Quantum speedup for matrix approximation via uniform sampling

Hugo Thomas[1] – Sorbonne Université, Master 1 in Quantum Information, Paris

December 1, 2022

Abstract

This document is the report of the three-month internship done under the supervision of Simon Apers[2] at the Institut de Recherche en Informatique Fondamentale (IRIF), Paris. We first propose a quantum algorithm for the ε-spectral approximation of arbitrary tall and thin matrices of dimensions $n \times d$ with $n \gg d$ where each row has at most S nonzero entries in $\tilde{O}\big(S\frac{\sqrt{nd}}{\varepsilon}+d)\big)$ quantum queries and time $\tilde{O}(S\frac{\sqrt{nd}}{\varepsilon}+d^\omega)$, which generalizes results by Apers and de Wolf [AdW20] that consider undirected weighted graphs. It is based on an approach by Cohen et al. ([Coh+14]), where the time complexity is slightly improved by considering *e.g.* Johnson-Lindenstrauss transforms. In the second part, we expose a practical application in convex optimization : approximate the central path of Interior-Point Methods.

Contents

1 Introduction

In this work, tall and thin matrices $\boldsymbol{A} \in \mathbb{R}^{n\times d}$ with $n = poly(d)$ are considered. This is the case for many types of matrices arising in real world problems, such as the description of graphs or linear programs.

The objective is to compute an ε-spectral approximation of the matrix, containing a much smaller number of rows: in this case $\tilde{O}(d/\varepsilon^2)$ rows, as it has been shown by Spielman and Teng ([ST08]).

The proofs are postponed to the end on purpose, for the sake of readability and conciseness.

2 Leverage scores

In order to estimate the importance of a row of a matrix relative to the others, we define its leverage score.

Definition 2.1 (Leverage score)**.** *Given a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times d}$, the leverage score of a_i^T, the i-th row of $\boldsymbol{A}$, is defined as*

$$\tau_i := a_i^T(\boldsymbol{A^T A})^+ a_i$$

where $\boldsymbol{A}^+$ denotes the Moore-Penrose pseudoinverse of $\boldsymbol{A}$.

In the algorithm, approximate leverage scores are computed, *i.e.,* leverage scores of $\boldsymbol{A}$ according to some approximation $\boldsymbol{B}$, denoted *generalized leverage scores.*

Definition 2.2 (Generalized leverage scores)**.** *Let $\boldsymbol{A} \in \mathbb{R}^{n\times d}$ and $\boldsymbol{B} \in \mathbb{R}^{m\times d}$. The leverage score of the i-th rows of $\boldsymbol{A}$ according to $\boldsymbol{B}$ is*

$$\tau_i^{\boldsymbol{B}}(\boldsymbol{A}) = \begin{cases} a_i^T(\boldsymbol{B^T B})^+ a_i & \text{if } a_i \perp \ker \boldsymbol{B} \\ \infty & \text{otherwise} \end{cases} .$$

The case distinction comes from the fact that if $a_i \not\perp \ker \boldsymbol{B}$, the generalized leverage score would be null, while a_i could be the only row pointing in its direction, and thus should be kept.

For the sake of conciseness, if $\boldsymbol{B} = \boldsymbol{A}$, we denote $\tau_i^{\boldsymbol{B}}(\boldsymbol{A})$ by $\tau_i(\boldsymbol{A})$, and as long as the knowledge $\boldsymbol{A}$ is not important, we simply denote $\tau_i(\boldsymbol{A})$ by τ_i.

If $\boldsymbol{A}$ and $\boldsymbol{B}$ are relatively close, *i.e.* $\boldsymbol{B}$ is an λ-spectral approximation of $\boldsymbol{A}$, it holds that

$$\frac{1}{\lambda}\boldsymbol{A}^T\boldsymbol{A} \preccurlyeq \boldsymbol{B}^T\boldsymbol{B} \preccurlyeq \boldsymbol{A}^T\boldsymbol{A}\,,$$

[1] hugo.thomas.3@etu.sorbonne-universite.fr

[2] apers@irif.fr

which we write

$$\boldsymbol{A} \approx_\lambda \boldsymbol{B}\,.$$

Theorem 2.3 (Leverage score approximate)**.** *Let $\boldsymbol{A} \in \mathbb{R}^{n\times d}$ and $\boldsymbol{B} \in \mathbb{R}^{m\times d}$. If $\boldsymbol{A} \approx_\lambda \boldsymbol{B}$, then*

$$\tau_i(\boldsymbol{A}) \leq \tau_i^{\boldsymbol{B}}(\boldsymbol{A}) \leq \lambda \cdot \tau_i(\boldsymbol{A})\,.$$

See proof on page 11.

3 Grover search

Considering the following function

$$f : [n] \to \{0,1\} \quad \text{s.t.} \quad \big|\{i : f(i) = 1\}\big| = k\ ,$$

f is the indicator of the rows kept in the matrix. In order to construct the search oracle, we need to consider a list $\boldsymbol{z}$ whose z_i are uniform random number in $[0,1]$ for all $i \in [n]$[3]. f is so that

$$f(i) = \begin{cases} 1 & \text{if } z_i \leq p_i \\ 0 & \text{otherwise} \end{cases}\ . \tag{1}$$

Note that there are two possibilities to find the k marked elements.

3.1 Repeated search

With the naive approach of repeating k Grover searches, it is very likely to get less than k disctinct marked elements. To address this issue, one can repeat $\Theta(k \log k)$ times Grover search over the set $[n]$. This finds all the distinct k marked elements with a sufficiently high probability, see *e.g.* the coupon collector's problem. This requires $O(\sqrt{nk}\log k) = \tilde{O}(\sqrt{nk})$ queries to f.

It is however possible to reduce the number of queries to $O(\sqrt{nk})$.

3.2 Updating the list

The list $\boldsymbol{z}$ of *choices* is classically stored offline, and both the access to an element and an update require time $O(1)$. Thus, after a call to f, it is possible to efficiently update $\boldsymbol{z}$ in order to *unmark* the returned element. The following Algorithm 1 exposes how the list $\boldsymbol{z}$ is updated at each iteration.

Algorithm 1 QueryAndUpdate($\boldsymbol{z} = \{z_i\}, f$)

```
1: $i \leftarrow$ Grover search with $f$ over $[n]$
2: $z_i \leftarrow \infty$
3: return $i$
```

Step 2 ensures that the i-th element is not marked anymore. At the j-th call to the above procedure there remains $(k-j)$ marked elements, thus finding one of them requires time $O(\sqrt{\frac{n}{k-j}})$. A single call to QueryAndUpdate is thus as expensive as a call to f.

Proposition 3.1. *k calls to* QueryAndUpdate *find k distinct marked elements among n in $O(\sqrt{nk})$ quantum queries.*

See proof on page 11.

4 Sampling procedure

The sampling procedure is given query access to a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times d}$, a matrix $\boldsymbol{B} \in \mathbb{R}^{\tilde{O}(d)\times d}$ and a number of rows to be sampled k. It then outputs a constant-factor approximation $\hat{\boldsymbol{A}}$ of $\boldsymbol{A}$.

This sampling procedure, by using Grover's algorithm in the quantum setting, avoids the explicit construction of the vector of leverage scores.

Algorithm 2 Sample($\boldsymbol{A}$, $\boldsymbol{B}$, k)

```
Ensure: $\boldsymbol{z} = \{z_i \sim \mathcal{U}(0,1)\}_{i\in[n]}$          ▷ Stored offline.
Ensure: $\boldsymbol{A} \in \mathbb{R}^{n\times d}$, $\boldsymbol{B} \in \mathbb{R}^{\tilde{O}(d)\times d}$ such that $A \approx_\lambda B$
allocate $\hat{\boldsymbol{A}}$
define a function $f$ such that
    $f(i) = \begin{cases} 1 & \text{if } z_i \leq \min\{1, \alpha \cdot a_i^T(\boldsymbol{B}^T\boldsymbol{B})^+ a_i \cdot c\log d\}, \\ 0 & \text{otherwise.} \end{cases}$
repeat
    $i \leftarrow$ QueryAndUpdate($\boldsymbol{z}$,$f$)
    add row $a_i$ to $\hat{\boldsymbol{A}}$, rescaled by $p_i^{-1/2}$
until sampled $O(k)$ rows
return $\hat{\boldsymbol{A}}$
```

One thing to note is that the parameter k is normally set to $O(d\log d)$ (as will be seen in more detail in Section 5), except when computing an ε-spectral approximation, in which case k is set to $O(\frac{d\log d}{\varepsilon})$.

5 Uniform sampling

Uniform sampling does not uses (R,k)-reductions *i.e.,* Johnson-Lindenstrauss random projections as depicted in [LMP13], and thus preserves the input matrix sparsity pattern, which is usefull especially in the graph case. This will allow us, in this case, to compute the leverage score of a single row in $O(1)$ quantum queries.

In our setting, computing leverages scores according to an approximation $\tilde{\boldsymbol{A}}$ of the input matrix $\boldsymbol{A}$ requires to sample

$$c\log d\sum_i \tilde{\tau}_i$$

rows for a fixed constant c, where the $\tilde{\tau}_i$ are computed according to $\tilde{\boldsymbol{A}}$.

