Approximating Matrix p-norms

Aditya Bhaskara * Aravindan Vijayaraghavan †

Abstract

We consider the problem of computing the $q \mapsto p$ norm of a matrix A, which is defined for $p, q \geq 1$, as

$$\|A\|_{q \mapsto p} = \max_{x \neq \vec{0}} \frac{\|Ax\|_p}{\|x\|_q}.$$

This is in general a non-convex optimization problem, and is a natural generalization of the well-studied question of computing singular values (this corresponds to $p = q = 2$). Different settings of parameters give rise to a variety of known interesting problems (such as the Grothendieck problem when $p = 1$ and $q = \infty$). However, very little is understood about the approximability of the problem for different values of p, q.

Our first result is an efficient algorithm for computing the $q \mapsto p$ norm of matrices with non-negative entries, when $q \geq p \geq 1$. The algorithm we analyze is based on a natural fixed point iteration, which can be seen as an analog of power iteration for computing eigenvalues.

We then present an application of our techniques to the problem of constructing a scheme for oblivious routing in the ℓ_p norm. This makes constructive a recent existential result of Englert and Räcke [ER09] on $O(\log n)$ competitive oblivious routing schemes (which they make constructive only for $p = 2$).

On the other hand, when we do not have any restrictions on the entries (such as non-negativity), we prove that the problem is NP-hard to approximate to any constant factor, for $2 < p \leq q$ and $p \leq q < 2$ (these are precisely the ranges of p, q with $p \leq q$ where constant factor approximations are not known). In this range, our techniques also show that if $\mathsf{NP} \notin \mathsf{DTIME}(n^{\mathrm{polylog}(n)})$, the problem cannot be approximated to a factor $2^{(\log n)^{1-\varepsilon}}$, for any constant $\varepsilon > 0$.

*Center for Computational Intractability, and Department of Computer Science, Princeton University. Supported by NSF CCF 0832797. Email: bhaskara@cs.princeton.edu

†Center for Computational Intractability, and Department of Computer Science, Princeton University. Supported by NSF CCF 0832797. Email: aravindv@cs.princeton.edu

1 Introduction

We study the problem of computing norms of matrices. The ℓ_q to ℓ_p norm of a matrix $A \in \mathbb{R}^{m \times n}$ is defined to be

$$\|A\|_{q \mapsto p} = \max_{x \in \mathbb{R}^n, x \neq \vec{0}} \frac{\|Ax\|_p}{\|x\|_q}, \qquad \text{where } \|x\|_p = (|x_1|^p + \cdots + |x_n|^p)^{1/p}$$

Throughout, we think of $p, q \geq 1$. If we think of the matrix as an operator from $\mathbb{R}^n$ with the ℓ_q norm to the space $\mathbb{R}^m$ with ℓ_p norm, the norm $\|A\|_{q \mapsto p}$ measures the 'maximum stretch' of a unit vector.

Computing the $q \mapsto p$-norm of a matrix is a natural optimization problem. For instance, it can be seen as a natural generalization of the extensively studied problem of computing the largest singular value of a matrix [HJ85]. This corresponds to the case $p = q = 2$. When $p = 1$ and $q = \infty$, it turns out to be the well-studied Grothendieck problem [Gro53, AN04], which is defined as

$$\max_{x_i, y_i \in \{-1,1\}} \sum_{i,j} a_{ij} x_i y_j.$$

Thus for different settings of the parameters, the problem seems to have very different flavors.

We study the question of approximating $\|A\|_{q \mapsto p}$ for different ranges of the parameters p, q. The case $p = q$ is referred to as the matrix p-norm (denoted by $\|A\|_p$), and has been considered in the scientific computing community. For instance, it is known to have connections with matrix condition number estimates (see [Hig92] for other applications). Computing $\|A\|_{q \mapsto p}$ has also been studied because of its connections to robust optimization [Ste05]. Another special case which has been studied [Boy74, Ste05] is one where the entries of the matrix A are restricted to be non-negative. Such instances come up in graph theoretic problems, like in the ℓ_p oblivious routing question of [ER09].

Note that computing the matrix $q \mapsto p$ norm is a problem of maximizing a convex function over a convex domain. While a convex function can be minimized efficiently over convex domains using gradient descent based algorithms, it is in general hard to maximize them. Thus it is interesting that our algorithm can efficiently compute the norm for non-negative matrices for a range of parameters.

Known algorithms. Very little is known about approximating $\|A\|_{q \mapsto p}$ in general. For computing p-norms (i.e., $q = p$), polynomial time algorithms for arbitrary A are known to exist only for $p = 1, 2$, and ∞. For the general problem, for $p \leq 2$, $q > 2$, Nesterov[Nes98] shows that the problem can be approximated to a constant factor (which can be shown to be < 2.3), using a semidefinite programming relaxation. When the matrix has only non-negative entries, this relaxation can be shown to be exact [Ste05].

For other ranges of p, q, the best known bounds are polynomial factor approximations, obtained by 'interpolating'. For instance, for computing $\|A\|_{p \mapsto p}$, computing the vectors that maximize the norm for $p = 1, 2, \infty$, and picking the best of them gives an $O(n^{1/4})$ approximation for all p (see [Hig92]). For the general problem of computing $\|A\|_{q \mapsto p}$, Steinberg [Ste05] gives an algorithm with an improved guarantee of $O(n^{25/128})$, by taking into account the approximation algorithms of Nesterov for certain ranges.

These algorithms use Hölder's inequality, and a fact which follows from the duality of ℓ_p spaces (this sometimes allows one to 'move' from one range of parameters to another):

$$\|A\|_{q \mapsto p} = \|A^T\|_{p' \mapsto q'},$$

where A^T is the transpose of the matrix A, and p' and q' are the 'duals' of p, q respectively (i.e. $1/p + 1/p' = 1$). See Appendix A.2 for a proof.

The hardness front. The problem is known to be NP-hard in the range $q \geq p \geq 1$ [Ste05]. Very recently in independent work, [HO09] show that it is NP-hard to compute the p-norm to arbitrary relative precision when $p \notin \{1, 2, \infty\}$ (i.e., there cannot be a $(1+\delta)$ approximation algorithm with run time $\text{poly}(n, m, 1/\delta)$).

1.1 Our Results

Non-negative matrices. We first consider the case of matrices A with non-negative entries. Here we prove that if $1 \leq p \leq q$, then $\|A\|_{q \mapsto p}$ can be computed in polynomial time. More precisely we give an algorithm which gives a $(1+\delta)$ approximation in time polynomial in n, m, and $(1/\delta)$.

Thus in particular, we give the first poly time guarantee (to the best of our knowledge) for computing the matrix p-norm for non-negative matrices. We give an analysis of a power iteration type algorithm for computing p-norms proposed by Boyd [Boy74]. The algorithm performs a fixed point computation, which turns out to mimic power iteration for eigenvalue computations.

Heuristic approaches to many optimization problems involve finding solutions via fixed point computations. Our analysis proves polynomial convergence time for one such natural fixed point algorithm. These techniques could potentially be useful in other similar settings. We believe that this algorithm could be useful as an optimization tool for other problems with objectives that involve p-norms (or as a natural extension of eigenvalue computations). We now mention one such application, to oblivious routing in the ℓ_p norm.

Application to Oblivious Routing. In the oblivious routing problem, we are given a graph G, and we need to output a 'routing scheme', namely a unit flow between every pair of vertices. Now given a set of demands (for a multicommodity flow), we can route them according to this scheme (by scaling the flows we output according to the demand in a natural way), and the total flow on each edge is obtained. The aim is to compete (in terms of, for instance, max congestion) with the best multicommodity flow 'in hindsight' (knowing the set of demands).

For max-congestion (maximum total flow on an edge – which is the ℓ_∞ norm of the vector of flows on edges), a beautiful result of [RÖ8] gives an $O(\log n)$ competitive routing scheme. Englert and Räcke [ER09] recently showed that there exists an oblivious routing scheme which attains a competitive ratio of $O(\log n)$ when the objective function is the ℓ_p-norm of the flow vector ($|E|$ dimensional vector). However, they can efficiently compute this oblivious routing scheme only for $p = 2$.

From the analysis of our algorithm, we can prove that for matrices with strictly positive entries there is a unique optimum. Using this and a related idea (Section 4), we can make the result of [ER09] constructive. Here matrix p-norm computation is used as a 'separation oracle' in a multiplicative weights style update, and this gives an $O(\log n)$-competitive oblivious routing scheme for all ℓ_p-norms ($p \geq 1$).

Hardness of approximation. For general matrices (with negative entries allowed), we show the inapproximability of *almost polynomial factor* for computing the $q \mapsto p$ norm of general matrices when $q \geq p$ and both p, q are > 2. By duality, this implies the same hardness when both p, q are < 2 and $q \geq p$.[1]

More precisely, for these ranges, we prove that computing $\|A\|_{q \mapsto p}$ upto any constant factor is NP-hard. Under the stronger assumption that $\mathsf{NP} \notin \mathsf{DTIME}(2^{\text{polylog}(n)})$, we prove that the problem is hard to approximate to a factor of $\Omega(2^{(\log n)^{1-\varepsilon}})$, for any constant $\varepsilon > 0$.

[1] When $p \leq 2$ and $q \geq 2$, Nesterov's algorithm gives a constant factor approximation.

Techniques. We first consider $p \mapsto p$ norm approximation, for which we show constant factor hardness by a gadget reduction from the gap version of MaxCut. Then we show that the $p \mapsto p$ norm multiplies upon tensoring, and thus we get the desired hardness amplification. While the proof of the small constant hardness carries over to the $q \mapsto p$ norm case with $q > p > 2$, in general these norms do not multiply under tensoring. We handle this by giving a way of starting with a hard instance of $p \mapsto p$ norm computation (with additional structure, as will be important), and convert it to one of $q \mapsto p$ norm computation.

We find the hardness results for computing the $q \mapsto p$ norm interesting because the bounds are very similar to hardness of combinatorial problems like label cover, and it applies to a natural numeric optimization problem.

The question of computing $\|A\|_{\infty \mapsto p}$ has a simple alternate formulation (Definition 6.9): given vectors $\mathbf{a_1}, \mathbf{a_2}, \ldots, \mathbf{a_n}$, find a $\{\pm\}$ combination of the vectors so as to maximize the length (in ℓ_p norm) of the resultant. The previous hardness result also extends to this case.

Comparison with previous work. For clarity, let us now tabulate our algorithmic and hardness results in Tables 1.1 and show how they compare with known results for different values of the parameters p, q. Each row and column in the table gives three things: the best known algorithm in general, the best known approximation algorithm when all entries of A are non-negative, and the best known hardness for this range of p, q. An entry saying "NP-hard" means that only exact polynomial time algorithms are ruled out.

