ABSTRACT

WENTWORTH, THOMAS ALLEN. Leverage Scores: Sensitivity and Applications to Randomized Algorithms.(Under the direction of Ilse Ipsen.)

In this thesis, we present various results pertaining to a matrix property called *leverage scores* and their application to randomized row sampling. We begin by investigating three uniform strategies for randomized row sampling from matrices with orthonormal columns (without replacement, with replacement, and Bernoulli sampling). Our analysis is focused on the two-norm condition number of the sampled matrices due to it's applications to the generation of efficient preconditioners for the randomized least squares solver *Blendenpik*. As part of our analysis, we present probabilistic bounds on the condition number of the sampled matrix in terms of both leverage scores and coherence (the largest leverage score). We also develop algorithms for generating test matrices with specified leverage scores.

Next, we derive leverage score perturbation bounds. These bounds show that the leverage scores of the perturbed matrix are close to the leverage scores of the original matrix if the two norm of the perturbation and the two norm of the left pseudoinverse of the original matrix are small. We also bound the change in the leverage scores in terms of the principal angles between the original matrix and the perturbed matrix.

Finally, we present kappa_SQ, a `Matlab` software package and GUI designed to run experiments on the two-norm condition number of a sampled matrix and produce paper-ready plots.

Leverage Scores: Sensitivity and Applications to Randomized Algorithms

by
Thomas Allen Wentworth

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Applied Mathematics

Raleigh, North Carolina

2014

APPROVED BY:

Petros Drineas

Jonathan Hauenstein

Stephen Campbell

Ilse Ipsen
Chair of Advisory Committee

DEDICATION

To my wife.

BIOGRAPHY

Thomas Allen Wentworth was born in Stoneham, Massachusetts on December 12$^{\text{th}}$, 1985. From an early age, he showed a fascination with all things technical and scientific and would barrage his parents, Bruce and Diana, with countless questions. This inquisitive nature flourished into a love for all science, and in 2004, Thomas began his undergraduate studies at Rensselaer Polytechnic Institute as a physics major. After his freshman year, having completed nearly all of his required math courses, he added mathematics as a second major. By the end of his junior year Thomas' love and appreciation for the intensely logical approach to problem solving in mathematics had convinced him to change his plans and apply to graduate programs in mathematics.

It was around this time that Thomas met his future wife Mami. She was also a dual math and physics major and the classes that Thomas and Mami shared afforded them with an ample amount of study time in which to get to know each other. They both graduated from Rensselaer Polytechnic Institute in 2008 and within 6 months were engaged to be married.

In August of 2008, Thomas began his graduate career in Applied Mathematics at North Carolina State University. In December of 2011 he received his Masters of Science in Applied Mathematics. He received a SAMSI fellowship in August of 2013. On October 7$^{\text{th}}$, 2013, Thomas and Mami had their first child, Koki. Thomas finished his doctorate in 2014.

ACKNOWLEDGEMENTS

I would like to extend my most sincere thanks to those who have helped me along my path to completing my dissertation.

First and foremost, I would like to start by thanking *Mami Wentworth*, my lovely wife. It is her love and support, both emotional and mathematical, that made this dissertation possible.

I want to thank *Ilse Ipsen*, my advisor, for her patience and dedication to my educational success. It is her efforts that have helped shape me from a student into a professional mathematician. Additionally, a significant portion of the work in this thesis is due to Ilse and I would also like to thank her for her contributions.

I would also like to thank my other committee members, *Steve Campbell*, *Jonathan Hauenstein*, *Petros Drineas* and *John Morillo* as well as *Michael Mahoney* for their time, comments and questions.

I want to thank John Holodnak for finding errors in one of the papers that comprises this thesis.

Finally, but not least, I would like to thank *Denise Seabrooks* for her invaluable help with the administrative side of my graduate career.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

The leverage scores of a $m \times n$ matrix A with full column rank are defined as follows:

$$\ell_i(A) = \|e_i^T Q\|_2^2, \text{ for } 1 \le i \le m,$$

where Q is any basis of orthonormal columns for the column space of A. Conceptually, the leverage scores give a "measure" of the relative importance of each row where the meaning of the word 'importance' depends on the application. They were first described in [20] where the authors used leverage scores as a regression diagnostic for least squares to help identify potential outliers in the data. In the paper, they describe how the magnitude of the i^{th} leverage score gives a measure of the influence of the i^{th} row on the least squares fit. If the i^{th} leverage score is large, and one questions the accuracy of the corresponding data point, then one may want to remove the data point in order to better fit the rest of the data. Leverage scores are also used in importance sampling for low rank matrix approximation. In [26, 14, 31], leverage scores are used to identify important rows and columns [1] which are then used (or sampled with higher probability) to approximate the matrix. Leverage scores have also been used in the following [25, 13, 14, 6, 35, 12, 7].