[3] $[n]$ shortens notation of the set $\{0,\cdots,n-1\}$

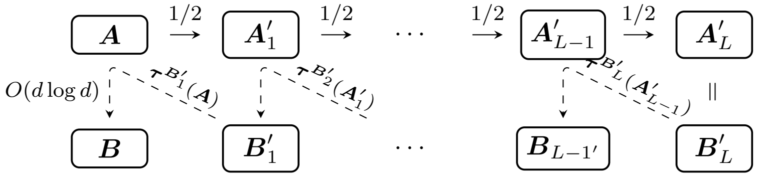

Figure 1: Structure of the recursive calls to Approximate. The leverage score vector $\boldsymbol{\tau}$ is indeed not explicitely computed, but is used for Grover search. The resulting matrix is $\boldsymbol{B} \approx_2 \boldsymbol{A}$.

In order to prove correctness of the algorithm we expose, it is convenient to repesent the sampling of a matrix through the product with a diagonal matrix, that we define below.

Definition 5.1. *Let* `Sample(`$\boldsymbol{\tau}, \alpha$`)` *be a procedure that returns a diagonal matrix S with independently chosen entries, and such $S_{ii} = p_i^{-1/2}$ with probability p_i and* 0 *otherwise, where $p_i = \min\{1, \alpha c \tau_i \log d\}$ for a fixed constant c.*

In the quantum setting, this is implemented by the procedure Sample shown in Algorithm 2, thus there is no need to explicitly construct the output matrix of the procedure. Nevertheless, it is necessary to show that the two representations are equivalent.

Claim 5.2. *Given a matrix $\boldsymbol{A}$, on one hand let $\boldsymbol{S} =$* `Sample(`$\boldsymbol{1}, O(m/n)$`)` *and consider $\boldsymbol{SA}$; on the other hand, let $\boldsymbol{A'}$ be a uniform ramdom subset of $O(m)$ rows of $\boldsymbol{A}$. Sampling according to $\boldsymbol{\tau}^{\boldsymbol{SA}}$ is equivalent to sampling according to $\boldsymbol{\tau}^{\boldsymbol{A'}}$.*

See proof on page 11. As long as the leverage score approximates used to sample are *upper bounds* on the true leverage score, *i.e.,* for all i, $\tilde{\tau}_i \geq \tau_i$, sampling $\boldsymbol{A}$ according to them yields a constant-factor spectral approximation [LMP13].

Theorem 5.3 ([Coh+14])**.** *Let $\boldsymbol{A} \in \mathbb{R}^{n\times d}$. Sampling uniformly at random $O(m)$ rows from $\boldsymbol{A}$ to form $\tilde{\boldsymbol{A}}$, implies, supposing one computes $\{\tilde{\tau}_i\}$ thanks to $\tilde{\boldsymbol{A}}$, that*

$$\mathbb{E}\left[\sum_i \tilde{\tau}_i\right] \leq \frac{nd}{m}\,.$$

The above theorem enables us to conclude on the number of rows to be sampled given m.

Theorem 5.4 ([Coh+14])**.** *Let $\boldsymbol{A} \in \mathbb{R}^{n\times d}$ suppose we sample uniformly $O(m)$ rows to form $\boldsymbol{A'}$. Computing $\tilde{\tau}_i = \min\{1, \tau_i^{\boldsymbol{A'}}(\boldsymbol{A})\}$ and sampling $\boldsymbol{A}$ accordingly returns with high probability a constant factor spectral approximation of $\boldsymbol{A}$ with at most $O(\frac{nd\log d}{m})$ rows.*

Thus Theorem 5.4 implies correctness of Algorithm 3.

Algorithm 3 Approximate($\boldsymbol{A}$)

```
Ensure: $\boldsymbol{A} \in \mathbb{R}^{n\times d}$
$\boldsymbol{A'} \leftarrow$ uniformly sample $\frac{n}{2}$ rows of $\boldsymbol{A}$
if $\boldsymbol{A'}$ has more than $O(d\log d)$ rows then
    $\boldsymbol{B'} \leftarrow$ Approximate($\boldsymbol{A'}$)
else
    $\boldsymbol{B'} \leftarrow \boldsymbol{A'}$
$\boldsymbol{B} \leftarrow$Sample($\boldsymbol{A}, \boldsymbol{B'}, O(d\log d)$)
return $\boldsymbol{B}$
```

Indeed, sampling $O(d\log d)$ rows is enough to obtain a constant factor approximation: letting $m = O(n/2)$ in Theorem 5.3 yields

$$\mathbb{E}\left[\sum_i \tilde{\tau}_i\right] = O(d)\,,$$

hence

$$\log d \sum_i \tilde{\tau}_i = O(d\log d)\,.$$

The resulting matrix has $O(d\log d)$ rows and there are $O\big(\log(\frac{n}{d\log d})\big)$ recursive calls to Approximate. It is important to note that $\boldsymbol{B}$ is a constant-factor-approximation of the matrix $\boldsymbol{A}$, which we write

$$\boldsymbol{A} \approx_{O(1)} \text{Approximate}(\boldsymbol{A})\,.$$

Figure 1 shows graphically how the recursive sampling works.

It is possible to further speed up the calculation of leverage score by using Johnson-Lindenstrauss transfom on each $\boldsymbol{B}'_l$, for all $1 \leq l \leq L$.

5.1 Johnson-Lindenstrauss tranform

For the sake of completeness, the Johnson-Lindenstrauss lemma is recalled in Section A.

5.1.1 Leverage score as a squared norm

Let $\boldsymbol{A} \in \mathbb{R}^{n\times d}$, and recall that, for a_i^T the i-th row of $\boldsymbol{A}$, its leverage score τ_i is defined in Definition 2.1 as

$$\tau_i = a_i^T(\boldsymbol{A}^T\boldsymbol{A})^+ a_i\,.$$

In order to express τ_i as a squared norm, let us denote

$$\boldsymbol{\chi}_i := \boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^+ a_i\,, \tag{2}$$

which yields the following proposition:

Proposition 5.5. *Given $\boldsymbol{A}$ an input matrix, one can compute the leverage scores of $\boldsymbol{A}$'s rows as follows:*

$$\tau_i(\boldsymbol{A}) = \|\boldsymbol{\chi}_i\|_2^2 \,.$$

See proof on page 11.

5.1.2 Application of the JL transfom

For our purpose, consider a matrix $\boldsymbol{A} \in \mathbb{R}^{\tilde{O}(d)\times d}$ and let $\boldsymbol{\chi}_i$ as introduced in Equation 2; thus, by Proposition 5.5, $\|\boldsymbol{\chi}_i\|_2^2$ is exactly τ_i. Also, since we can write $\Pi\boldsymbol{\chi}_i = \Pi\boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^+a_i$ with $\Pi\boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^+ \in \mathbb{R}^{\tilde{O}(\varepsilon^{-2})\times d}$, we denote $\|\Pi\boldsymbol{\chi}_i\|_2^2$ by $\hat{\tau}_i$. Thereby, we can restate Lemma A.1 as follows:

Lemma 5.6 (DJL lemma, restated)**.** *For any $0 < \varepsilon < 1$, $\delta < 1/2$ and $d \in \mathbb{N}$, there exists a distribution over $\mathbb{R}^{k\times d}$ from which the matrix Π is drawn such that for $k = O\big(\varepsilon^{-2}\log(\delta^{-1})\big)$ and any vector $\chi \in \mathbb{R}^d$, the following claim holds:*

$$\mathbb{P}\Big(\, |\hat{\tau}_i - \tau_i| > \varepsilon \cdot \tau_i \Big) < \delta \,.$$

Let us denote by X_i the event «$|\hat{\tau}_i - \tau_i| > \varepsilon \cdot \tau_i$». Thus, taking the union bound over all possible leverage scores, we have

$$\mathbb{P}(\bigcup_{i=1}^{n} X_i) \leq \sum_{i=1}^{n}\mathbb{P}(X_i) < \sum_{i=1}^{n}\delta = n\delta$$

Hence, setting $\delta = n^{-2}$ yields that, with probability $\geq 1 - \frac{1}{n}$, all leverage scores in the reduced space are an ε-approximation of the original ones, *i.e.*, for all $i \in [n]$,

$$|\hat{\tau}_i - \tau_i| \leq \varepsilon \cdot \tau_i \,. \tag{3}$$

Proposition 5.7. *It is possible to compute a single leverage score in time $\hat{O}(\varepsilon^{-2}S)$, and such leverage score can be used to sample the input matrix and obtain a spectral approximation.*

See proof on page 12.