Discussion and open questions. All our algorithms and hardness results apply to the case $p \leq q$, but we do not know either of these (even for non-negative matrices) for $p > q$ (which is rather surprising!). For algorithmic results (for positive matrices, say) the fact that we can optimize $\|Ax\|_p / \|x\|_q$ seems closely tied to the fact that the set $\{x \ : \ \|x\|_p / \|x\|_q > \tau\}$ is convex for any $\tau > 0$ and $p \leq q$. However, we are not aware of any formal connection. Besides, when $p > q$, even for non-negative matrices there could be multiple optima (we prove uniqueness of optimum when $p \leq q$).

On the hardness front, the $q < p$ case seems more related to questions like the Densest k-subgraph problem (informally, when the matrix is positive and $p < q$, if there is a 'dense enough' submatrix, the optimum vector would have most of its support corresponding to this). Thus the difficulties in proving hardness for the norm question may be related to proving hardness for densest subgraph.

Hypercontractive norms (corresponding to $q < p$) have been well-studied [KKL88], and have also found prior use in inapproximability results for problems like maxcut. Also, known integrality gap instances for unique games [KV05] are graphs that are hypercontractive. We believe that computability of hypercontractive norms of a matrix could reveal insights into the approximability of problems like small set expansion [RS10] and the planted dense k-subgraph problem [BCC$^+$10].

1.2 Related work.

A question that is very related to matrix norm computation is the L_p Grothendieck problem, which has been studied earlier by [KNS08]. The problem is to compute

$$\max_{||\mathbf{x}||_p \leq 1} \mathbf{x}^t B \mathbf{x}$$

The question of computing $\|A\|_{p \mapsto 2}$ is a special case of the L_p Grothendieck problem (where $B \succeq 0$). [KNS08] give an optimal (assuming UGC) $O(p)$ approximation algorithm. For B being p.s.d.,

Table 1: **Previous work**

		$1 < q < 2$	$q = 2$	$q > 2$
Best Approximation Hardness Non-negative matrices	$1 < p < 2$	poly(n) $p < q$: NP-hard	$O(1)$ [Nes98] NP-hard Exact [Ste05]	$O(1)$[Nes98] NP-hard Exact [Ste05]
Best Approximation Hardness Non-negative matrices	$p = 2$	poly(n)	Exact Exact	$O(1)$ [Nes98] NP-hard Exact [Ste05]
Best Approximation Hardness Non-negative matrices	$p > 2$	poly(n)	poly(n)	poly(n) $p < q$: NP-hard

Table 2: **Our results.** We give better algorithms for non-negative matrices and obtain almost-polynomial hardness results when $q \geq p$.

		$1 < q < 2$	$q = 2$	$q > 2$
Hardness Non-negative matrices	$1 < p < 2$	$p \leq q$: $2^{(\log n)^{1-\epsilon}}$-hard $p \leq q$: Exact	Exact	Exact
Hardness Non-negative matrices	$p = 2$		Exact	Exact
Hardness Non-negative matrices	$p > 2$			$p \leq q$: $2^{(\log n)^{1-\epsilon}}$-hard $p \leq q$: Exact

constant factor approximation algorithms are known, due to [Nes98]. Computing $\|A\|_{\infty \mapsto 2}$ reduces to maximizing a quadratic form over ± 1 domain for p.s.d matrices [CW04, Nes98].

Recently, [DVTV09] studies an optimization problem which has an ℓ_p norm objective – they wish to find the best k-dimensional subspace approximation to a set of points, where one wishes to minimize the ℓ_p distances to the subspace (there are other problems in approximation theory which are of similar nature). When $k = n - 1$ this can be shown to reduce to the L_p Grothendieck problem for the matrix A^{-1}.

1.3 Paper Outline

We start by presenting the algorithm for positive matrices (Section 3.1), and prove poly time convergence (Section 3.2). Some additional properties of the optimization problem are discussed in Section 4 (such as unique maximum, concavity around optimum), which will be useful for an oblivious routing application. This will be presented in Section 5. Finally in Section 6, we study the inapproximability of the problem: we first show a constant factor hardness for $\|A\|_{p \mapsto p}$ (Section 6.1), and show how to amplify it (Section 6.2). Then we use this to show hardness for $\|A\|_{q \mapsto p}$ in section 6.3.

2 Notation and Simplifications

We write $\mathbb{R}_+$ for the set of non-negative reals. For a matrix A, we let A_i denote the ith row of A. Also a_{ij} denotes the element in the ith row and jth column. Similarly for a vector $\mathbf{x}$, we denote the ith co-ordinate by x_i. We say that a vector $\mathbf{x}$ is *positive* if the entries x_i are all > 0. Finally, for two vectors $\mathbf{x}, \mathbf{y}$, we write $\mathbf{x} \propto \mathbf{y}$ to mean that $\mathbf{x}$ is *proportional to* $\mathbf{y}$, i.e., $\mathbf{x} = \lambda \mathbf{y}$ for some λ (in all places we use it, λ will be > 0).

For our algorithmic results, it will be much more convenient to work with matrices where we restrict the entries to be in $[1/N, 1]$, for some parameter N (zero entries can cause minor problems). If we are interested in a $(1+\delta)$ approximation, we can first scale A such that the largest entry is 1, pick $N \approx (m+n)^2/\delta$, where m, n are the dimensions of the matrix, and work with the matrix $A + \frac{1}{N}J$ (here J is the $m \times n$ matrix of ones). The justification for this can be found in Appendix A.3. We will refer to such A as a *positive matrix*.

3 An Iterative Algorithm

In this section, we consider positive matrices A, and prove that if $1 < p \leq q$, we can efficiently compute $\|A\|_{q \mapsto p}$. Suppose A is of dimensions $n \times n$, and define $f : \mathbb{R}^n \mapsto \mathbb{R}$ by

$$f(\mathbf{x}) = \frac{\|A\mathbf{x}\|_p}{\|\mathbf{x}\|_q} = \frac{(\sum_i |A_i \mathbf{x}|^p)^{1/p}}{(\sum_i |x_i|^q)^{1/q}}.$$

We present an algorithm due to Boyd [Boy74], and prove that it converges quickly to the optimum vector. The idea is to consider ∇f, and rewrite the equation $\nabla f = 0$ as a fixed point equation (i.e., as $S\mathbf{x} = \mathbf{x}$, for an appropriate operator S). The iterative algorithm then starts with some vector $\mathbf{x}$, and applies S repeatedly. Note that in the case $p = 2$, this mimics the familiar power iteration (in this case S will turn out to be multiplication by the matrix A (up to normalization)).

3.1 Algorithm description

Let us start by looking at ∇f.

$$\frac{\partial f}{\partial x_i} = \frac{\|x\|_q \|Ax\|_p^{1-p} \cdot \sum_j a_{ij} |A_j x|^{p-1} - \|Ax\|_p \|x\|_q^{1-q} \cdot |x_i|^{q-1}}{\|x\|_q^2} \tag{1}$$

At a critical point, $\frac{\partial f}{\partial x_i} = 0$ for all i. Thus for all i,

$$|x_i|^{q-1} = \frac{\|x\|_q^q}{\|Ax\|_p^p} \cdot \sum_j a_{ij} |A_j x|^{p-1} \tag{2}$$

Define an operator $S : \mathbb{R}_+^n \to \mathbb{R}_+^n$, with the ith co-ordinate of Sx being (note that all terms involved are positive)

$$(Sx)_i = \big(\sum_j a_{ij} (A_j x)^{p-1} \big)^{1/(q-1)}$$

Thus, at a critical point, $Sx \propto x$. Now consider the the following algorithm:
(**Input.** An $n \times n$ matrix A with all entries in $[\frac{1}{N}, 1]$, error parameter δ.)

1: Initialize $x = \frac{\mathbf{1}}{\|\mathbf{1}\|_p}$.
2: **loop** $\{T$ times (it will turn out $T = (Nn) \cdot \text{polylog}(N, n, 1/\delta))\}$

3: set $x \leftarrow Sx$.
4: normalize x to make $\|x\|_q = 1$.
5: **end loop**

A *fixed point* of the iteration is a vector x such that $Sx \propto x$. Thus every critical point of f is a fixed point. It turns out that every *positive* fixed point is also a critical point. Further, there will be a unique positive fixed point, which is also the unique maximum of f.

3.2 Analyzing the Algorithm

We will treat $f(\mathbf{x})$ as defined over the domain $\mathbb{R}^n_+$. Since the matrix A is positive, the maximum must be attained in $\mathbb{R}^n_+$. Since f is invariant under scaling x, we restrict our attention to points in $\mathcal{S}^n_q = \{x \ : \ x \in \mathbb{R}^n_+, \|x\|_q = 1\}$. Thus the algorithm starts with a point in $\mathcal{S}^n_q$, and in each iteration moves to another point, until it converges.

First, we prove that the maximum of f over $\mathbb{R}^n_+$ occurs at an interior point (i.e., none of the co-ordinates are zero). Let x^* denote a point at which maximum is attained, i.e., $f(x^*) = \|A\|_{q\mapsto p}$ (x^* need not be unique). Since it is an interior point, $\nabla f = 0$ at x^*, and so x^* is a fixed point for the iteration.

Lemma 3.1. *Let $x^* \in \mathcal{S}^n_q$ be a point at which f attains maximum. Then each co-ordinate of x^* is at least $\frac{1}{(Nn)^2}$.*

The proof of this can be found in Section 4. Next, we show that with each iteration, the value of the function cannot decrease. This was proved by [Boy74] (we refer to their paper for the proof).

Lemma 3.2. *([Boy74]) For any vector x, we have*

$$\frac{\|ASx\|_p}{\|Sx\|_q} \geq \frac{\|Ax\|_p}{\|x\|_q}$$

The analysis of the algorithm proceeds by maintaining two *potentials*, defined by

$$m(x) = \min_i \frac{(Sx)_i}{x_i} \qquad \text{and} \qquad M(x) = \max_i \frac{(Sx)_i}{x_i}.$$

If x is a fixed point, then $m(x) = M(x)$. Also, from Section 3.1, each is equal to $\left(\frac{\|x\|_q^q}{\|Ax\|_p^p}\right)^{1/(q-1)}$. As observed in [Boy74], these quantites can be used to 'sandwich' the norm – in particular,

Lemma 3.3. *For any positive vector x with $\|x\|_q = 1$, we have*

$$m(x)^{q-1} \leq \|A\|^p_{q\mapsto p} \leq M(x)^{q-1}$$

The lemma is crucial – it relates the norm (which we wish to compute) to certain quantities we can compute starting with any positive vector x. We now give a proof of this lemma. Our proof, however, has the additional advantage that it immediately implies the following:

Lemma 3.4. *The maximum of f on $\mathcal{S}^n_q$ is attained at a unique point x^*. Further, this x^* is the unique critical point of f on $\mathcal{S}^n_q$ (which also means it is the unique fixed point for the iteration).*

Proof (of Lemma 3.3). Let $x \in \mathcal{S}^n_q$ be a positive vector. Let $x^* \in \mathcal{S}^n_q$ be a vector which maximizes $f(x)$.

