.t.

$$g = (\widetilde{A}^T \widetilde{A})^{1/2} \cdot \mathbb{E}[Y] \qquad \text{and} \qquad \Sigma_Y \preceq 1.1\, n\, I_d$$

$\Rightarrow O(d\sqrt{n})$ samples

LPs and IPMs

Approximate Hessian

Approximate gradient

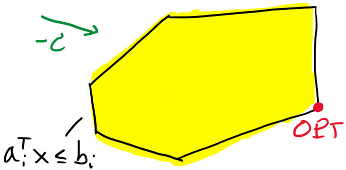
Quantum LP solver

Quantum LP solver

explicitly returns $\tilde{x}$ satisfying

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

and $\quad A\tilde{x} \leq b$

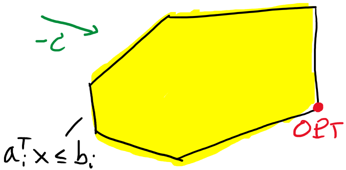


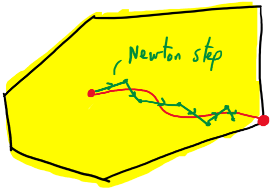
Quantum LP solver

explicitly returns $\tilde{x}$ satisfying

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

and $\qquad A\tilde{x} \leq b$





 # row queries*:

$$(\#\text{ steps}) \times (\#\text{ cost Newton step})$$

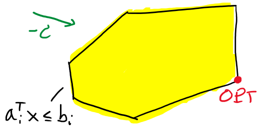
$$= \sqrt{d}\,\log(1/\varepsilon) \times \sqrt{n}d^{2.5} \in \widetilde{O}(\sqrt{n}d^3)$$

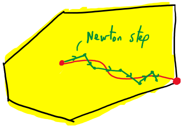
time complexity: $\sqrt{n}\,\log(1/\varepsilon)\,\mathrm{poly}(d)$

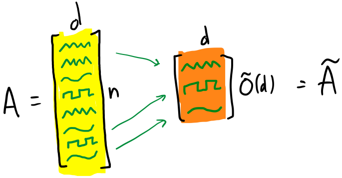
* using Lewis weight barrier.

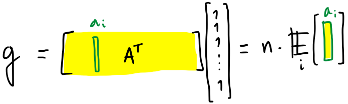
log-barrier: $\sqrt{n}\log(1/\varepsilon) \times \sqrt{n}d$, volumetric barrier: $(nd)^{1/4}\log(1/\varepsilon) \times \sqrt{n}d^2$

Summary and open questions

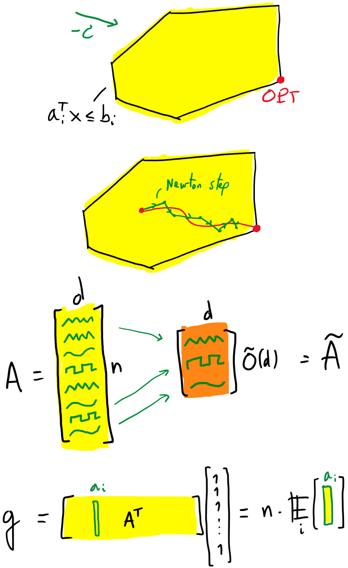






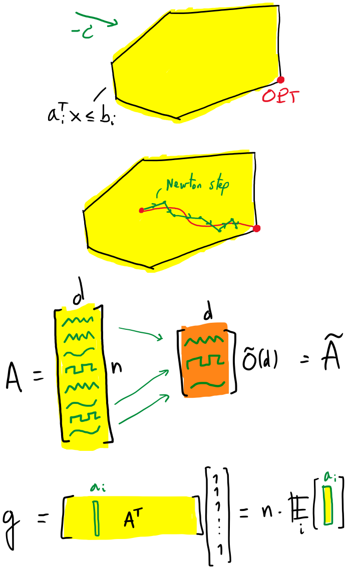


Summary and open questions



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

Summary and open questions



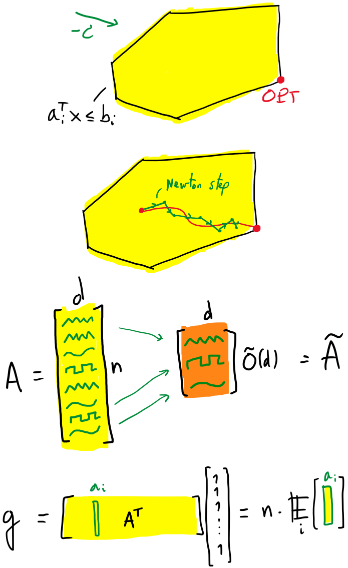
– **quantum IPM for solving LPs** (without condition numbers)

– **runtime** $(n \gg d)$

$$\sqrt{n} \cdot \mathrm{poly}(d) \cdot \log(1/\varepsilon)$$

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

Summary and open questions



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

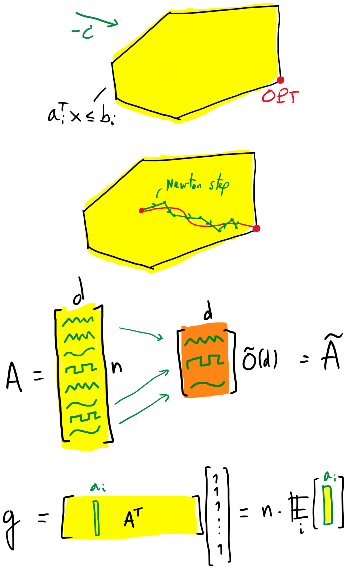
– **runtime** $(n \gg d)$

$$\sqrt{n} \cdot \mathrm{poly}(d) \cdot \log(1/\varepsilon)$$

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** $(n \gg d)$

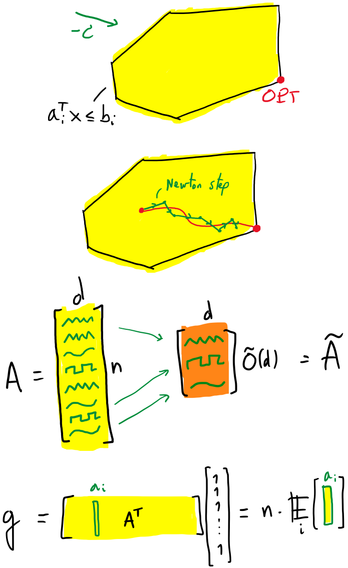
$$\sqrt{n} \cdot \mathrm{poly}(d) \cdot \log(1/\varepsilon)$$

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** $(n \gg d)$

$$\sqrt{n} \cdot \mathrm{poly}(d) \cdot \log(1/\varepsilon)$$

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

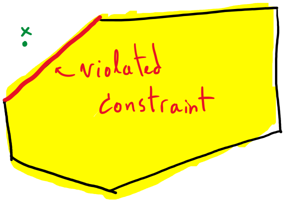
– **new tools**
spectral approximation (Grover)
approximate matrix-vector (mean estimation)

– **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