5.1.3 Time complexity of the scheme

Given as input a matrix $\boldsymbol{A} \in \mathbb{R}^{\tilde{O}(d)\times d}$, there is a procedure with running time $\tilde{O}(d\varepsilon^{-2})$ that returns the matrix $\Pi \in \mathbb{R}^{\tilde{O}(\varepsilon^{-2})\times\tilde{O}(d)}$ – this procedure simply consists in setting each entry of Π to independant Gaussian random variable [DG03]. It takes time $\tilde{O}(d^\omega)$ to obtain $\boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^+$ and additional $\tilde{O}(d^2\varepsilon^{-2})$ to compute the matrix-matrix product $\Pi\boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^+$. Thus, computing a single leverage score in time $\tilde{O}(\varepsilon^{-2}S)$ requires a preprocessing time of $\tilde{O}(d^\omega)$. It suffices to set $0 < \varepsilon < 1$ to be constant, so that each leverage score requires time $\tilde{O}(S)$ to be computed. Doing so it holds, for any $0 \leq l \leq L$, that $\boldsymbol{B}_l'$ is still $O(1)$-spectral approximation of $\boldsymbol{A}_l'$ since the number of sampled rows stays unchanged and the leverage score used remains upper bounds on the actual approximations, *i.e.*, upper bounds on the true leverage scores.

5.2 Overall query complexity

Let L be the number of iterations required to obtain a matrix of $O(d\log d)$ rows. Thus, it holds that

$$\frac{n}{2^L} = O(d\log d)\,,$$

hence, there is a total of

$$L = O(\log(\frac{n}{d\log d})) = \tilde{O}(1)$$

iterations.

Given a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times d}$, let $S \leq d$ such that each row of $\boldsymbol{A}$ has at most S nonzero entries. Each call to SAMPLE computes $\tilde{O}(d)$ scores, where each score requires $O(S)$ queries. Since each matrix has fewer than n rows, the total number of rows of all of the matrices together can be bounded by $Ln = \tilde{O}(n)$. Thus, by Proposition 3.1, sampling $\tilde{O}(d)$ rows among $O(n)$ thoughout the L iterations makes a total of $\tilde{O}(S\sqrt{nd})$ quantum queries.

Furthermore, it is possible to store implicitly each reduced matrix, by implicitly keeping track of the discarded rows through a string as shown in Section 6.1. However the last matrix has to be explicitly written in order to compute $(\boldsymbol{A_L'}^T\boldsymbol{A_L'})^+$: this requires additional $\tilde{O}(dS)$ queries. Hence, the query complexity of Algorithm 3 is $\tilde{O}(S(\sqrt{nd}+d))$.

6 Time complexity analysis

To obtain the time complexity of the whole procedure, it suffices considering the time of a single iteration, and since there is a logarithmic number of iterations, considering all of them fits in the $\tilde{O}$ notation.

In order to end up with a sublinear time algorithm, it is necessary to avoid representing explicitly the intermediate matrices $\boldsymbol{A}_l'$.

6.1 Data structure for matrix sampling

In order to represent each subsampling $\boldsymbol{A}_l'$ of the original matrix, a random n-bit-string Z_l is associated to the l-th iteration, and is such that

$$(Z_l)_i{}^4 = \begin{cases} 1 & \text{w.p. } \frac{1}{2}\,, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Claim 6.1. *Since the set of the rows of $\boldsymbol{A}_l'$ must be a subset of those of $\boldsymbol{A}_{l-1}'$, the matrix $\boldsymbol{A}_l'$ can be represented through a bit-string, denoted Z_l', obtained by doing the element-wise product (logical conjunction) of the* previous *strings. That is,*

$$(Z_l')_i = \bigwedge_{k\leq l}(Z_k)_i, \quad \forall i \in [n]\,. \tag{5}$$

[4] Given a string z, $(z)_i$ denotes its i-th bit.

See proof on page 12. It would not be possible to explicitly compute this string while staying in sub-linear time. In any cases, at each iteration $\tilde{O}(\sqrt{nd}) < n$ queries are done to the matrix, so the full string doesn't need to be explicitly constructed.

6.2 k-wise independent strings

We use k-wise independent string to simulate access to the random string. The result we use is the following theorem from [Zha15].

Theorem 6.2. *Any q-query algorithm cannot distinguish between a uniformly random string and a $2q$-wise independent string.*

This allows us to discard an explicit string Z_l of size $O(n)$ and use instead a k-wise independent string with $k \in \tilde{O}(\sqrt{nd})$. This is achived thanks to k-independent hashing functions whose definition is recalled in Section B.

From Theorem B.2, we can assert that such a data structure can be constructed in time $\tilde{O}(\sqrt{nd})$. An access to a bit is done in time $\tilde{O}(1)$ and thus obtaining a single bit of Z_l' as depicted in Equation 5 takes time $\tilde{O}(1)$ since $L = \tilde{O}(1)$.

Hence, at first – recalling the result of Section 5.1.3 – we compute a single leverage score in time $O(S)$ with an additional preprocessing time of $O(d^\omega)$. Thus the whole sampling procedure, which yields an approximation $\boldsymbol{B}$ of the input matrix $\boldsymbol{A}$ the $\tilde{O}(d)$ rows and such that

$$\boldsymbol{A} \approx_{O(1)} B$$

is achieved in time $\tilde{O}(S\sqrt{nd} + d^\omega)$.

6.3 ε-spectral approximation

The ouput matrix of Approximate is a $O(1)$-spectral approximation of the input matrix $\boldsymbol{A}$ with $O(d \log d)$ rows. In order to obtain, in the end, an ε-spectral approximation $\boldsymbol{B}$ of $\boldsymbol{A}$, *i.e.,* a matrix $\boldsymbol{B} \in \mathbb{R}^{O(\varepsilon^{-2} d \log d) \times d}$ such that, for $\varepsilon > 0$ and for all $\chi \in \mathbb{R}^d$,

$$(1-\varepsilon)\chi^T \boldsymbol{A}\chi \leq \chi^T \boldsymbol{B}\chi \leq (1+\varepsilon)\chi^T \boldsymbol{A}\chi\,,$$

it suffices, with input $\boldsymbol{A}$ and as long as we obtained $\boldsymbol{A} \approx_{O(1)} \boldsymbol{B}$, to sample $O(\varepsilon^{-2} d \log d)$ rows of $\boldsymbol{A}$ according to $\boldsymbol{\tau}^{\boldsymbol{B}}(\boldsymbol{A})$. This additional step requires $\tilde{O}(S\frac{\sqrt{nd}}{\varepsilon})$ quantum queries and supplementary time of $\tilde{O}(S\frac{\sqrt{nd}}{\varepsilon} + d^\omega)$.

7 Application

There are several applications of matrix sparsification, and the one we chosed to depict is convex optimization, and more precisely the interior points methods (IPM). A quick summary on IPM – with the explicit algorithm – is provided in Section C.

We denote herin the distance between two vectors $u, v \in \mathbb{R}^n$ by the following weighed norm with respect to the operator Q, which we denote the Q-induced norm,

$$\|u - v\|_Q = \sqrt{(u-v)^T Q (u-v)}\,, \tag{6}$$

to quantify the convergence rate. For our purpose, we denote by x the actual *current* value of Newton's iterations, by y the approximated one, and by x' and y' the result of a Newton's iteration starting from x and y respectively; the start superscript denotes the minimizer of Φ. We want to bound $\|y' - x^*\|$, so that it describes a path that converges towards to the actual minimizer.

7.1 Approximate Hessian

Considering a λ-spectral approximation of $S^{-1}B$, and taking into account that our Hessian is $H = B^T S^{-2} B$, we denote the approximated Hessian $\tilde{H}$ for some $\lambda \geq 1$. First, note that this implies

$$\begin{aligned} & \frac{1}{\lambda} H \preccurlyeq \tilde{H} \preccurlyeq H \\ \Leftrightarrow \quad & H^{-1} \preccurlyeq \tilde{H}^{-1} \preccurlyeq \lambda H^{-1}\,. \end{aligned} \tag{7}$$

At first, we consider an iteration where the initial points are equal. In order to use the above inequality, we need to express $\|x' - y'\|$ in terms of both Hessians, *i.e.,* considering Newton's step. That is,

$$\begin{aligned} \|x' - y'\| &= \left\| \left(x - H^{-1}\nabla\Phi_\mu(x)\right) - \left(y - \tilde{H}^{-1}\nabla\Phi_\mu(y)\right) \right\| \\ &= \left\| x - H^{-1}\nabla\Phi_\mu(x) - x + \tilde{H}^{-1}\nabla\Phi_\mu(x) \right\| \\ &= \left\| \tilde{H}^{-1}\nabla\Phi_\mu(x) - H^{-1}\nabla\Phi_\mu(x) \right\| \\ &= \left\| \left[\tilde{H}^{-1} - H^{-1}\right]\nabla\Phi_\mu(x) \right\|\,. \end{aligned} \tag{8}$$

Thus, it suffices to bound $\left[\tilde{H}^{-1} - H^{-1}\right]$ to obtain bounds on $\|x' - y'\|$. A straightforward consequence of Equation 7, is that

$$\mathbf{0} \preccurlyeq \tilde{H}^{-1} - H^{-1} \preccurlyeq (\lambda - 1) H^{-1}\,,$$

where $\mathbf{0}$ is a full-zero matrix. Therefore, it holds for all vectors, and especially $\nabla\Phi_\mu(x)$, that

$$0 \leq \left\| \left[\tilde{H}^{-1} - H^{-1}\right]\nabla\Phi_\mu(x) \right\| \leq \left\| (\lambda - 1) H^{-1}\nabla\Phi_\mu(x) \right\|\,,$$

Note that $\left\|H^{-1}\nabla\Phi_\mu(x)\right\|$ is equal to $\|x' - x\|$ *i.e.,* the length of the *actual* Newton's step. As long as $\lambda < 2$, y' is closer to x' than was x. The above equation can thus be equivalently rewritten thanks to Equation 8 as

$$0 \leq \|x' - y'\| \leq (\lambda - 1)\,\|x' - x\|\,. \tag{9}$$

In other words, it is possible to bound the distance bewteen x', the actual point we would like to obtain, and its approximation y' within a multiplicative factor as long as we consider λ-spectral approximation of the Hessian.