In Chapter 2, motivated by the least squares solver *Blendenpik*[1], we investigate three strategies for uniform sampling of rows from matrices Q with orthonormal columns. The goal is to determine, with high probability, how many rows are required so that the sampled matrices have full rank and are well-conditioned with respect to inversion. Extensive numerical experiments illustrate that the three sampling strategies (without replacement, with replacement, and Bernoulli sampling) behave almost identically, for small to moderate amounts of sampling. In particular, sampled matrices of full rank tend to have two-norm condition numbers of at

[1] There are more general definitions of leverage scores than what we examine in this dissertation. Leverage scores for the columns of a matrix can be defined as the row norms squared of the right singular vector matrix. For more details, see [39, Eqn. 3], [31], [26, Eqn. 3].

most 10.

We derive a bound on the condition number of the sampled matrices in terms of the coherence μ of Q where the coherence is simply the largest leverage score. This bound applies to all three different sampling strategies; it implies a, not necessarily tight, lower bound of $\mathcal{O}\left(m\mu \ln n\right)$ for the number of sampled rows; and it is realistic and informative even for matrices of small dimension and the stringent requirement of a 99 percent success probability.

For uniform sampling with replacement we derive a potentially tighter condition number bound in terms of the leverage scores of Q. To obtain a more easily computable version of this bound, in terms of just the largest leverage scores, we first derive a general bound on the two-norm of diagonally scaled matrices.

To facilitate the numerical experiments and test the tightness of the bounds, we present deterministic algorithms to generate matrices with user-specified coherence and leverage scores.

In Chapter 3, we derive bounds for the leverage scores as well as leverage scores computed from the top k left singular vectors. Our bounds show that if the principal angles between A and B are small, then the leverage scores of B are close to the leverage scores of A. Next, we use two results by Wedin [41, 40] to bound the principal angles from above in terms of the singular values of A and $\|B-A\|_2$. Finally, we combine these results and derive bounds for the leverage scores of B in terms of the singular values of A and $\|B-A\|_2$ and show that if $\|B-A\|_2$ and $\sigma_n(A)^{-1}$ are small, then the leverage scores of B are close to the leverage scores of A. Additionally, we show that if $\|B-A\|_2$ is small with respect to the k^{th} singular value gap, then the leverage scores of A and B, as computed from the top k left singular vectors, are close.

In Chapter 4, we present kappa_SQ, a `Matlab` software package and GUI designed to run experiments on the two-norm condition number of a sampled matrix, $\kappa(SQ)$, where S is a random row sampling matrix and produce paper-ready plots. Via a simple GUI, kappa_SQ can generate test matrices, perform various types of row sampling, measure the condition number of the sampled matrix, calculate bounds and produce high quality plots of the results. All of the important codes are written in separate `Matlab` function files in a standard format which makes it easy for a user to either use the codes by themselves or incorporate their own codes into the kappa_SQ package.

Chapter 2

The Effect of Coherence on Sampling From Matrices With Orthonormal Columns, and Preconditioned Least Squares Problems

2.1 Introduction

This chapter was inspired by Avron, Maymounkov and Toledo's *Blendenpik* algorithm and analysis [1].

Blendenpik is an iterative method for solving overdetermined least squares/regression problems $\min_x \|Ax - b\|_2$ with the Krylov space method *LSQR* [30]. In order to accelerate convergence, *Blendenpik* constructs a preconditioner R_s and solves instead the preconditioned least squares problem $\min_z \|AR_s^{-1}z - b\|_2$. The solution to the original problem is recovered by solving a linear system with coefficient matrix R_s. The innovative feature is the construction of the preconditioner R_s by a random sampling method.

2.1.1 Motivation

The purpose of this chapter is a thorough experimental and analytical investigation of random sampling strategies for producing efficient preconditioners. The challenge is to ensure not only that R_s is nonsingular, but also that AR_s^{-1} is well-conditioned with respect to inversion, which is required for fast convergence and numerical stability.

Here is a conceptual point of view of how *Blendenpik* constructs the preconditioner: First it "smoothes out" the rows of A by applying a randomized unitary transform F, and then it uniformly samples (i.e. selects) a small number of rows M_s from FA. At last it computes a QR factorization of the smaller sampled matrix, $M_s = Q_s R_s$, where the triangular factor R_s serves as the preconditioner.