The first inequality is a simple averaging argument:

$$\frac{\sum_i x_i \cdot (Sx)_i^{q-1}}{\sum_i x_i \cdot x_i^{q-1}} = \frac{\sum_i x_i \cdot \sum_j a_{ij}(A_j x)^{p-1}}{\sum_i x_i^q} \tag{3}$$

$$= \frac{\sum_j (A_j x)^{p-1} \sum_i a_{ij} x_i}{\sum_i x_i^q} \tag{4}$$

$$= \frac{\sum_j (A_j x)^p}{\sum_i x_i^q} = \frac{\|Ax\|_p^p}{\|x\|_q^q} \leq \|A\|_{q \mapsto p}^p \tag{5}$$

The last inequality uses $\|x\|_q = 1$. Thus there exists an index i such that $(Sx)_i^{q-1}/x_i^{q-1} \leq \|Ax\|_{q \mapsto p}$.

The latter inequality is more tricky – it gives an upper bound on $f(x^*)$, no matter which $x \in \mathcal{S}_q^n$ we start with. To prove this, we start by observing that x^* is a fixed point, and thus for all k,

$$m(x^*)^{q-1} = M(x^*)^{q-1} = \frac{(Sx^*)_k^{q-1}}{(x^*)_k^{q-1}}.$$

Call this quantity λ. Now, let $\theta > 0$ be the smallest real number such that $x - \theta x^*$ has a zero co-ordinate, i.e., $x_k = \theta x_k^*$, and $x_j \geq \theta x_j^*$ for $j \neq k$. Since $\|x\|_q = \|x^*\|_q$ and $x \neq x^*$, θ is well-defined, and $x_j > \theta x_j^*$ (strictly) for some index j. Because of these, and since each a_{ij} is strictly positive, we have $Sx > S(\theta x^*) = \theta^{(p-1)/(q-1)} S(x^*)$ (clear from the definition of S).

Now, for the index k, we have

$$\frac{(Sx)_k^{q-1}}{x_k^{q-1}} > \frac{\theta^{p-1}(Sx^*)_k^{q-1}}{(\theta x_k^*)^{q-1}} = \theta^{(p-q)} \cdot \lambda$$

Thus we have $M(x)^{q-1} > \lambda$ (since $q \geq p$, and $0 < \theta < 1$), which is what we wanted to prove. □

Let us see how this implies Lemma 3.4.

Proof (of Lemma 3.4). Let $x^* \in \mathcal{S}_q^n$ denote a vector which maximizes f over $\mathcal{S}_q^n$ (thus x^* is one fixed point of S). Suppose, if possible, that y is another fixed point. By the calculation in Eq.(5) (and since y is a fixed point and x^* maximizes f), we have

$$M(y)^{q-1} = m(y)^{q-1} = \frac{\|Ay\|_p^p}{\|y\|_q^q} = f(y)^q \leq f(x^*)^p$$

Now since $y \neq x^*$, the argument above (of considering the smallest θ such that $y - \theta x^*$ has a zero co-ordinate, and so on) will imply that $M(y)^{q-1} > \lambda = f(x^*)^p$, which is a contradiction.

This proves that there is no other fixed point. □

The next few lemmas say that as the algorithm proceeds, the value of $m(x)$ increases, while $M(x)$ decreases. Further, it turns out we can quantify how much they change: if we start with an x such that $M(x)/m(x)$ is 'large', the ratio drops significantly in one iteration.

Lemma 3.5. *Let x be a positive vector. Then $m(x) \leq m(Sx)$, and $M(x) \geq M(Sx)$.*

Proof. Suppose $m(x) = \lambda$. So for every i, we have $(Sx)_i \geq \lambda x_i$. Now fix some index i and consider the quantity

$$(SSx)_i^{q-1} = \sum_j a_{ij}(A_j Sx)^{q-1}.$$

Since A is a positive matrix and $(Sx)_i \geq \lambda x_i$, we must have $(A_j Sx) \geq \lambda \cdot (A_j x)$ for every j. Thus

$$(SSx)_i^{q-1} \geq \lambda^{q-1} \sum_j a_{ij} (A_j x)^{q-1} = \lambda^{q-1} (Sx)_i^{q-1}.$$

This shows that $m(Sx) \geq \lambda$. A similar argument shows that $M(Sx) \leq M(x)$. □

Lemma 3.6. *Let* $\mathbf{x}$ *be a positive vector with* $\|\mathbf{x}\|_q = 1$*, and suppose* $M(\mathbf{x}) \geq (1+\alpha) m(\mathbf{x})$*. Then* $m(S\mathbf{x}) \geq \big(1 + \frac{\alpha}{Nn}\big) m(\mathbf{x})$*.*

Proof. Let $m(\mathbf{x}) = \lambda$, and suppose k is an index such that $(S\mathbf{x})_k \geq (1+\alpha)\lambda \cdot x_k$ (such an index exists because $M(\mathbf{x}) > (1+\alpha)\lambda$. In particular, $(S\mathbf{x}) \geq \lambda \mathbf{x} + \alpha\lambda \cdot \mathbf{e}_k$, where $\mathbf{e}_k$ is the standard basis vector with the kth entry non-zero. Thus we can say that for every j,

$$A_j(S\mathbf{x}) \geq \lambda A_j \mathbf{x} + \alpha\lambda A_j \mathbf{e}_k.$$

The second term will allow us to quantify the improvement in $m(\mathbf{x})$. Note that $A_j \mathbf{e}_k = a_{jk} \geq \frac{1}{Nn} A_j \mathbf{1}$ (since A_{jk} is not too small). Now $\mathbf{1} \geq \mathbf{x}$ since $\mathbf{x}$ has q-norm 1, and thus we have

$$A_j(S\mathbf{x}) \geq \big(1 + \frac{\alpha}{Nn}\big)\lambda \cdot A_j \mathbf{x}$$

Thus $(SS\mathbf{x})_i^{q-1} \geq \big(1 + \frac{\alpha}{Nn}\big)^{q-1} \lambda^{q-1} (S\mathbf{x})_i^{q-1}$, implying that $m(S\mathbf{x}) \geq \big(1 + \frac{\alpha}{Nn}\big)\lambda$. □

This immediately implies that the value $\|A\|_{q \mapsto p}$ can be computed quickly. In particular,

Theorem 3.7. *For any* $\delta > 0$*, after* $O(Nn \cdot \mathrm{polylog}(N, n, \frac{1}{\delta}))$ *iterations, the algorithm of Section 3.1 finds a vector* x *such that* $f(x) \geq (1-\delta) f(x^*)$

Proof. To start with, the ratio $\frac{M(x)}{m(x)}$ is at most Nn (since we start with $\mathbf{1}$, and the entries of the matrix lie in $[1/N, 1]$). Lemma 3.6 now implies that the ratio drops from $(1+\alpha)$ to $(1 + \frac{\alpha}{2})$ in Nn iterations. Thus in $T = (Nn)\mathrm{polylog}(N, n, 1/\delta)$ steps, the x we end up with has $\frac{M(x)}{m(x)}$ at most $\big(1 + \frac{\delta}{(Nn)^c}\big)$ for any constant c. This then implies that $f(x) \geq f(x^*)\big(1 - \frac{\delta}{(Nn)^c}\big)$, after T iterations. □

4 Proximity to the optimum

The argument above showed that the algorithm finds a point x such that $f(x)$ is close to $f(x^*)$. We proved that for positive matrices, x^* is unique, and thus it is natural to ask if the vector we obtain is 'close' to x^*. This in fact turns out to be important in an application to oblivious routing which we consider in Section 5.

We can prove that the x we obtain after $T = (Nn)\mathrm{polylog}(N, n, 1/\delta)$ iterations is 'close' to x^*. The rough outline of the proof is the following: we first show that $f(\mathbf{x})$ is strictly concave 'around' the optimum [2]. Then we show that the 'level sets' of f are 'connected' (precise definitions follow). Then we use these to prove that if $f(x)$ is close to $f(x^*)$, then $x - x^*$ is 'small' (the choice of norm does not matter much).

Some of these results are of independent interest, and shed light into why the $q \mapsto p$ problem may be easier to solve when $p \leq q$ (even for non-negative matrices).

[2] Note that the function f is not concave everywhere (see Appendix A.1)

Concavity around the optimum. We now show that the neighborhood of every critical point (where ∇f vanishes) is strictly concave. This is another way of proving that every critical point is a maximum (this was the way [ER09] prove this fact in the $p = q$ case).

Taking partial derivatives of $f(x) = \frac{\|Ax\|_p}{\|x\|_q}$, we observe that

$$\frac{\partial f}{\partial x_i} = f(x)\Big(\frac{\sum_k (A_k x)^{p-1} a_{ki}}{\|Ax\|_p^p} - \frac{x_i^{q-1}}{\|x\|_q^q}\Big) \tag{6}$$

where A_k refers to the k^{th} row of matrix A. Now, consider a critical point z, with $\|z\|_q = 1$ (w.l.o.g.). We can also always assume that w.l.o.g. the matrix A is such that $\|Az\|_p = 1$. Thus at a critical point z, as in Eq.(2), we have that for all i:

$$\sum_k (A_k z)^{p-1} a_{ki} = z_i^{q-1} \tag{7}$$

Computing the second derivative of f at z, and simplifying using $\|Az\|_p = \|z\|_q = 1$, we obtain

$$\frac{1}{p} \cdot \frac{\partial^2 f}{\partial x_i \partial x_j}\Big|_z = (p-1)\sum_k (A_k z)^{p-2} a_{ki} a_{kj} + (q-p) z_i^{q-1} z_j^{q-1} \tag{8}$$

$$\frac{1}{p} \cdot \frac{\partial^2 f}{\partial x_i^2}\Big|_z = (p-1)\sum_k (A_k z)^{p-2} a_{ki}^2 + (q-p) z_i^{2q-2} - (q-1) z_i^{q-2} \tag{9}$$