In order to prove convergence of the Newton's method when we consider λ-spectral approximation of the Hessian, we bound $\|y' - x^*\|$ in terms of $\|x - x^*\|$. It holds that

$$\begin{aligned} \|y' - x^*\| &= \|y' - x' + x' - x^*\| \\ &\leq \|y' - x'\| + \|x' - x^*\| \\ &\leq (\lambda - 1)\|x - x'\| + \frac{1}{2}\|x - x^*\|^2 \\ &\leq (\lambda - 1)\|x - x^*\| + \frac{1}{2}\|x - x^*\| \\ &= (\lambda - \frac{1}{2})\|x - x^*\| \qquad , \end{aligned} \tag{10}$$

which bounds the step of an approximated iteration. However, this enforces $1 \leq \lambda < \frac{3}{2}$ (*i.e.,* choose $\lambda = 1 + \varepsilon$ for an $\varepsilon \ll \frac{1}{2}$) in order to have $\lambda - \frac{1}{2} < 1$ and hence a step *towards* the minimizer y^* (y' in a *ball* of radius $< \|x - x^*\|$ around x^*).

7.2 Gradient approximation

The goal here is to obtain an approximation of $\nabla\Phi$, the gradient involved in the interior points method.

7.2.1 Initial approach

To do so, we'll consider a vector of random variables, such that its mean is equal to our gradient, using result of Cornelissen, Hamoudi, and Jerbi which exposes a quantum algorithm for estimating the mean of a n-dimensional random variable; they provide the following informal theorem.

Theorem 7.1 ([CHJ22])**.** *Given a n-dimensional random variable X, there exists a quantum algorithm that returns with high probability an estimate $\tilde{\mu}$ of the mean μ of X such that*

$$\|\tilde{\mu} - \mu\|_2 \leq \frac{\sqrt{n\mathrm{tr}(\Sigma_X)}}{T}$$

in $\tilde{O}(T)$ queries as long as $T > n$.

Note that the bound is obtained in 2-norm, but the convergence rate of Newton's method is in terms of the H-norm. As usual, we use $B \in \mathbb{R}^{m\times n}, s \in \mathbb{R}^m$ and $H = B^T diag(s)^{-2} B$ where we denote $diag(s)$ by S. Let $X = B^T S^{-1}\left|e\right\rangle$, where $e \sim \mathcal{U}[m]$. We denote $h := \sum_e B^T S^{-1}\left|e\right\rangle = B^T S^{-1}\left|\mathbf{1}\right\rangle$, where $\left|\mathbf{1}\right\rangle$ is the full-one vector, thus $\mu_X := \mathbb{E}[X] = \frac{1}{m}B^T S^{-1}\left|\mathbf{1}\right\rangle = \frac{1}{m}h$. We denote by Σ_Z the covariance matrix of the n-dimensional random variable Z. The goal is to obtain an $\tilde{h}$ such that $\|h - \tilde{h}\|_{H^{-1}} \leq \varepsilon$. In other words, we want $\tilde{\mu}_X$ such that

$$\|\mu_X - \tilde{\mu}_X\|_{H^{-1}} \leq \frac{\varepsilon}{m}$$

which, by using Proposition D.5, means

$$\left\|H^{-\frac{1}{2}}\mu_X - H^{-\frac{1}{2}}\tilde{\mu}_X\right\|_2 \leq \frac{\varepsilon}{m} ;$$

that is,

$$\|\mu_Y - \tilde{\mu}_Y\|_2 \leq \frac{\varepsilon}{m} , \tag{11}$$

for a certain random variable Y. As such, let $Y = H^{-\frac{1}{2}}X = H^{-\frac{1}{2}}B^T S^{-1}\left|e\right\rangle$ where $e \sim \mathcal{U}[m]$, hence $\mu_Y = H^{-\frac{1}{2}}\mu_X$; we use the result of [CHJ22] to get the $\tilde{\mu}_Y$.

The covariance matrix of Y is

$$\begin{aligned} \Sigma_Y &= \mathbb{E}\left[YY^T\right] - \mathbb{E}\left[Y\right]\mathbb{E}\left[Y^T\right] \\ &= H^{-\frac{1}{2}}\Sigma_X H^{-\frac{1}{2}} - \mathbb{E}\left[Y\right]\mathbb{E}\left[Y^T\right] , \end{aligned}$$

where

$$\Sigma_X = \frac{1}{m}H - \frac{1}{m^2}hh^T \quad \text{and} \quad \mathbb{E}\left[Y\right] = \frac{1}{m}H^{-\frac{1}{2}}h ,$$

hence

$$\Sigma_Y = \frac{1}{m}\mathbb{1} - \frac{2}{m^2}H^{-\frac{1}{2}}hh^T H^{-\frac{1}{2}} \preccurlyeq \mathbb{E}\left[YY^T\right] .$$

Claim 7.2. $\Sigma_Y \preccurlyeq \frac{1}{m}\mathbb{1}$.

See proof on page 12. Using Theorem 7.1 with the random variable Y yields $\tilde{\mu}_Y$ such that

$$\|\mu_Y - \tilde{\mu}_Y\|_2 \leq \frac{\sqrt{n\mathrm{tr}(\Sigma_Y)}}{T} . \tag{12}$$

Having $\Sigma_Y \preccurlyeq \frac{1}{m}\mathbb{1}$ implies $\mathrm{tr}(\Sigma_Y) \leq \frac{n}{m}$ since $\Sigma_Y \in \mathbb{R}^{n\times n}$. Following our initial assumption – Equation 11 – we want the left side of Equation 12 to be smaller than $\frac{\varepsilon}{m}$, *i.e.*,

$$\frac{n}{\sqrt{m}T} \leq \frac{\varepsilon}{m} ,$$

which is achieved if and only if

$$T \geq \frac{n\sqrt{m}}{\varepsilon} .$$

7.2.2 Using sparsified Hessian

We consider we have a λ-spectral approximation $\tilde{H}$ of H such that

$$\frac{1}{\lambda}H \preccurlyeq \tilde{H} \preccurlyeq H , \tag{13}$$

which we can rephrase

$$\frac{1}{\sqrt{\lambda}}H^{\frac{1}{2}} \preccurlyeq \tilde{H}^{\frac{1}{2}} \preccurlyeq H^{\frac{1}{2}} , \tag{14}$$

and we'll use another straightforward formulation, *i.e.*,

$$H^{-\frac{1}{2}} \preccurlyeq \tilde{H}^{-\frac{1}{2}} \preccurlyeq \sqrt{\lambda}H^{-\frac{1}{2}} . \tag{15}$$

In a similar manner as in Section 7.2.1, we define $\tilde{Y} = \tilde{H}^{-\frac{1}{2}}X = \tilde{H}^{-\frac{1}{2}}B^T S^{-1}\left|e\right\rangle$ where $e \sim \mathcal{U}[m]$. This yields

$$\begin{aligned} \mathbb{E}\left[\tilde{Y}\tilde{Y}^T\right] &= \tilde{H}^{-\frac{1}{2}}\mathbb{E}\left[XX^T\right]\tilde{H}^{-\frac{1}{2}} \\ &= \frac{1}{m}\tilde{H}^{-\frac{1}{2}}H\tilde{H}^{-\frac{1}{2}} . \end{aligned}$$

From Proposition D.4, we deduce

$$\frac{1}{m}\mathbb{1} \preccurlyeq \mathbb{E}\left[YY^T\right] \preccurlyeq \frac{\lambda}{m}\mathbb{1}.$$

Tracing each part, and using Claim 7.2 yields

$$\mathrm{tr}(\Sigma_{\tilde{Y}}) \leq \frac{n\lambda}{m}\,.$$

The Theorem 7.1 ensures the existence of $\tilde{\mu}_{\tilde{Y}}$ such that

$$\|\mu_{\tilde{Y}} - \tilde{\mu}_{\tilde{Y}}\|_2 \leq \frac{\sqrt{n\mathrm{tr}(\Sigma_{\tilde{Y}})}}{T} \leq \frac{n\sqrt{\lambda}}{\sqrt{m}T}.$$