We will now show that the Hessian H_f is negative semi-definite, which proves that f is strictly concave at the critical point z. Let ε be any vector in $\mathbb{R}^n$. Then we have (the $(q-1)z_i^{q-2}$ in (9) is split as $(p-1)z_i^{q-2} + (q-p)z_i^{q-2}$, and $\sum_{i,j}$ includes the case $i = j$)

$$\begin{aligned}
\varepsilon^T H_f \varepsilon &= p(p-1)\Big(\sum_{i,j}\sum_k (A_k z)^{p-2} \cdot a_{ki} a_{kj} \cdot \varepsilon_i \varepsilon_j - \sum_i z_i^{q-2} \varepsilon_i^2\Big) \\
&\quad + p(q-p)\Big(\sum_{i,j} (z_i z_j)^{q-1} \varepsilon_i \varepsilon_j - \sum_i z_i^{q-2} \varepsilon_i^2\Big) \\
&\equiv T_1 + T_2 \qquad \text{(say)}
\end{aligned}$$

We consider T_1 and T_2 individually and prove that they are negative. First consider T_2. Since $\sum_i z_i^q = 1$, we can consider z_i^q to be a probability distribution on integers $1, \ldots, n$. Cauchy-Schwartz now implies that $\mathbb{E}_i[(\varepsilon_i/z_i)^2] \geq \big(\mathbb{E}_i[(\varepsilon_i/z_i)]\big)^2$. This is equivalent to

$$\sum_i z_i^q \cdot \frac{\varepsilon_i^2}{z_i^2} \geq \Big(\sum_i z_i^q \cdot \frac{\varepsilon_i}{z_i}\Big)^2 = \sum_{i,j} z_i^q z_j^q \cdot \frac{\varepsilon_i \varepsilon_j}{z_i z_j} \tag{10}$$

Noting that $q \geq p$, we can conclude that $T_2 \leq 0$. Now consider T_1. Since z is a fixed point, it satisfies Eq. (7), thus we can substitute for x_i^{q-1} in the second term of T_1. Expanding out $(A_k z)$ once and simplifying, we get

$$\begin{aligned}
\frac{T_1}{p(p-1)} &= \sum_{i,j}\sum_k (A_k z)^{p-2} a_{ki} a_{kj}\Big(\varepsilon_i \varepsilon_j - \frac{z_j}{z_i} \cdot \varepsilon_i^2\Big) \\
&= -\sum_k (A_k z)^{p-2} \sum_{i,j} a_{ki} a_{kj} \cdot z_i z_j \cdot \Big(\frac{\varepsilon_i}{z_i} - \frac{\varepsilon_j}{z_j}\Big)^2 \\
&\leq 0
\end{aligned}$$

This proves that f is concave around any critical point z.

Level sets of f. Let $\mathcal{S}_q^n$, as earlier, denote the (closed, compact) set $\{\mathbf{x} \in \mathbb{R}_+^n \ : \ \|\mathbf{x}\|_q = 1\}$. Let $\mathcal{N}_\tau$ denote $\{x \in \mathcal{S}_q^n \ : \ f(x) \geq \tau\}$, i.e., $\mathcal{N}_\tau$ is an 'upper level set'. (it is easy to see that since f is continuous and A is positive, $\mathcal{N}_\tau$ is closed).

Let $S \subseteq \mathcal{S}_q^n$. We say that two points x and y are *connected* in S, if there exists a path (a continuous curve) connecting x and y, entirely contained in S (and this is clearly an equivalence relation). We say that a set S is *connected* if every $x, y \in S$ are connected in S. Thus any subset of $\mathcal{S}_q^n$ can be divided into connected components. With this notation, we show ([ER09] proves the result when $p = q$).

Lemma 4.1. *The set $\mathcal{N}_\tau$ is connected for every $\tau > 0$.*

This follows easily from techniques we developed so far.

Proof. Suppose if possible, that $\mathcal{N}_\tau$ has two disconnected components S_1 and S_2. Since there is a unique global optimum x^*, we may suppose S_1 does not contain x^*. Let y be the point in S_1 which attains maximum (of f) over S_1 (y is well defined since $\mathcal{N}$ is closed). Now if $\nabla f|_y = \vec{0}$, we get a contradiction since f has a unique critical point, namely x^* (Lemma 3.4). If $\nabla f|_y \neq \vec{0}$, it has to be normal to the surface $\mathcal{S}_q^n$ (else it cannot be that y attains maximum in the connected component S_1). Let $\mathbf{z}$ be the direction of the (outward) normal to $\mathcal{S}_q^n$ at the point y. Clearly, $\langle \mathbf{z}, y \rangle > 0$ (intuitively this is clear; it is also easy to check).

We argued that $\nabla f|_y$ must be parallel to $\mathbf{z}$, and thus it has a non-zero component along y – in particular if we scale y (equivalent to moving along y), the value of f changes, which is clearly false! Thus $\mathcal{N}_\tau$ has only one connected component. □

Since we need it for what follows, let us now prove Lemma 3.1.

Proof of Lemma 3.1. Let x^* be the optimum vector, and suppose $\|x^*\|_q = 1$. Consider the quantity

$$f(x^*)^p = \frac{\sum_i (A_i x^*)^p}{\left(\sum_i (x^*)^q\right)^{p/q}}.$$

First, note that $x_i^* \neq 0$ for any i. Suppose there is such an i. If we set $x_i = \delta$, each term in the numerator above increases by at least $\frac{p \cdot \delta}{N^p}$ (because $A_i x^*$ is at least $\frac{1}{N}$, and $(\frac{1}{N} + \frac{\delta}{N})^p > \frac{1}{N^p}(1 + p\delta)$), while the denominator increases from 1 to $(1 + \delta^q)^{p/q} \approx 1 + (p/q)\delta^q$ for small enough δ. Thus since $q > 1$, we can set δ small enough and increase the objective. This implies that x^* is a positive vector.

Note that $A_j x^* \geq \frac{\vec{\mathbf{1}}}{N} \cdot x^* \geq \frac{1}{N}$ (because the $\|x^*\|_1 \geq \|x^*\|_q = 1$). Thus for every i,

$$(Sx^*)_i^{q-1} = \sum_j a_{ij} (A_j x^*)^{p-1} \geq \frac{n}{N^p}.$$

Further, $\|A\|_p^p \leq n^{p+1}$, because each $a_{ij} \leq 1$ and so $A_j x \leq n x_{\max}$ (where $x_{\max}$ denotes the largest co-ordinate of x). Now since Eqn.(2) holds for x^*, we have

$$n^{p+1} \geq \|A\|_p^p = \frac{(Sx^*)_i^{q-1}}{(x^*)_i^{q-1}} \geq \frac{n}{N^p (x^*)_i^{q-1}}.$$

This implies that $x_i^* > \frac{1}{(Nn)^2}$, proving the lemma (we needed to use $q \geq p > 1$ to simplify). □

We now show that if $x \in \mathcal{S}_q^n$ is 'far' from x^*, then $f(x)$ is bounded away from $f(x^*)$. This, along with the fact that $\mathcal{N}_\tau$ is connected for all τ, implies that if $f(x)$ is very close to $f(x^*)$, then $\|x - x^*\|_1$ must be small. For ease of calculation, we give the formal proof only for $p = q$ (this is also the case which is used in the oblivious routing application). It should be clear that as long as we have that the Hessian at x^* is negative semidefinite, and third derivatives are bounded, the proof goes through.

Lemma 4.2 (Stability)**.** *Suppose $x \in \mathcal{S}_q^n$, with $\|x - x^*\|_1 = \delta \leq \frac{1}{(Nn)^{12}}$. Then*

$$f(x) \leq f(x^*)\Big(1 - \frac{\delta^2}{(Nn)^6}\Big) \tag{11}$$

Proof. Let ε denote the 'error vector' $\varepsilon = x - x^*$. We will use the Taylor expansion of f around x^*. H_f denotes the Hessian of f and g_f is a term involving the third derivatives, which we will get to later. Thus we have: (note that ∇f and H_f are evaluated at x^*)

$$f(x) = f(x^*) + \varepsilon \cdot \nabla f_{|x^*} + \frac{1}{2}\,\varepsilon^T H_{f|x^*}\varepsilon + g_f(\varepsilon') \tag{12}$$

At x^*, the ∇f term is 0. From the proof above that the Hessian is negative semidefinite, we have

$$\varepsilon^T H_f \varepsilon = -p(p-1)\sum_s (A_s x^*)^{p-2}\Big(\sum_{i,j} a_{si}a_{sj}x_i^* x_j^* \big(\frac{\varepsilon_i}{x_i^*} - \frac{\varepsilon_j}{x_j^*}\big)^2\Big) \tag{13}$$

We want to say that if $\|\varepsilon\|_1$ is large enough, this quantity is sufficiently negative. We should crucially use the fact that $\|x^*\|_p = \|x^* + \varepsilon\|_p = 1$ (since x is a unit vector in p-norm). This is the same as

$$\sum_i |x_i^* + \varepsilon_i|^p = \sum_i |x_i^*|^p.$$

Thus not all ε_i are of the same sign. Now since $\|\varepsilon\|_1 > \delta$, at least one of the ε_i must have absolute value at least δ/n, and some other ε_j must have the opposite sign, by the above observation. Now consider the terms corresponding to these i, j in Eqn.(13). This gives

$$\varepsilon^T H_f \varepsilon \leq -p(p-1)\sum_s (A_s x^*)^{p-2} \cdot a_{si}a_{sj} \cdot \frac{x_j^*}{x_i^*} \cdot \frac{\delta^2}{n^2} \tag{14}$$

$$\leq -p(p-1)\sum_s (A_s x^*)^{p-2} \frac{(\sum_i a_{si})^2}{(Nn)^2} \cdot \frac{1}{(Nn)^2} \cdot \frac{\delta^2}{n^2} \tag{15}$$

$$\leq -p(p-1) \cdot \frac{\delta^2}{N^4 n^6} \cdot \|Ax^*\|_p^p \tag{16}$$

Note that we used the facts that entries a_{ij} lie in $[\frac{1}{N}, 1]$ and that $x_i^* \in [\frac{1}{(Nn)^2}, 1]$. Thus it only remains to bound the third order terms (g_f, in Eqn.(12)). This contribution equals

$$g_f(\varepsilon) = \frac{1}{3!}\sum_i \varepsilon_i^3 \frac{\partial^3 f}{\partial x_i^3} + \frac{1}{2!}\sum_{i,j} \varepsilon_i^2 \varepsilon_j \frac{\partial^3 f}{\partial x_i^2 \partial x_j} + \sum_{i<j<k} \varepsilon_i\varepsilon_j\varepsilon_k \frac{\partial^3 f}{\partial x_i \partial x_j \partial x_k} \tag{17}$$

It can be shown by expanding out, and using the facts that $m_{si} \leq N(M_s x^*)$ and $\frac{1}{x_i^*} \leq (Nn)^2$, that for i, j, k,

$$\frac{\partial^3 f}{\partial x_i \partial x_j \partial x_k} \leq 10p^3(Nn)^3\|Ax^*\|_p^p.$$

Thus, the higher order terms can be bounded by

$$g_f(\varepsilon) \leq 10p^3 \cdot n^6 N^3 \cdot \|Ax^*\|_p^p \cdot \delta^3$$

So, if $\delta < \frac{1}{10p^3} \cdot \frac{1}{(Nn)^{12}}$, the Hessian term dominates. Thus we have, as desired:

$$f(x) \leq f(x^*)\Big(1 - \frac{\delta^2}{(Nn)^6}\Big)$$

□

This proves that the vector we obtain at the end of the T iterations (for T as specified) has an ℓ_1 distance at most $\frac{1}{(Nn)^c}$ to x^*. Thus we have a polynomial time algorithm to compute x^* to any accuracy.

5 An Application - $O(\log n)$ Oblivious routing scheme for ℓ_p

We believe that our algorithm for computing the $\|A\|_{q\mapsto p}$ (for non-negative matrices) could find good use as an optimization tool. For instance, eigenvalue computation is used extensively, not just for partitioning and clustering problems, but also as a subroutine for solving semi-definite programs [GLS88]. We now give one application of our algorithm and the techniques we developed in section 4 to the case of oblivious routing in the ℓ_p norm.

Oblivious routing. As outlined in the Introduction, the aim in oblivious routing is, given a graph $G = (V, E)$, to specify how to route a unit flow between every pair of vertices in V. Now, given a demand vector (demands between pairs of vertices), these unit flows are scaled linearly by the demands, and routed (let us call this the oblivious flow). This oblivious flow is compared to the best flow in hindsight i.e. knowing the demand vector, with respect to some objective (say congestion), and we need to come up with a scheme which bounds this competitive ratio in the worst case.

Gupta et al. [GHR06] consider the oblivious routing problem where the cost of a solution is the ℓ_p norm of the 'flow vector' (the vector consisting of total flow on each edge). In the case $p = \infty$, this is the problem of minimizing congestion, for which the celebrated result of [RÖ08] gave an $O(\log n)$ competitive scheme. For the ℓ_1 version of the problem, the optimal solution (as is easily seen) is to route along shortest paths for each demand pair. The ℓ_p version tries to trade-off between these two extremes.

By a clever use of zero sum games, [ER09] reduced the problem of showing existence good oblivious routing schemes for any p to the ℓ_∞ case. This showed (by a *non-constructive* argument) the existence of an $O(\log n)$ oblivious routing scheme for any $p \geq 1$. They then make their result constructive for $p = 2$ (the proof relies heavily on eigenvectors being orthogonal). Using our algorithm for finding the ℓ_p-norm of a matrix and the stability of our maxima (Lemma 4.2), we make the result constructive for all ℓ_p.

Zero-sum game framework of [ER09]: We first give a brief overview of the non-constructive proof from [ER09]. The worst-case demands for any tree-based oblivious routing scheme can be shown to be those with non-zero demands only on the edges of the graph. The *competitive ratio* of any tree-based oblivious routing scheme can then be reduced to a matrix p-norm computation: if M is a $|E| \times |E|$-dimensional matrix which represents a tree-based oblivious routing scheme which

specifies unit flows for each demand across an edge of the graph, the competitive ratio is given by $\max_{\|\mathbf{u}\|_p \leq 1} \|M\mathbf{u}\|_p$ where $\mathbf{u} \in \mathbb{R}^{|E|}$.

To show the existence of an oblivious routing scheme with competitive ratio $O(\log n)$, [ER09] define a continuous two player zero-sum game. The first player (row player) chooses from the set of all *tree-based oblivious routing matrices* (of dimension $|E| \times |E|$). The second player's (column player) strategy set is the set of vectors $\mathbf{u} \in \mathbb{R}^{|E|}$ with positive entries, and $\|u\|_p = 1$, and the value of the game is $\|M\mathbf{u}\|_p$. With a clever use of min-max duality in zero sum games and the oblivious routing scheme of [R08] for congestion (ℓ_∞-norm) as a blackbox, [ER09] show the non-constructive existence of an oblivious routing scheme M which gets a value of $O(\log n)$ for all demand vectors.

Finding such a (tree-based) oblivious routing scheme requires us to solve this zero-sum game efficiently. The constructive algorithm from [ER09] for ℓ_2, however crucially uses the ortho-normality of the eigenspace for $\|M\|_2$ computation, to solve the aforementioned zero-sum game. First we state without proof a couple of lemmas from [ER09], which will also feature in our algorithm.

Lemma 5.1. *Let OBL be a tree-based oblivious routing scheme given by a $|E| \times \binom{n}{2}$ dimensional matrix (non-negative entries) and let its restriction to edges be $OBL' \in \mathbb{R}^{|E| \times |E|}$. The competitive ratio of the oblivious algorithm is at most $\|OBL'\|_p$.*

Henceforth, we shall abuse notation and use OBL to refer to both the tree-based Oblivious routing matrix and its restriction to edges interchangeably. Further,

Lemma 5.2. *For any given vector $\mathbf{u} \in \mathbb{R}^{|E|}$, there exists an tree-based Oblivious routing scheme (denoted by matrix OBL) such that*

$$\|OBL \cdot \mathbf{u}\|_p \leq O(\log n)\|\mathbf{u}\|_p$$

This lemma shows that for every vector $\mathbf{u}$, there exists some routing scheme (which could depend on the vector) that is $O(\log n)$ competitive. We will now show how to compute *one* tree-based routing matrix OBL that works for all vectors i.e. $\|OBL\|_p \leq 1$. From Lemma 5.2, we know that for every unit vector $\mathbf{u}$, there exists an tree-based oblivious routing matrix such that $\|M \cdot \mathbf{u}\|_p \leq O(\log n)$. We use this to construct one tree-based oblivious routing matrix OBL that works for every load vector $\mathbf{u}$. Note that the set of tree-based oblivious routing schemes is convex. Before, we show how to construct the oblivious routing scheme, we present a simple lemma which captures the continuity of the p-norm function.

Lemma 5.3. *Let $f = \frac{\|Ax\|_p^p}{\|x\|_p^p}$, where A is an $n \times n$ matrix with minimum entry $\frac{1}{N}$ and let y be an n-dimensional vector with minimum entry $\frac{1}{(Nn)^2}$. Let x be a vector in the δ-neighborhood of y i.e. $\|x - y\|_1 = \delta \leq \frac{1}{(Nn)^{12}}$. Then,*

$$f(x) \leq f(y) + 1 \tag{18}$$

Proof. The proof follows just from the continuity and differentiability of the p-norm function at every point. Using the Taylor's expansion of f, we see that

$$f(x) = f(y) + \varepsilon \cdot \nabla f_y + \frac{1}{2}\, \varepsilon'^T H_{f|y} \varepsilon' \tag{19}$$

where $0 \leq \varepsilon' \leq \varepsilon$. Choosing $\varepsilon = \delta = \frac{1}{(Nn)^{12}}$ and using the lower bounds the matrix entries and the co-ordinates of y as in Lemma 4.2, we see that the lemma follows. □

We now sketch how to find a tree-based oblivious routing matrix when the aggregation function is an ℓ_p norm.

Theorem 5.4. *There exists a polynomial time algorithm that computes an oblivious routing scheme with competitive ratio $O(\log n)$ when the aggregation function is the ℓ_p norm with $p > 1$ and the load function on the edges is a norm.*

Proof sketch. The algorithm and proof follow roughly along the lines of the constructive version for $p = 2$ in [ER09]. As mentioned earlier, their proof uses inner products among the vectors (and the computation of eigenvalues). However, we show that the procedure still works because of the stability of our solution (Lemma 4.2.

Let J_ε be an $|E| \times |E|$ matrix will all entries being ϵ. Let $f(M) = \|M + J_{\frac{1}{|E|}}\|_p$. We want a tree-based oblivious routing matrix OBL such that $f(OBL) \leq c \log n$ for some large enough constant c. We follow an iterative procedure to obtain this matrix OBL starting with an arbitrary tree-based routing matrix M_0. At stage i, we check if for the current matrix M_i , $\|M_i\|_p \leq c \log n$. If not, using the iterative algorithm in Section 3, we obtain unit vector $x^*_{(i)}$ which maximizes $\|M_{(i)}x\|_p$. Let $\tilde{M}_{(i)}$ be the tree-based oblivious routing matrix from Lemma 5.2 such that $\|\tilde{M}_{(i)}x^*_{(i)}\|_p \leq c \log n/2 - 2$. We now update

$$M_{i+1} = (1 - \lambda)M_i + \lambda \tilde{M}_i$$

Observe that this is also a tree-based oblivious routing matrix. We now show that $\|M_{i+1}\|_p$ decreases by an amount $\Omega(\frac{1}{\text{poly}(n)})$.

At step i, roughly speaking, for all vectors y that are far enough from $x^*_{(i)}$, $\|M_i y\|_p \leq \|M_i y\|_p - \frac{1}{\text{poly}(n)}$ from Lemma 4.2 (stability). Choosing $\lambda = \Theta(n^{-c})$ for some large enough constant $c > 0$, it easily follows that $\|M_{i+1}y\|_p \leq \|M_i y\|_p - \frac{1}{\text{poly}(n)}$. On the other hand, consider y in the δ-neighborhood of $x^*_{(i)}$. Using Lemma 5.3,

$$\|\tilde{M}_i y\|_p \leq \frac{c \log n}{2}$$

Hence,

$$\begin{aligned} \|M_{i+1}y\|_p &= (1 - \lambda)\|M_i y\|_p + \lambda \frac{c}{2} \log n \\ &\leq \|M_i y\|_p - \lambda \times \frac{c}{2} \log n \qquad (\text{since } \|M_i y\|_p \geq c \log n \ ) \\ &\leq \|M_i y\|_p - \frac{1}{\text{poly}(n)} \end{aligned}$$

Hence, it follows that the matrices M_i decrease in their p-norm by a small quantity $\Omega(\frac{1}{\text{poly}(n)})$ in every step. It follows that this iterative algorithm finds the required tree-based oblivious routing scheme in $\text{poly}(n)$ steps. □

6 Inapproximability results

We will now prove that it is NP-hard to approximate $\|A\|_{q \mapsto p}$-norm of a matrix to any fixed constant, for any $q \geq p > 2$. We then show how this proof carries over to the hardness of computing the $\infty \mapsto p$ norm.

6.1 Inapproximability of $\|A\|_{p \mapsto p}$

Let us start with the question of approximating $\|A\|_{p \mapsto p}$. We first show the following:

Proposition 6.1. *For $p \geq 2$ it is NP-hard to approximate that $\|A\|_p$ to some (small) constant factor $\eta > 1$.*

Proof: We give a reduction from the gap version of the MaxCut problem. The following is well-known (c.f. [Hås01])

> There exist constants $1/2 \leq \rho < \rho' < 1$ such that given a regular graph $G = (V, E)$ on n vertices and degree d, it is hard to distinguish between:
> YES case: G has a cut containing at least $\rho'(nd/2)$ edges, and
> NO case: No cut in G cuts more that $\rho(nd/2)$ edges.