Recall that we want $\|\mu_{\tilde{Y}} - \tilde{\mu}_{\tilde{Y}}\|_2 \leq \frac{\varepsilon}{m}$. As such, we set

$$T \geq \frac{n\sqrt{m\lambda}}{\varepsilon}\,.$$

Therefore, choosing T as defined above yields

$$\|\mu_{\tilde{Y}} - \tilde{\mu}_{\tilde{Y}}\|_2 = \left\|\tilde{H}^{-\frac{1}{2}}(\mu_X - \tilde{\mu}_X)\right\|_2 = \|\mu_X - \tilde{\mu}_X\|_{\tilde{H}^{-1}} \leq \frac{\varepsilon}{m},$$

and consequently

$$\|h - \tilde{h}\|_{\tilde{H}^{-1}} \leq \varepsilon\,, \tag{16}$$

which is a quantity preserved by the initial sparsification: Equation 13 ensures that for all $\chi \in \mathbb{R}^n$, the following holds :

$$\sqrt{\chi^T H^{-1}\chi} \leq \sqrt{\chi^T \tilde{H}^{-1}\chi} \leq \sqrt{\lambda} \cdot \sqrt{\chi^T H^{-1}\chi}\,,$$

hence, by definition of $\|\cdot\|_{H^{-1}}$, we have

$$\|h - \tilde{h}\|_{H^{-1}} \leq \|h - \tilde{h}\|_{\tilde{H}^{-1}} \leq \sqrt{\lambda}\|h - \tilde{h}\|_{H^{-1}}\,.$$

Note that the above inequality would have hold if we were doing $T = \frac{n\sqrt{m}}{\varepsilon}$ queries to compute the approximate with respect to both H^{-1} and $\tilde{H}^{-1}$. However, to compute $\tilde{h}$ and bound it with respect to $\|\cdot\|_{\tilde{H}^{-1}}$, we do $\sqrt{\lambda}T$ queries, and as such, Equation 16 effectively holds.

7.2.3 Application within Newton's step – Measure of progress

We consider we have an approximate of the gradient $\nabla\Phi$, $\tilde{\nabla}\Phi$, such that for all $\chi \in \mathbb{R}^n$, it holds that

$$\left\|\nabla\Phi(\chi) - \tilde{\nabla}\Phi(\chi)\right\|_{H^{-1}} \leq \varepsilon\,, \tag{17}$$

where H is the current (in terms of the Newton's steps) Hessian matrix. The above inequality can be directly used to bound the convergence of Newton's method. We first recall that

$$\begin{aligned} x' &= x - H^{-1}\nabla\Phi(x)\,; \\ y' &= x - \tilde{H}^{-1}\tilde{\nabla}\Phi(x)\,, \end{aligned}$$

and we define

$$\hat{y} = x - H^{-1}\tilde{\nabla}\Phi(x)\,,$$

where we only use the approximate gradient, the Hessian is the actual one. The measure of progress can be expressed as follows, using the triangular inequality

$$\|y' - x^*\|_H \leq \|y' - \hat{y}\|_H + \|\hat{y} - x'\|_H + \|x' - x^*\|_H\,,$$

where we'll bound each of the three terms. The first term the least straightforward to bound.

$$\begin{aligned} \|y' - \hat{y}\|_H &= \|\tilde{H}^{-1}\tilde{\nabla}\Phi(x) - H^{-1}\tilde{\nabla}\Phi(x)\|_H \\ &= \|H^{\frac{1}{2}}\tilde{H}^{-1}\tilde{\nabla}\Phi(x) - H^{\frac{1}{2}}H^{-1}\tilde{\nabla}\Phi(x)\|_2 \\ &= \|H^{\frac{1}{2}}\tilde{H}^{-1}H^{\frac{1}{2}}(H^{-\frac{1}{2}}\tilde{\nabla}\Phi(x)) \\ &\qquad - H^{\frac{1}{2}}H^{-1}H^{\frac{1}{2}}(H^{-\frac{1}{2}}\tilde{\nabla}\Phi(x))\|_2 \\ &= \|(H^{\frac{1}{2}}\tilde{H}^{-1}H^{\frac{1}{2}} - \mathbb{1})(H^{-\frac{1}{2}}\tilde{\nabla}\Phi(x))\|_2 \\ &\leq (\lambda - 1)\left(\|x - x^*\|_H - \varepsilon\right)\,. \end{aligned} \tag{18}$$

The first step is the straight application of Proposition D.5. The second step is a multiplication by $H^{\frac{1}{2}}H^{-\frac{1}{2}} = \mathbb{1}$ in order to factorize and simplify in step 3. The last step is obtained as follows: we use triangular inequality to separate it into a product of norms, and on the one hand we have

$$\begin{aligned} \|H^{-\frac{1}{2}}\tilde{\nabla}\Phi(x)\|_2 &\leq \|H^{-\frac{1}{2}}\nabla\Phi(x)\|_2 + \varepsilon \\ &= \|H^{-1}\nabla\Phi(x)\|_H + \varepsilon \\ &= \|x' - x\|_H + \varepsilon \\ &\leq \|x - x^*\|_H + \varepsilon \end{aligned}$$

assuming that Newton's method does not overshoot. And on the other, hand we use Proposition D.4 together with Equation 13 to obtain

$$\|H^{\frac{1}{2}}\tilde{H}^{-1}H^{\frac{1}{2}} - \mathbb{1}\|_H \leq \lambda - 1\,.$$

For the second term, from Equation 17 we have

$$\begin{aligned} \|\hat{y} - x'\|_H &= \|H^{-1}\nabla\Phi(x) - H^{-1}\tilde{\nabla}\Phi(x)\|_H \\ &= \|\nabla\Phi(x) - \tilde{\nabla}\Phi(x)\|_{H^{-1}} \\ &\leq \varepsilon \end{aligned} \tag{19}$$

The last one is bounded by the convergence rate of the exact Newton's method (see *e.g.,* [NW06, p. 43]); such that

$$\|x' - x^*\|_H \leq \frac{1}{2}\|x - x^*\|_H^2\,. \tag{20}$$

Gathering Equation 18, Equation 19, and Equation 20, we obtain the following measure of progress.

$$\begin{aligned} \|y' - x^*\|_H &\leq (\lambda - 1)\left(\|x - x^*\|_H - \varepsilon\right) + \varepsilon + \frac{1}{2}\|x - x^*\|_H \\ &\leq (\lambda - \frac{1}{2})\|x - x^*\|_H + (2 - \lambda)\varepsilon\,. \end{aligned}$$

In order to have, let's say, $\lambda - \frac{1}{2} = \frac{9}{10}$, we set $\lambda = \frac{14}{10}$. Recall that λ is the approximation factor for the sparsification of the Hessian, and from Equation 10, we wanted $1 \leq \lambda < \frac{3}{2}$, which is maintained here. In order to end up with a *clean* expression for the measure of progress, we set $\varepsilon \leftarrow \frac{3}{5}\varepsilon$, and as such, we have

$$\|y' - x^*\|_H \leq \frac{9}{10}\|x - x^*\|_H + \varepsilon\,.$$

With this measure of progress, we can prove that we converge towards the minimizer x^* when we consider the approximate procedure. More formally we show that y^* is in an $O(\varepsilon)$-ball around x^*.

7.3 Convergence of the Approximate Newton's method

See Section E for the method for finding the k-th term of a recursive sequence. We first recall that we have the following bound

$$\|y_1 - x^*\|_H \leq \frac{9}{10}\|y_0 - x^*\|_H + \varepsilon\,, \tag{21}$$

as such we can define the following recursive sequence :

$$\begin{cases} a_0 = \|y_0 - x^*\|_H \\ a_{k+1} = \frac{9}{10}a_k + \varepsilon \end{cases}, \tag{22}$$

It is easy to see that Equation 21 shows an upper bound on a_1. For the sake of simplicity, we can set $a_1 = \|y_1 - x^*\|_H$.

Claim 7.3. *With Proposition E.4, we show that*

$$a_{k+1} = (\frac{9}{10})^{k+1}a_0 + O(\varepsilon)\,.$$

proving that our procedure converges.

See proof on page 12. By reducing Proposition E.5 to our case where is f defined in Equation 22 we have $p = \frac{9}{10}$, hence, f is a contraction mapping. Theorem E.3 states that f has an unique fixed point $\bar{a}$. Since

$$\lim_{k\to\infty}(\frac{9}{10})^k = 0\,,$$

The fixed point of f is

$$\begin{aligned} \bar{a} &= \lim_{k\to\infty} a_k \\ &= \lim_{k\to\infty}\left((\frac{9}{10})^k a_0 + 10\varepsilon\left(1 - (\frac{9}{10})^k\right)\right) \\ &= 10\varepsilon \end{aligned}$$

It is important to stress that y^* such that $\tilde{\nabla}\Phi(y^*) = 0$ is never computed. Typically there is not a finite rank r such that $a_r = \bar{a}$. However, a_r can be arbitrarily close to $\bar{a}$ for a finite r. We choose r such that, for instance, $a_r \leq \frac{11}{10}\bar{a}$, which implies that for all $k > r$,

$$\|y_k - x^*\| \leq 11\varepsilon\,.$$

Thus, for all such k, Newton's steps will keep on oscillating inside the 11ε-ball around x^*. As such, as soon as we obtain a y_k that is within the 11ε-ball around x^*, it is wise to stop the procedure since doing one more step will produce a y_{k+1} for which we have no way to say whether it will be closer to x^* than y_k was. Using Equation 27, we can easily show that

$$r \leq \log_p(\varepsilon^{-1})\,,$$

with $p = \frac{9}{10}$. Consequently, the stop-condition of Algorithm 4 becomes $\|y_k - x^*\| \leq 11\varepsilon$, and we can expect that to happen after $\log_p(\varepsilon^{-1})$ Newton's steps.