Suppose we are given a graph $G = (V, E)$ which is regular and has degree d. The p-norm instance we consider will be that of maximizing $g(x_0, \ldots, x_n)$ ($x_i \in \mathbb{R}^n$), defined by

$$g(x_0, x_1, \ldots, x_n) = \frac{\sum_{i \sim j} |x_i - x_j|^p + Cd \cdot \left(\sum_i |x_0 + x_i|^p + |x_0 - x_i|^p \right)}{n|x_0|^p + \sum_i |x_i|^p}.$$

Here C will be chosen appropriately later. Note that if we divide by d, we can see $g(\mathbf{x})$ as the ratio

$$\frac{g(\mathbf{x})}{d} = \frac{\sum_{i \sim j} \left(|x_i - x_j|^p + C(|x_0 + x_i|^p + |x_0 - x_i|^p + |x_0 + x_j|^p + |x_0 - x_j|^p) \right)}{\sum_{i \sim j} 2|x_0|^p + |x_i|^p + |x_j|^p}. \tag{20}$$

The idea is to do the analysis on an edge-by-edge basis. Consider the function

$$f(x, y) = \frac{|x - y|^p + C\big(|1 + x|^p + |1 - x|^p + |1 + y|^p + |1 - y|^p\big)}{2 + |x|^p + |y|^p}.$$

Definition. A tuple (x, y) is *good* if both $|x|$ and $|y|$ lie in the interval $(1 - \varepsilon, 1 + \varepsilon)$, and $xy < 0$. A technical lemma concerning f is the following

Lemma 6.2. *For any $\varepsilon > 0$, there is a large enough constant C such that*

$$f(x, y) \leq \begin{cases} C \cdot 2^{p-1} + \frac{(1+\varepsilon)2^p}{2 + |x|^p + |y|^p}, & \text{if } (x, y) \text{ is good} \\ C \cdot 2^{p-1} & \text{otherwise} \end{cases} \tag{21}$$

We now present the proof of Lemma 6.2. We first start with a simpler inequality - note that this is where the condition $p > 2$ comes in.

Lemma 6.3. *For all $x \in \mathbb{R}$, we have*

$$\frac{|1 + x|^p + |1 - x|^p}{1 + |x|^p} \leq 2^{p-1}.$$

Further, for any $\varepsilon > 0$, there exists a $\delta > 0$ such that if $|x| \notin [1 - \varepsilon, 1 + \varepsilon]$, then

$$\frac{|1 + x|^p + |1 - x|^p}{1 + |x|^p} \leq 2^{p-1} - \delta.$$

Proof. We may assume $x > 0$. First consider $x > 1$. Write $x = 1 + 2\theta$, and thus the first inequality simplifies to

$$\left[(1 + 2\theta)^p - (1 + \theta)^p \right] \geq (1 + \theta)^p - 1 + 2\theta^p.$$

Now consider

$$I = \int_{x=1}^{1+\theta} \big((x+\theta)^{p-1} - x^{p-1}\big)dx.$$

For each x, the function being integrated is $\geq \theta^{p-1}$, since $p > 2$ and $x > 0$. Thus the integral is at least θ^p. Now evaluating the integral independently and simplifying, we get

$$(1+2\theta)^p - 2(1+\theta)^p + 1 \geq p \cdot \theta^p,$$

which gives the inequality since $p > 2$. Further there is a slack of $(p-2)\theta^p$. Now suppose $0 < x < 1$. Writing $x = 1 - 2\theta$ and simplifying similarly, the inequality follows. Further, since we always have a slack, the second inequality is also easy to see. □

Proof (of Lemma 6.2). The proof is a straight-forward case analysis. Call x (resp. y) 'bad' if $|x| \notin [1-\varepsilon, 1+\varepsilon]$. Also, $b(x)$ denotes a predicate which is 1 if x is bad and 0 otherwise.
Case 1. (x, y) is good. The upper bound in this case is clear (using Lemma 6.3).
Case 2. Neither of x, y are bad, but $xy > 0$. Using Lemma 6.3, we have $f(x, y) \leq C \cdot 2^{p-1} + \varepsilon$, which is what we want.
Case 3. At least one of x, y are bad (i.e., one of $b(x), b(y)$ is 1). In this case Lemma 6.3 gives

$$\begin{aligned} f(x,y) &\leq \frac{|x-y|^p + C\big((1+|x|^p)(2^{p-1} - \delta b(x)) + (1+|y|^p)(2^{p-1} - \delta b(y))\big)}{2 + |x|^p + |y|^p} \\ &= C \cdot 2^{p-1} + \frac{|x-y|^p - C\big(\delta b(x)(1+|x|^p) + \delta b(y)(1+|y|^p)\big)}{2 + |x|^p + |y|^p} \end{aligned}$$

Since $|x-y|^p \leq 2^{p-1}(|x|^p + |y|^p)$, and one of $b(x), b(y) > 0$, we can choose C large enough (depending on δ), so that $f(x, y) \leq C \cdot 2^{p-1}$. □

Soundness. Assuming the lemma, let us see why the analysis of the NO case follows. Suppose the graph has a Max-Cut value at most ρ, i.e., every cut has at most $\rho \cdot nd/2$ edges. Now consider the vector x which maximizes $g(x_0, x_1, \ldots, x_n)$. It is easy to see that we may assume $x_0 \neq 0$, thus we can scale the vector s.t. $x_0 = 1$. Let $S \subseteq V$ denote the set of 'good' vertices (i.e., vertices for which $|x_i| \in (1-\varepsilon, 1+\varepsilon)$).

Lemma 6.4. *The number of good edges is at most* $\rho \cdot \frac{(|S|+n)d}{4}$.

Proof. Recall that good edges have both end-points in S, and further the corresponding x values have opposite signs. Thus the lemma essentially says that there is no cut in S with $\rho \cdot \frac{(|S|+n)d}{4}$ edges.

Suppose there is such a cut. By greedily placing the vertices of $V \setminus S$ on one of the sides of this cut, we can extend it to a cut of the entire graph with at least

$$\rho \cdot \frac{(|S|+n)d}{4} + \frac{(n-|S|)d}{4} = \frac{\rho nd}{2} + \frac{(1-\rho)(n-|S|)}{4} > \frac{\rho nd}{2}$$

edges, which is a contradiction. This gives the bound. □

Let $\mathcal{N}$ denote the numerator of Eq.(20). We have

$$\begin{aligned}\mathcal{N} &= \sum_{i\sim j} f(x_i, x_j)(2 + |x_i|^p + |x_j|^p) \\ &\leq C \cdot 2^{p-1} \cdot \big(nd + d\sum_i |x_i|^p\big) + \sum_{i\sim j,\ \text{good}} (1+\varepsilon)2^p \\ &\leq Cd \cdot 2^{p-1} \cdot \big(n + \sum_i |x_i|^p\big) + \frac{\rho d(n+|S|)}{4} \cdot 2^p(1+\varepsilon).\end{aligned}$$

Now observe that the denominator is $n + \sum_i |x_i|^p \geq n + |S|(1-\varepsilon)^p$, from the definition of S. Thus we obtain an upper bound on $g(x)$

$$g(x) \leq Cd \cdot 2^{p-1} + \frac{\rho d}{4} \cdot 2^p(1+\varepsilon)(1-\varepsilon)^{-p}.$$

Hardness factor. In the YES case, there is clearly an assignment of ± 1 to x_i such that $g(\mathbf{x})$ is at least $Cd \cdot 2^{p-1} + \frac{\rho' d}{4} \cdot 2^p$. Thus if ε is small enough (this will make us pick C which is large), the gap between the optimum values in the YES and NO cases can be made $\big(1 + \frac{\Omega(1)}{C}\big)$, where the $\Omega(1)$ term is determined by the difference $\rho' - \rho$. This proves that the p-norm is hard to approximate to some fixed constant factor.
Note. In the analysis, ε was chosen to be a small constant depending on p and the gap between ρ and ρ'; C is a constant chosen large enough, depending on ε.

The Instance. We have argued about the hardness of computing the function $g(x_0, x_1, \ldots, x_n)$ to some constant factor. This can be formulated as an instance of p-norm in a natural way. We describe this formally (though this is clear, the formal description will be useful when arguing about certain properties of the tensored instance which we need for proving hardness of $\|A\|_{q\mapsto p}$ for $p < q$).

First we do a simple change of variable and let $z = n^{1/p}x_0$. Now, we construct the $5|E| \times (n+1)$ matrix M. For each edge $e = \{i, j\}$ in $E(G)$, we have five rows in M. Let the column indices run from $0 \leq \ell \leq n$.

$$\mathbf{M}_{e_1,\ell} = \begin{cases} 1 \text{ if } \ell = i & \text{and} \quad -1 \text{ if } \ell = j \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{M}_{e_2,\ell} = \begin{cases} n^{-1/p} & \text{if } \ell = 0 \quad \text{and} \quad -1 \quad \text{if } \ell = i \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{M}_{e_3,\ell} = \begin{cases} n^{-1/p} & \text{if } \ell = 0 \quad \text{and} \quad 1 \quad \text{if } \ell = i \\ 0 & \text{otherwise} \end{cases}$$

We have two similar rows for $\mathbf{M}_{e_4,\ell}$ and $\mathbf{M}_{e_5,\ell}$ where we have corresponding values with j instead of i. It is easy to see that this matrix actually takes the same value $\|M\|_p$ as g. Further in the YES case, there is a vector $x = (n^{1/p}, x_1, x_2, \ldots, x_n)$ with $x_i = \pm 1$, that attains the high value $(C.d.2^{p-1} + \rho' d.2^{p-2})$.

6.2 Amplifying the gap by tensoring

We observe that the matrix $p \mapsto p$-norm is multiplicative under tensoring. More precisely,

Lemma 6.5. *Let M, N be square matrices with dimensions $m \times m$ and $n \times n$ respectively, and let $p \geq 1$. Then $\|M \otimes N\|_p = \|M\|_p \cdot \|N\|_p$.*

The tensor product $M \otimes N$ is defined in the standard way – we think of it as an $m \times m$ matrix of blocks, with the i, jth block being a copy of N scaled by m_{ij}. It is well-known that eigenvalues ($p = 2$) mutliply under tensoring. We note that it is crucial that we consider $\|A\|_p$. Matrix norms $\|A\|_{q \mapsto p}$ for $p \neq q$ do not in general multiply upon tensoring.

Proof. Let $\lambda(A)$ denote the p-norm of a matrix A. Let us first show the easy direction, that $\lambda(M \otimes N) \geq \lambda(M) \cdot \lambda(N)$. Suppose x, y are the vectors which 'realize' the p-norm for M, N respectively. Then

$$\begin{aligned}\|(M \otimes N)(x \otimes y)\|_p^p &= \sum_{i,j} |(M_i \cdot x)(N_j \cdot y)|^p \\ &= \Big(\sum_i (M_i \cdot x)^p\Big)\Big(\sum_j (N_j \cdot y)^p\Big) \\ &= \lambda(M)^p \cdot \lambda(N)^p\end{aligned}$$

Also $\|x \otimes y\|_p = \|x\|_p \cdot \|y\|_p$, thus the inequality follows.

Let us now show the other direction, i.e., $\lambda(M \otimes N) \leq \lambda(M) \cdot \lambda(N)$. Let x, z be mn dimensional vectors such $z = (A \otimes B)x$. We will think of x, z as being divided into m blocks of size n each. Further by $x^{(i)}$ (and $z^{(i)}$), we denote the vector in $\mathbb{R}^n$ which is formed by the ith block of x (resp. z).

For ease of notation, let us consider $n \times m$ matrix X with the i^{th} column of X being vector $x^{(i)}$ (and similarly define Z). At the expense of abusing notation, let X_j^t refer to the j^{th} row of matrix X. Also, let the element-wise p-norm of matrix M be defined as

$$|M|_{\odot p} = \big(\sum_{i,j} m_{ij}^p\big)^{1/p}$$

It is easy to observe that $Z^t = AX^tBT$. Further, $\|z\|_p = |Z|_{\odot p}$.

We now expand out Z and rearrange the terms to separate out the operations of B and A, in order to bound $|Z|_{\odot p}$ using the p-norms of A and B. Hence, we have

$$\begin{aligned}\|z\|_p^p &= |AX^tB^t|_{\odot p}^p \\ &= \sum_{i=1}^m \sum_{j=1}^n \Big((A_iX_1, A_iX_2, \ldots, A_iX_n)B_j^t\Big)^p\end{aligned}$$

But from definition, $\|Mx\|_p^p = \sum_k (B_kx)^p \leq \lambda(M)\|x\|_p^p$.

Hence, by applying the operator p-norm bound of B,

$$\begin{aligned}\|z\|_p^p &\le \lambda(B)^p \sum_i \|(A_iX_1, A_iX_2, \ldots, A_iX_n)\|_p^p \\ &= \lambda(B)^p \sum_{i=1}^{m}\sum_{k=1}^{n}(A_iX_k)^p = \lambda(B)^p \sum_k \sum_i (A_iX_k)^p \\ &\le \lambda(B)^p\lambda(A)^p \sum_k \|X_k\|_p^p \\ &= \lambda(A)^p\lambda(B)^p |X|_{\odot p}^p\end{aligned}$$

Since $|X|_{\odot p} = \|x\|_p = 1$, we have $\|z\|_p \le \lambda(A)\lambda(B)$.

□

Note. We note that the tensoring result is stated for square matrices, while the instances above are rectangular. This is not a problem, because we can pad 0s to the matrix to make it square without changing the value of the norm.

The tensored matrix and amplification Consider any constant $\gamma > 0$. We consider the instance of the matrix M obtained in the proof of Proposition 6.1, and repeatedly tensor it with M $k = \log_\eta \gamma$ times to obtain $M' = M^{\otimes k}$. From Lemma 6.5, there exist τ_C and τ_S where $\tau_C/\tau_S > \gamma$ such that in the NO case, for every vector $\mathbf{y} \in \mathbb{R}^{(n+1)k}$, $\|Ay\|_p \le \tau_S$.

Further, in the YES case, there is a vector $y' = (n^{1/p}, x_1, x_2, \ldots, x_n)^{\otimes k}$ where $x_i = \pm 1$ (for $i = 1, 2, \ldots n$) such that $\|M'y'\|_p \ge \tau_C$.

Note: Our techniques work even when we take the tensor product $\log^c n$ for some constant c. Thus we can conclude:

Theorem 6.6. *For any $\gamma > 0$ and $p \ge 2$, it is NP-hard to approximate the p-norm of a matrix within a factor γ. Also, it is hard to approximate the matrix p-norm to a factor of $\Omega(2^{(\log n)^{1-\varepsilon}})$ for any constant $\varepsilon > 0$, unless $\mathsf{NP} \subseteq \mathsf{DTIME}(2^{polylog(n)})$.*

Properties of the tensored instance: We now establish some structure about the tensored instance, which we will use crucially for the hardness of $q \mapsto p$ norm. Let the entries in vector y' be indexed by k-tuple $I = (i_1, i_2, \ldots, i_k)$ where $i_k \in \{0, 1, \ldots, n)$. It is easy to see that

$$y'_I = \pm n^{w(I)/p} \text{ where } w(I) \text{ number of } 0s \text{ in tuple}$$

Let us introduce variables $x_I = n^{-w(I)/p} y_I$ where $w(I) =$ number of $0s$ in tuple I. It is easy to observe that there is a matrix B such that

$$\frac{\|M'y\|_p}{\|y\|_p} = \frac{\|B\mathbf{x}\|_p}{\sum_I n^{w(I)}|x_I|^p} = g'(x)$$

Further, it can also be seen that in the YES case, there is a ± 1 assignment for x_I which attains the value $g'(x) = \tau_C$.

6.3 Approximating $\|A\|_{q\mapsto p}$ when $p \ne q$.

Let us now consider the question of approximating $\|A\|_{q\mapsto p}$. The idea is to use the hardness of approximating $\|A\|_{p\mapsto p}$. We observed in the previous section that the technique of amplifying

hardness for computing the $q \mapsto p$-norm by tensoring a (small) constant factor hardness does not work when $q \neq p$. However, we show that we can obtain such amplified label-cover like hardness if the instance has some additional structure. In particular, we show the instances that we obtain from the tensoring the hard instances of $\|A\|_p$ can be transformed to give such hard instances for $\|A\|_{q\mapsto p}$.

We illustrate the main idea by first showing a (small) constant factor hardness: let us start with the following maximization problem (which is very similar to Eqn.(20))

$$g(x_0, x_1, \ldots, x_n) = \frac{\Big(\sum_{i\sim j} |x_i - x_j|^p + Cd \cdot \big(\sum_i |x_0 + x_i|^p + |x_0 - x_i|^p\big)\Big)^{1/p}}{\big(n|x_0|^q + \sum_i |x_i|^q\big)^{1/q}}. \tag{22}$$

Notice that x_0 is now 'scaled differently' than in Eq.(20). This is crucial. Now, in the YES case, we have

$$\max_{\mathbf{x}} \; g(\mathbf{x}) \geq \frac{\big(\rho'(nd/2) \cdot 2^p + Cnd \cdot 2^p\big)^{1/p}}{(2n)^{1/q}}.$$

Indeed, there exists a ± 1 solution which has value at least the RHS. Let us write $\mathcal{N}$ for the numerator of Eq.(22). Then

$$g(\mathbf{x}) = \frac{\mathcal{N}}{\big(n|x_0|^p + \sum_i |x_i|^p\big)^{1/p}} \times \frac{\big(n|x_0|^p + \sum_i |x_i|^p\big)^{1/p}}{\big(n|x_0|^q + \sum_i |x_i|^q\big)^{1/q}}.$$

Suppose we started with a NO instance. The proof of the $q = p$ case implies that the first term in this product is at most (to a $(1+\varepsilon)$ factor)

$$\frac{\big(\rho(nd/2) \cdot 2^p + Cnd \cdot 2^p\big)^{1/p}}{(2n)^{1/p}}.$$

Now, we note that the second term is at most $(2n)^{1/p}/(2n)^{1/q}$. This follows because for any vector $y \in \mathbb{R}^n$, we have $\|y\|_p/\|y\|_q \leq n^{(1/p)-(1/q)}$. We can use this with the $2n$-dimensional vector $(x_0, \ldots, x_0, x_1, x_2, \ldots, x_n)$ to see the desired claim.

From this it follows that in the NO case, the optimum is at most (upto an $(1+\varepsilon)$ factor)

$$\frac{\big(\rho(nd/2) \cdot 2^p + Cnd \cdot 2^p\big)^{1/p}}{(2n)^{1/q}}.$$

This proves that there exists an $\alpha > 1$ s.t. it is NP-hard to approximate $\|A\|_{q\mapsto p}$ to a factor better than α.

A key property we used in the above argument is that in the YES case, there exists a ± 1 solution for the x_i ($i \geq 0$) which has a large value. It turns out that this is the only property we need. More precisely, suppose A is an $n \times n$ matrix, let α_i be positive integers (we will actually use the fact that they are integers, though it is not critical). Now consider the optimization problem $\max_{\mathbf{y}\in\mathbb{R}^n} g(\mathbf{y})$, with

$$g(\mathbf{y}) = \frac{\|A\mathbf{y}\|_p}{(\sum_i \alpha_i |y_i|^p)^{1/p}} \tag{23}$$

In the previous section, we established the following claim from the proof of Theorem 6.6.

Claim 6.7. *For any constant $\gamma > 1$, there exist thresholds τ_C and τ_S with $\tau_C/\tau_S > \gamma$, such that it is NP-hard to distinguish between:*

YES *case. There exists a* ± 1 *assignment to* y_i *in* (23) *with value at least* τ_C*, and*
NO *case. For all* $\mathbf{y} \in \mathbb{R}^n$*,* $g(\mathbf{y}) \leq \tau_S$*.*

Proof. Follows from the structure of Tensor product instance.

We can now show that Claim 6.7 implies the desired result.

Theorem 6.8. *It is NP-hard to approximate* $\|A\|_{q \mapsto p}$ *to any fixed constant* γ *for* $q \geq p > 2$*.*

Proof. As in previous proof (Eq.(22)), consider the optimization problem $\max_{\mathbf{y} \in \mathbb{R}^n} h(\mathbf{y})$, with

$$h(\mathbf{y}) = \frac{\|A\mathbf{y}\|_p}{(\sum_i \alpha_i |y_i|^q)^{1/q}} \tag{24}$$

By definition,

$$h(\mathbf{y}) = g(\mathbf{y}) \cdot \frac{(\sum_i \alpha_i |y_i|^p)^{1/p}}{(\sum_i \alpha_i |y_i|^q)^{1/q}}. \tag{25}$$

Completeness. Consider the value of $h(\mathbf{y})$ for A, α_i in the YES case for Claim 6.7. Let $\mathbf{y}$ be a ± 1 solution with $g(\mathbf{y}) \geq \tau_C$. Because the y_i are ± 1, it follows that

$$h(\mathbf{y}) \geq \tau_C \cdot \big(\sum_i \alpha_i\big)^{(1/p)-(1/q)}.$$

Soundness. Now suppose we start with an A, α_i in the NO case for Claim 6.7.