References

[AdW20] Simon Apers and Ronald de Wolf. "Quantum Speedup for Graph Sparsification, Cut Approximation and Laplacian Solving". In: *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*. Durham, NC, USA: IEEE, Nov. 2020, pp. 637–648. ISBN: 978-1-72819-621-3. DOI: 10.1109/FOCS46700.2020.00065.

[CHJ22] Arjan Cornelissen, Yassine Hamoudi, and Sofiene Jerbi. "Near-optimal Quantum algorithms for multivariate mean estimation". In: *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 2022. DOI: 10.1145/3519935.3520045.

[Coh+14] Michael B. Cohen et al. *Uniform Sampling for Matrix Approximation*. Aug. 2014.

[CPT15] Tobias Christiani, Rasmus Pagh, and Mikkel Thorup. "From Independence to Expansion and Back Again". In: (2015). DOI: 10.48550/ARXIV.1506.03676.

[DG03] Sanjoy Dasgupta and Anupam Gupta. "An elementary proof of a theorem of Johnson and Lindenstrauss". en. In: *Random Structures and Algorithms* 22.1 (Jan. 2003), pp. 60–65. DOI: 10.1002/rsa.10073.

[JL84] William Johnson and Joram Lindenstrauss. "Extensions of Lipschitz maps into a Hilbert space". In: *Contemporary Mathematics* 26 (Jan. 1984), pp. 189–206. DOI: 10.1090/conm/026/737400.

[LMP13] Mu Li, Gary L. Miller, and Richard Peng. *Iterative Row Sampling*. Apr. 2013.

[NW06] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. en. Springer Series in Operations Research and Financial Engineering. Springer New York, 2006. ISBN: 9780387303031. DOI: 10.1007/978-0-387-40065-5.

[ST08] Daniel A. Spielman and Shang-Hua Teng. *Spectral Sparsification of Graphs*. 2008. DOI: 10.48550/ARXIV.0808.4134.

[Zha15] Mark Zhandry. "Secure identity-based encryption in the quantum random oracle model". en. In: *International Journal of Quantum Information* 13.04 (June 2015), p. 1550014. DOI: 10.1142/S0219749915500148.

Appendices

A Johnson-Lindenstrauss lemma

The Johnson-Lindenstrauss lemma [JL84] is a result stating that it is possible to embed with *low-distortion*, *i.e.,* by preserving the pairwise distance between any pair of points within a factor $(1\pm\varepsilon)$, points from high-dimensional into low-dimensional Euclidean space. It is appropriate to consider the distributional statement of the Johnson-Lindenstrauss lemma (DJL), in order to address it not in terms of parwise distances but with respect to the resulting distortion of the mapping.

Lemma A.1 (Distributional Johnson-Lindenstrauss lemma)**.** *For any $0 < \varepsilon < 1$, $\delta < 1/2$ and $d \in \mathbb{N}$, there exists a distribution over $\mathbb{R}^{k\times d}$ from which the matrix Π is drawn such that for $k = O\big(\varepsilon^{-2}\log(\delta^{-1})\big)$ and any vector $\chi \in \mathbb{R}^d$, the following claim holds.*

$$\mathbb{P}\Big(\left|\|\Pi\chi\|_2^2 - \|\chi\|_2^2\right| > \varepsilon \cdot \|\chi\|_2^2\Big) < \delta\,.$$

This lemma states that there exists a matrix Π that reduces the column space of the initial space, and with probability $\geq 1-\delta$ the norm of a given vector in the reduced space is a good multiplicative approximation of its norm original norm.

B k-independent hash functions

It is important to stress that the results presented in this subsection are purely classical.

Definition B.1 (k-independent hashing functions)**.** *Let $\mathcal{U}$ be the set of keys. A family $\mathcal{H} = \big\{h : \mathcal{U} \to [m]\big\}$ is said to be k-independent if for all keys $x_1, \cdots, x_k$ in $\mathcal{U}$ pairwise distinct and for all values $v_1, \cdots, v_k$ in $[m]$,*

$$\left|\{h \in \mathcal{H}\ ;\ h(x_1) = v_1, \cdots, h(x_k) = v_k\}\right| = \frac{|\mathcal{H}|}{m^k}\ ,$$

in other words, by providing $\mathcal{H}$ with the uniform probability, for any $h \in \mathcal{H}$

$$\mathbb{P}\Big(h(x_1) = v_1, \cdots, h(x_k) = v_k\Big) = \frac{1}{m^k}\ .$$

Using the result of [CPT15] on k-wise independent hash functions, rephrased by [AdW20] yields the following theorem.

Theorem B.2. *It is possible to construct in time $\tilde{O}(k)$ a data structure of size $\tilde{O}(k)$ that allows to simulate queries to a k-independent hash function in $\tilde{O}(1)$ time per query.*

C Interior Points Method

In the IPM framework, the following linear program is considered, which corresponds to the minimum cost flow problem.

$$\begin{array}{ll} \text{minimize} & c^T f \\ \text{subject to} & B^T f = d. \\ & f \geq 0 \end{array} \tag{23}$$

By using the Barrier formulation, one can put the constraints in the ojective and enable an iterative approximation of the optimal solution by studying the objective function.

$$\text{minimize} \quad \Phi_\mu(x) = -\tfrac{d^T x}{\mu} - \underbrace{\sum_{i=1}^{n} \ln(\underbrace{c - Bx}_{\text{slack}})}_{\text{log barrier}} \tag{24}$$

The log term provides a penality when the slack is close to 0 (so when the current solution is close to the edges of the polytope). The slack should always be positive, i.e. $c - Bx \geq 0$ – this is a difficult condition to maintain. The μ term should be very small for a good approximation of the optimal solution, e.g. $\mu = n^{-100}$, but in the first steps of IPM we start with a large value of μ in such a way that the only significant term in the objective function is the log barrier. We define

$$X^*_\mu = \arg\min_x \Phi_\mu|x|,$$

such that the

$$\text{central path} = \big\{X^*_\mu : \mu \in [0, \infty]\big\}\,.$$

Iterating over close values of μ is efficient: i.e. it is efficient to compute $X^*_{\mu_l}$ from $X^*_{\mu_{l-1}}$ and $X^*_{\mu_{l+1}}$ from $X^*_{\mu_l}$, each iteration requires $\sqrt{m}$ iterations with the Newton's method.

We consider the gradient

$$\nabla\Phi_\mu(X) = -\frac{d}{\mu} + \frac{B^T}{\vec{s}},$$

which, when at optimal, gives

$$B^T \frac{\mu}{\vec{s}} = d.$$

We have

$$\nabla^2\Phi_\mu(X) = B^T S^{-2} B,$$

where $S = \text{diag}(\vec{s})$. By applying the Newton's method, we obtain

$$X' = X - (\underbrace{B^T S^{-2} B}_{\text{the Hessian } H})^{-1}\nabla\Phi_\mu(x)\,.$$

The expression $H^{-1}\nabla\Phi_\mu(X)$ corresponds to a Δx that satisfies $B^T S^{-2} B\Delta x = \nabla\Phi_\mu(x)$ and is referred to as the vector potential. Thus it is not necessary to inverse H, one could solve $H\Delta x = \nabla\Phi_\mu(x)$ to find Δx.

A summary of the algorithm is given here.

Algorithm 4 Interior points method

```
Require: $B \in \mathbb{R}^{n\times m}, \varepsilon > 0, \mu \gg 1, \delta = \frac{1}{10\sqrt{m}}, x_0$ feasible solution
if $\mu = \varepsilon$ then
    exit
$k \leftarrow 0$
while $\nabla\Phi_\mu(x_k) \neq 0$ do                    ▷ Newton's method
    $x_{k+1} \leftarrow x_k - \left[B^T S^{-2} B\right]^{-1} \nabla\Phi_\mu(x_k)$
    $k \leftarrow k+1$
                                            ▷ here $x = \arg\min \Phi_\mu$
$\mu \leftarrow \frac{\mu}{1+\delta}$                ▷ Advancing down the central path
iterate once again with new values of $x_k$ and $\mu$
```

Complexity-wise, it requires to solve $\tilde{O}(\sqrt{m})$ linear systems to move from x^*_μ to $x^*_{\frac{\mu}{1+\delta}}$ (*i.e.,* finding $\arg\min \Phi_\mu$), hence a total of $\tilde{O}(\frac{\sqrt{m}}{\varepsilon})$ to get to $\mu = \varepsilon$.

D Some properties of PSD matrices

We first define positive-semidefinite (PSD) matrices, and then expose some properties that we use along the report.

Definition D.1 (PSD matrix)**.** *An $n \times n$ symmetric real matrix M is said to be positive-semidefinite if, for all nonzero vector $\chi \in \mathbb{R}^n$,*

$$\chi^T M \chi \geq 0 \,.$$

And, equivalently, M is said to be positive-semidefinite if all its eigenvalues are nonnegative.

If M is PSD, then we write $M \succcurlyeq 0$.

Proposition D.2 (Inverse of PSD matrix)**.**

$$H \succ 0 \Rightarrow H^{-1} \succ 0$$

Proof. If $H \succ 0$, then H is invertible. Let $y := Hx$, thus for all x

$$y^T H^{-1} y = x^T H^T H^{-1} H x = x^T H^T x = x^T H x > 0 \,.$$

□

Proposition D.3 (Root of PSD matrix)**.**

$$H^{-1} \succcurlyeq 0 \Rightarrow H^{-\frac{1}{2}} \succcurlyeq 0$$

Proof. Diagonalize H^{-1}, obtain $H^{-\frac{1}{2}}$ by taking the root of the eigenvalues. Since the function $f : x \mapsto \sqrt{x}$ is a positive function, all eigenvalues of $H^{-\frac{1}{2}}$ are nonnegative. □

Proposition D.4 (PSD matrix product)**.**

$$A \preccurlyeq B \Rightarrow \mathbb{1} \preccurlyeq A^{-\frac{1}{2}} B A^{-\frac{1}{2}} \,.$$

Proof. By definition of Loewner order, we say that $A \preccurlyeq B$ if and only if $0 \preccurlyeq B - A$, *i.e.,* for all vector x, it holds that

$$x^T (B - A) x \geq 0 \,. \tag{25}$$

Multiplying each element of Equation 25 by $A^{-\frac{1}{2}}$ gives

$$(A^{-\frac{1}{2}}x)^T (A^{-\frac{1}{2}} B A^{-\frac{1}{2}} - \mathbb{1})(A^{-\frac{1}{2}}x) \geq 0 \,.$$

where we can define $y := A^{-\frac{1}{2}}x$, and as such it holds that for all y,

$$y^T (A^{-\frac{1}{2}} B A^{-\frac{1}{2}} - \mathbb{1}) y \geq 0 \,,$$

that is,

$$\mathbb{1} \preccurlyeq A^{-\frac{1}{2}} B A^{-\frac{1}{2}} \,.$$

□

Proposition D.5 (Changing the norm)**.** $\forall \chi \in \mathbb{R}^n, H \succcurlyeq 0 \in \mathbb{R}^{n\times n}$,

$$\|\chi\|_H = \|H^{\frac{1}{2}}\chi\|_2$$

Proof. By definition of the weighted norm with respect to a matrix H (Equation 6), and since H is assumed PSD,

$$\begin{aligned}
\|\chi\|_H &= \sqrt{\chi^T H \chi} \\
&= \sqrt{\chi^T H^{\frac{1}{2}} H^{\frac{1}{2}} \chi} \\
&= \sqrt{\chi^T (H^{\frac{1}{2}})^T H^{\frac{1}{2}} \chi} \\
&= \sqrt{(H^{\frac{1}{2}}\chi)^T H^{\frac{1}{2}} \chi} \\
&= \|H^{\frac{1}{2}}\chi\|_2
\end{aligned}$$

□

E Recursive sequences

A recursive sequence $(a_k)_{k\in\mathbb{N}}$ is defined by as follows

$$\begin{cases} a_0 \in \mathbb{R} \\ a_{k+1} = p a_k + q = f(a_k), \quad p, q \in \mathbb{R} \end{cases} . \tag{26}$$

Definition E.1 (Fixed point of a sequence)**.** *A fixed point of a sequence is an $\bar{a}$ such that*

$$f(\bar{a}) = \bar{a} \,.$$

Definition E.2 (Contraction mapping)**.** *A function f is said to be a contraction if for all x, y and a constant $0 \leq K < 1$,*

$$\|f(x) - f(y)\| \leq K\|x - y\| \,.$$

Theorem E.3 (Banach fixed-point theorem, informal)**.** *If $f : \mathbb{R} \to \mathbb{R}$ is a contraction mapping, then f admits an unique fixed point $\bar{a}$. Furthermore,*

$$\lim_{k\to\infty} a_k = \bar{a} \,.$$

Given the definition of f, it is possible to directly evaluate a_{k+1}.

Proposition E.4. $\forall k \geq 0, \quad a_{k+1} = (a_1 - \frac{q}{1-p})p^k + \frac{q}{1-p}$.

Proof. By recurrence on k.
Base case: $(a_1 - \frac{q}{1-p})p^0 + \frac{q}{1-p} = a_1$
Inductive case: We assume

$$p_k : a_{k+1} = (a_1 - \frac{q}{1-p})p^k + \frac{q}{1-p}$$

true, and we show that

$$p_{k+1} : a_{k+2} = (a_1 - \frac{q}{1-p})p^{k+1} + \frac{q}{1-p}$$

is also true.

$$\begin{aligned}
a_{k+2} &= pa_{k+1} + q \\
&= p\left((a_1 - \frac{q}{1-p})p^k + \frac{q}{1-p}\right) + q \\
&= (a_1 - \frac{q}{1-p})p^{k+1} + \frac{pq}{1-p} + \frac{q(1-p)}{1-p} \\
&= (a_1 - \frac{q}{1-p})p^{k+1} + \frac{pq - pq + q}{1-p} \\
&= (a_1 - \frac{q}{1-p})p^{k+1} + \frac{q}{1-p}
\end{aligned}$$

Conclusion: p_k implies p_{k+1} for all k, which proves Proposition E.4. □

Proposition E.5. *f as defined in Equation 26 is a contraction mapping if* $0 \leq |p| < 1$.

Proof. For all x, y we have

$$\|f(x) - f(y)\| = \|p(x-y)\| \leq |p| \cdot \|x - y\| \, .$$

Thus, by Definition E.2 f is a contraction mapping if and only if $0 \leq |p| < 1$. □

F Proofs

Proof of Theorem 2.3. By definition of $\boldsymbol{A} \approx_\lambda \boldsymbol{B}$, we have

$$\frac{1}{\lambda}\boldsymbol{A}^T\boldsymbol{A} \preccurlyeq \boldsymbol{B}^T\boldsymbol{B} \preccurlyeq \boldsymbol{A}^T\boldsymbol{A} \, ,$$

thus

$$(\boldsymbol{A}^T\boldsymbol{A})^+ \preccurlyeq (\boldsymbol{B}^T\boldsymbol{B})^+ \preccurlyeq \lambda \cdot (\boldsymbol{A}^T\boldsymbol{A})^+ \, ,$$

which yields, using the definition of the generalized leverage scores and by the positive semi-definiteness of each of the matrices,

$$\tau_i(\boldsymbol{A}) \leq \tau_i^{\boldsymbol{B}}(\boldsymbol{A}) \leq \lambda \cdot \tau_i(\boldsymbol{A}) \, .$$

□

Proof of Proposition 3.1. Let S be the total number of queries.

It can be expressed as a sum of all expected number of queries (we omit the big O notation on purpose):

$$S = \sum_{i=0}^{k-1} \sqrt{\frac{n}{k-i}} = \sum_{i=1}^{k} \sqrt{\frac{n}{i}} = \sqrt{n} \cdot \sum_{i=1}^{k} i^{-\frac{1}{2}} \, .$$

Let $g(x) = x^{-\frac{1}{2}}$, since $g'(x) < 0$ for all $x > 0$, g is a strictly decreasing function over $\mathbb{R}_*^+$. In addition, a formula to bound a sum of a stricly decreasing function by integrals states that

$$\int_a^{b+1} g(s)ds \leq \sum_{i=a}^{b} g(i) \leq \int_{a-1}^{b} g(s)ds \, ,$$

which yields as long as $a = 1$ and $b = k$

$$\begin{aligned}
\int_1^{k+1} g(s)ds \quad &\leq \quad \sum_{i=1}^{k} g(i) \quad \leq \int_0^k g(s)ds \\
\Leftrightarrow \quad \sqrt{n}\left[2\sqrt{s}\right]_1^{k+1} \quad &\leq \sqrt{n}\sum_{i=1}^{k} i^{-\frac{1}{2}} \leq \sqrt{n}\left[2\sqrt{s}\right]_0^k \\
\Leftrightarrow 2\sqrt{n}(\sqrt{k+1} - 1) \leq &\qquad S \qquad \leq \quad 2\sqrt{nk} \, .
\end{aligned}$$

Thus, k calls to QueryAndUpdate find k distinct marked elements in $O(\sqrt{nk})$ quantum queries.

Note that herin we do not consider the time to construct f at each iteration. It is well when considering only the query complexity. Such implementation requires a circuit of $O(\log n)$ qubits, and the associated quantum circuit runs in time $O(h(n))$ as long as classicaly f runs in time $O(h(n))$. □

Proof of Claim 5.2. A convient definition to consider is the one of the pseudoinverse of a vector.