First, note that the second term in Eq.(25) is at most $\big(\sum_i \alpha_i\big)^{(1/p)-(1/q)}$. To see this, we note that α_i are positive integers. Thus by considering the vector $(y_1, \ldots, y_1, y_2, \ldots, y_2, \ldots)$, (where y_i is duplicated α_i times), and using $\|u\|_p / \|u\|_q \leq d^{(1/p)-(1/q)}$ for $u \in \mathbb{R}^d$, we get the desired inequality.

This gives that for all $\mathbf{y} \in \mathbb{R}^n$,

$$h(\mathbf{y}) \leq g(\mathbf{y}) \cdot \big(\sum_i \alpha_i\big)^{(1/p)-(1/q)} \leq \tau_S \cdot \big(\sum_i \alpha_i\big)^{(1/p)-(1/q)}.$$

This proves that we cannot approximate $h(\mathbf{y})$ to a factor better than τ_C/τ_S, which can be made an arbitrarily large constant by Claim 6.7. This finishes the proof, because the optimization problem $\max_{\mathbf{y} \in \mathbb{R}^n} h(\mathbf{y})$ can be formulated as a $q \mapsto p$ norm computation for an appropriate matrix as earlier. □

Note that this hardness instance is not obtained by tensoring the $q \mapsto p$ norm hardness instance. It is instead obtained by considering the $\|A\|_p$ hardness instance and transforming it suitably.

6.4 Approximating $\|A\|_{\infty \mapsto p}$

The problem of computing the $\infty \mapsto p$ norm of a matrix A turns out to have a very natural and elegant statement in terms of column vectors of the matrix A. We first introduce the following problem:

Definition 6.9 (Longest Vector Problem)**.** *Let* $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}$ *be vectors over* $\mathbb{R}$*. The Longest Vector problem asks for the*

$$\max_{\mathbf{x} \in \{-1,1\}^n} \|\sum_i x_i \mathbf{v_i}\|_p$$

Note that this problem differs from the well-studied Shortest Vector Problem [Kho04] for lattices,which has received a lot of attention in the cryptography community over the last decade [Reg06]. The shortest vector problem asks for *minimizing* the same objective in Definition 6.9 when $x_i \in \mathbb{Z}$.

We now observe that computing the $\infty \mapsto p$ norm of the matrix is equivalent to finding the length of the longest vector, where the vectors $\mathbf{v_i}$ are the columns of A.

Observation 6.10. *Computing the $\|A\|_{\infty \mapsto p}$ norm of a matrix is equivalent to computing the length of the* Longest vector problem *where the vectors are the column vectors of A.*

Proof. First note that $\|A\mathbf{x}\|_p = \|\sum_i x_i \mathbf{a_i}\|_p$. The observation follows by noticing that this is maximized when $|x_i| = 1$ for all i. □

The $\infty \mapsto p$ norm of the matrix also seems like a natural extension of the Grothendieck problem[AN04, KNS08]. When $p = 1$, we obtain the original Grothendieck problem, and the $p = 2$ case is the ℓ_2 Grothendieck problem and maximizes the quadratic form for p.s.d. matrices. Further, as mentioned earlier there is a constant factor approximation for $1 \leq p \leq 2$ using [Nes98]. However, for the $p > 2$, we show that there is $\Omega(2^{(\log n)^{1-\epsilon}})$ hardness for computing $\infty \mapsto p$ norm assuming NP does not have quasipolynomial time algorithms using the same techniques from Theorem 6.8.

Theorem 6.11. *It is NP-hard to approximate $\|A\|_{\infty \mapsto p}$ to any constant γ for $p > 2$ and hard to approximate within a factor of $\Omega(2^{(\log n)^{1-\varepsilon}})$ for any constant $\varepsilon > 0$, assuming $\mathsf{NP} \notin \mathsf{DTIME}(2^{polylog(n)})$.*

The proof of Theorem 6.8 also works out for $q = \infty$ by noting that the second expression in Eq. (25) is instead $\max_{\mathbf{x}} \frac{(\sum_i \alpha_i |x_i|^p)^{1/p}}{||x||_\infty}$ which is also maximized when $|x_i| = 1$ for all i.

7 Acknowledgements

We would like to thank our advisor Moses Charikar for many useful suggestions and comments throughout the progress of the work. We would also like to thank David Steurer for several discussions and pointing out connections to other problems. Finally we would like to thank Rajsekar Manokaran for various interactions about the inapproximability results.

References

[AN04] Noga Alon and Assaf Naor. Approximating the cut-norm via grothendieck's inequality. In *STOC '04: Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 72–80, New York, NY, USA, 2004. ACM.

[BCC+10] Aditya Bhaskara, Moses Charikar, Eden Chlamtac, Uriel Feige, and Aravindan Vijayaraghavan. Detecting high log-densities – an $o(n^1/4)$ approximation for densest k-subgraph. *CoRR*, abs/1001.2891, 2010.

[Boy74] David W. Boyd. The power method for p norms. *Linear Algebra and its Applications*, 9:95 – 101, 1974.

[CW04] Moses Charikar and Anthony Wirth. Maximizing quadratic programs: Extending grothendieck's inequality. *Foundations of Computer Science, Annual IEEE Symposium on*, 0:54–60, 2004.

[DVTV09] Amit Deshpande, Kasturi R. Varadarajan, Madhur Tulsiani, and Nisheeth K. Vishnoi. Algorithms and hardness for subspace approximation. *CoRR*, abs/0912.1403, 2009.

[ER09] Matthias Englert and Harald Räcke. Oblivious routing in the l_p-norm. In *Proc. of the 50th FOCS*, 2009.

[GHR06] Anupam Gupta, Mohammad T. Hajiaghayi, and Harald Räcke. Oblivious network design. In *SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 970–979, New York, NY, USA, 2006. ACM.

[GLS88] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, New York, 1988.

[Gro53] Alexander Grothendieck. Resume de la theorie metrique des produits tensoriels topologiques. *Bol. Soc. Mat. Sao Paulo*, 8:1–79, 1953.

[Hås01] Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.

[Hig92] Nicholas J. Higham. Estimating the matrix p-norm. *Numer. Math*, 62:511–538, 1992.

[HJ85] Roger A. Horn and Charles R. Johnson. *Matrix analysis / Roger A. Horn, Charles R. Johnson*. Cambridge University Press, Cambridge [Cambridgeshire] ; New York, 1985.

[HO09] Julien M. Hendrickx and Alex Olshevsky. Matrix p-norms are np-hard to approximate if p neq 1,2,infty. *CoRR*, abs/0908.1397, 2009.

[Kho04] Subhash Khot. Hardness of approximating the shortest vector problem in lattices. *Foundations of Computer Science, Annual IEEE Symposium on*, 0:126–135, 2004.

[KKL88] J. Kahn, G. Kalai, and N. Linial. The influence of variables on boolean functions. In *SFCS '88: Proceedings of the 29th Annual Symposium on Foundations of Computer Science*, pages 68–80, Washington, DC, USA, 1988. IEEE Computer Society.

[KNS08] Guy Kindler, Assaf Naor, and Gideon Schechtman. The ugc hardness threshold of the ℓ_p grothendieck problem. In *SODA '08: Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 64–73, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics.

[KV05] Subhash A. Khot and Nisheeth K. Vishnoi. The unique games conjecture, integrality gap for cut problems and embeddability of negative type metrics into ℓ_1. In *FOCS '05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 53–62, Washington, DC, USA, 2005. IEEE Computer Society.

[Nes98] Yurii Nesterov. Semidefinite relaxation and nonconvex quadratic optimization. *Optimization Methods and Software*, 9:141–160, 1998.

[RÖ8] Harald Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *STOC '08: Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 255–264, New York, NY, USA, 2008. ACM.

[Reg06] Oded Regev. Lattice-based cryptography. In *In Proc. of the 26th Annual International Cryptology Conference (CRYPTO*, pages 131–141, 2006.

[RS10] Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In *STOC '10: Proceedings of ACM symposium on Theory of Computing*, 2010.

[Ste05] Daureen Steinberg. Computation of matrix norms with applications to robust optimization. Research thesis, Technion - Israel University of Technology, 2005.

A Miscellany

A.1 Non-convex optimization

Note that computing the $p \mapsto p$ norm is in general not a convex optimization problem. i.e., the function f defined by $f(x) = \frac{\|Ax\|_p}{\|x\|_p}$ is not in general concave. For example, consider

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 1 \end{pmatrix}; \quad \mathbf{x} = \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix}; \quad \mathbf{y} = \begin{pmatrix} 0.2 \\ 0.5 \end{pmatrix}$$

In this case, with $p = 2.5$, for instance, it is easy to check that $f((\mathbf{x}+\mathbf{y})/2) < (f(\mathbf{x}) + f(\mathbf{y}))/2$. Thus f is not concave. However, it could still be that f raised to a certain power is concave.

A.2 Duality

The following equality is useful in 'moving' from one range of parameters to another. We use the fact that $\|u\|_p = \max_{y \,:\, \|y\|_{p'}=1} y^T x$, where p' is the 'dual norm', satisfying $1/p+1/p' = 1$. (similarly q' denotes the dual norm of q)

$$\|A\|_{q\mapsto p} = \max_{\|x\|_q=1} \|Ax\|_p = \max_{\substack{\|x\|_q=1 \\ \|y\|_{p'}=1}} y^T A x = \max_{\substack{\|x\|_q=1 \\ \|y\|_{p'}=1}} x^T A^T y = \|A^T\|_{p'\mapsto q'} \tag{26}$$

A.3 Moving to a positive matrix

We now show that by adding a very small positive number to each entry of the matrix, the $q \mapsto p$-norm does not change much.

Lemma A.1. *Let A be an $n \times n$ matrix where the maximum entry is scaled to 1. Let J_ϵ be the matrix with all entries being ϵ.*

$$\|A + J_\varepsilon\|_{q\mapsto p} \leq \|A\|_{q\mapsto p}\big(1 + \epsilon n^{1+\frac{1}{p}-\frac{1}{q}}\big)$$

Proof. We first note that $\|A\|_{q\mapsto p} \geq 1$ (because the maximum entry is 1). It is also easy to see that J_ε is maximized by the vector with all equal entries. Hence $\|J_\varepsilon\|_{q\mapsto p} \leq n^{1+\frac{1}{p}-\frac{1}{q}}\epsilon$. Hence, by using the fact that $\|\cdot\|_{q\mapsto p}$ is a norm, the lemma follows. □