Definition F.1 (Pseudoinverse of a vector)**.** *Let x be a vector, the pseudoinverse of x is*

$$x^+ = \begin{cases} \boldsymbol{0}^T & , \textit{if } x = \boldsymbol{0}\, ; \\ \frac{x^*}{x^*x} & , \textit{otherwise.} \end{cases}$$

For the uniform sampling step, we consider $\boldsymbol{S}$ a sampling matrix with $s_{ii} = 1$ *w.p.* $\frac{m}{n}$ and 0 otherwise. If $s_{ii} = 0$, then $(sa_i)^T$, the i-th row of $\boldsymbol{SA}$ is $\boldsymbol{0}^T$, thus $(sa_i)^{T+}$ is $\boldsymbol{0}$: we can simply *remove* this row since it does not infere in the calculation of $\left((\boldsymbol{SA})^T\boldsymbol{SA}\right)^+$ according to Definition F.1. Removing all the rows of $\boldsymbol{A}$ where $s_{ii} = 0$ yields exactly $\boldsymbol{A'}$. □

Proof of Proposition 5.5. Considering Equation 2 yields

$$\begin{aligned}
\|\boldsymbol{\chi}_i\|_2^2 &= a_i^T\left((\boldsymbol{A}^T\boldsymbol{A})^+\right)^T\boldsymbol{A}^T\boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^+a_i \\
&= a_i^T\left((\boldsymbol{A}^T\boldsymbol{A})^T\right)^+\boldsymbol{A}^T\boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^+a_i \\
&= a_i^T(\boldsymbol{A}^T\boldsymbol{A})^+\boldsymbol{A}^T\boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^+a_i \\
&= a_i^T(\boldsymbol{A}^T\boldsymbol{A})^+a_i \\
&= \tau_i(\boldsymbol{A})
\end{aligned}$$

The second step comes from the commutation of the pseudoinversion with transposition[5], the third step follows from the symmetry of $\boldsymbol{A}^T\boldsymbol{A}$, and the fourth step is the application of the weak inverse property of pseudoinverses[6]. □

Proof of Proposition 5.7. First, in order to have a more suitable expression of Equation 3, it is convenient to break down the absolute value and examine both cases.

- Case $\hat{\tau}_i - \tau_i \leq 0$, then *w.h.p.*

$$\begin{array}{rrcl} & \hat{\tau}_i - \tau_i & \geq & -\varepsilon \cdot \tau_i \\ \Leftrightarrow & \hat{\tau}_i & \geq & -\varepsilon \cdot \tau_i + \tau_i \\ \Leftrightarrow & \hat{\tau}_i & \geq & (1-\varepsilon) \cdot \tau_i \end{array}$$

- Case $\hat{\tau}_i - \tau_i \geq 0$, then *w.h.p.*

$$\begin{array}{rrcl} & \hat{\tau}_i - \tau_i & \leq & \varepsilon \cdot \tau_i \\ \Leftrightarrow & \hat{\tau}_i & \leq & \varepsilon \cdot \tau_i + \tau_i \\ \Leftrightarrow & \hat{\tau}_i & \leq & (1+\varepsilon) \cdot \tau_i \end{array}$$

Thus, still with probability $\geq 1 - \frac{1}{n}$, Equation 3 can be rephrased as follows,

$$(1-\varepsilon) \cdot \tau_i \;\leq\; \hat{\tau}_i \;\leq\; (1+\varepsilon) \cdot \tau_i \,, \quad \forall i \in [n] \,.$$

However, in order to obtain at the end a ε-spectral approximation when sampling according to $\hat{\tau}_i$, it must hold that these approximations are *upper bounds* on the original ones. Note that here, what we denoted by the *original scores* are actually approximate of the *true scores*, *i.e.,* the $\tilde{\tau}_i$. Thus, it suffices to sample according to $\frac{1}{1-\varepsilon}\hat{\tau}_i$, since for all i, it holds that

$$\tilde{\tau}_i \;\leq\; \frac{1}{1-\varepsilon}\hat{\tau}_i \;\leq\; (\frac{1+\varepsilon}{1-\varepsilon})\tilde{\tau}_i \,,$$

which, by the way, implies that

$$\mathbb{E}\left[\sum_{i=1}^{n} \tilde{\tau}_i\right] \;\leq\; \mathbb{E}\left[\sum_{i=1}^{n} \frac{1}{1-\varepsilon}\hat{\tau}_i\right] \;\leq\; \mathbb{E}\left[\sum_{i=1}^{n} \frac{1+\varepsilon}{1-\varepsilon} \cdot \tilde{\tau}_i\right] \,,$$

and equivalently, since the expectation of the sum of the approximate leverage scores is bounded thanks to Theorem 5.3, it holds that

$$\frac{nd}{m} \;\leq\; \mathbb{E}\left[\sum_{i=1}^{n} \frac{1}{1-\varepsilon}\hat{\tau}_i\right] \;\leq\; (\frac{1+\varepsilon}{1-\varepsilon})\frac{nd}{m} \,.$$

Hence, it is possible to apply Johnson-Lindenstrauss transfrom to each $\boldsymbol{B}'_l$ to get a matrix of dimension $\tilde{O}(\varepsilon^{-2}) \times d$, and thus computing a single leverage score in time $\tilde{O}(\varepsilon^{-2}S)$.

□

Proof of Claim 6.1. By simple probabilistic argument. Let $1 \leq l \leq L$. For any $1 \leq k \leq l$, it holds by Equation 4 that the i-th bit of Z_k is 1 with probability half for all i in $[n]$, which implies that

$$(Z'_l)_i = \wedge_{k=1}^{l}(Z_k)_i = 1 \;\; w.p. \;\; \frac{1}{2^l} \,.$$

Thus, the expected number of ones in Z'_l is $\frac{n}{2^l}$, which is exactly the expected number of rows of $\boldsymbol{A}_l$. □

Proof of Claim 7.2. In order to prove Claim 7.2, we'll use the properties of positive-(semi)definite matrices shown in Section D to show that $\frac{1}{m^2}H^{-\frac{1}{2}}hh^TH^{-\frac{1}{2}} \succcurlyeq 0$, where $H = AA^T$, $A = B^TS^{-1}$ and $h = B^TS^{-1}\,|\mathbf{1}\rangle$.

Since $H = AA^T$, $H \succcurlyeq 0$[7] thus by Proposition D.2 $H^{-1} \succ 0$ and by Proposition D.3 $H^{-\frac{1}{2}} \succ 0$ Hence,

$$\begin{aligned} H^{-\frac{1}{2}}hh^TH^{-\frac{1}{2}} &= (H^{-\frac{1}{2}}h)(h^TH^{-\frac{1}{2}}) \\ &= (H^{-\frac{1}{2}}h)(H^{-\frac{1}{2}}h)^T \\ &\succcurlyeq 0 \end{aligned}$$

which implies,

$$\Sigma_Y \preccurlyeq \frac{1}{m}\mathbb{1} - \frac{1}{m^2}H^{-\frac{1}{2}}hh^TH^{-\frac{1}{2}} \preccurlyeq \frac{1}{m}\mathbb{1}$$

□

Proof of Claim 7.3. Here $p = \frac{9}{10}$ and $q = \varepsilon$.

$$\begin{aligned} a_{k+1} &= (a_1 - \frac{q}{1-p})p^k + \frac{q}{1-p} \\ &= (a_1 - 10\varepsilon)(\frac{9}{10})^k + 10\varepsilon \\ &= (\frac{9}{10})^k a_1 - (\frac{9}{10})^k 10\varepsilon + 10\varepsilon \\ &\leq (\frac{9}{10})^k(\frac{9}{10}a_0 + \varepsilon) - (\frac{9}{10})^k 10\varepsilon + 10\varepsilon \\ &= (\frac{9}{10})^k\frac{9}{10}a_0 + (\frac{9}{10})^k\varepsilon - (\frac{9}{10})^k 10\varepsilon + 10\varepsilon \\ &= (\frac{9}{10})^{k+1}a_0 + 10\varepsilon\left(\frac{1}{10}(\frac{9}{10})^k - (\frac{9}{10})^k + 1\right) \\ &= (\frac{9}{10})^{k+1}a_0 + 10\varepsilon\left((\frac{9}{10})^k(\frac{1}{10} - 1) + 1\right) \\ &= (\frac{9}{10})^{k+1}a_0 + 10\varepsilon\left(-(\frac{9}{10})^k\frac{9}{10} + 1\right) \\ &= (\frac{9}{10})^{k+1}a_0 + 10\varepsilon\left(1 - (\frac{9}{10})^{k+1}\right) \end{aligned} \tag{27}$$

□

[5] $(\boldsymbol{A}^+)^T = (\boldsymbol{A}^T)^+$

[6] $\boldsymbol{A}^+\boldsymbol{A}\boldsymbol{A}^+ = \boldsymbol{A}^+$

[7] We can assume $H \succ 0$ since – in the context of IPM – H is nonsingular.