The neat and crucial observation in [1] is to realize that sampling rows from FA amounts, conceptually, to sampling rows from an orthonormal basis of FA. That is, if the columns of Q represent an orthonormal basis for the column space of FA, and if S is a sampling matrix then SQ has the same two-norm condition number as AR_s^{-1}. This means, it suffices to consider sampling from matrices Q with orthonormal columns.

The analysis in [1] suggests that SQ is well conditioned, if Q has low "coherence". Intuitively, coherence gives information about the localization or "uniformity" of the elements of Q. Mathematically, coherence is the largest (squared) norm of any row of Q. For instance, if Q consists of canonical vectors, then the non-zero elements are concentrated in only a few rows, so that Q has high coherence. However, if Q is a submatrix of a Hadamard matrix, then all elements have the same magnitude, so that Q has low coherence.

If Q has low coherence, then, in the context of sampling, all rows are equally important. Hence any sampled matrix SQ with sufficiently many rows is likely to have full rank. The purpose of the randomized transform F is to produce a matrix FA whose orthonormal basis Q has low coherence.

We were intrigued by the analysis of *Blendenpik* because it appears to be the first to exploit the concept of coherence for numerical purposes. We also wanted to get a better understanding of the condition number bound for SQ in [1, Theorem 3.2], which contains an unspecified constant, and of the effect of uniform sampling strategies.

2.1.2 Overview and main results

We survey the contents of the chapter , with a focus on the main results.

From preconditioned matrices to sampled matrices with orthonormal columns (Section 2.2)

We start with a brief sketch of the *Blendenpik* least squares solver (Section 2.2.1), and make the important transition from preconditioned matrices AR_s^{-1} to sampled matrices SQ with orthonormal columns, made possible by the observation ([1, 33] and Lemma 1) that both have the same two-norm condition number[1],

$$\kappa(AR_s^{-1}) = \kappa(SQ).$$

Then we discuss the notion of coherence and its properties (Section 2.2.2). For a $m \times n$ matrix Q with orthonormal columns, $Q^TQ = I_n$, the *coherence*

$$\mu \equiv \max_{1 \leq j \leq m} \|e_j^T Q\|_2^2$$

is the largest squared row norm[2].

Sampling methods (Section 2.3)

We discuss three randomized methods for producing sampling matrices S: Sampling without replacement (Section 2.3.1), sampling with replacement (Section 2.3.2), and Bernoulli sampling (Section 2.3.3). We show that Bernoulli sampling can be viewed as a form of sampling without replacement (Section 2.3.4).

The sampling matrices S from all three methods are constructed so that S^TS is an unbiased estimator of the identity matrix. The action of applying S to a matrix Q with orthonormal columns, SQ, amounts to randomly sampling rows from Q.

The numerical experiments (Section 2.3.5) illustrate two points: First, the three sampling methods behave almost identically, in terms of the percentage of sampled matrices SQ that have full rank and their condition numbers, in particular for small to moderate sampling amounts. Second, those sampled matrices SQ that have full rank tend to be very well-conditioned, with condition numbers $\kappa(SQ) \leq 10$.

As a consequence (Section 2.3.6), we recommend sampling with replacement for *Blendenpik*, because it is fast, and it is easy to implement.

Numerical experiments

Since random sampling methods can be expected to work well in the asymptotic regime of very large matrix dimensions, we restrict all numerical experiments to matrices of small dimension.

Furthermore, we consider only matrices that have many more rows than columns, $m \gg n$. This is the situation where random sampling methods can be most efficient. In contrast, random sampling methods are not efficient for matrices that are almost square, because the number of rows in SQ has to be at least equal to n, otherwise $\text{rank}(SQ) = n$ is not possible.

[1] Here $\kappa(X) \equiv \|X\|_2 \, \|X^\dagger\|_2$ denotes the Euclidean two-norm condition number with respect to inversion of a full rank matrix X. The matrix $X^\dagger$ is the Moore-Penrose inverse of X.

[2] The superscript T denotes transpose, and I_n is the $n \times n$ identity matrix with columns e_j.

Condition number bounds based on coherence (Section 2.4)

We derive a probabilistic bound, in terms of coherence, for the condition numbers of the sampled matrices (Theorem 7 in Section 2.4.1). The bound applies to all three sampling methods. From this we derive the following lower bound, not necessarily tight, on the required number of sampled rows.

Preview of Corollary 8 *Given a failure probability* $0 < \delta < 1$, *and a tolerance* $0 \leq \epsilon < 1$. *To achieve the condition number bound* $\kappa(SQ) \leq \sqrt{\frac{1+\epsilon}{1-\epsilon}}$, *the number of rows from* Q, *sampled by any of three methods, should be at least*

$$c \geq 3m\mu \, \frac{\ln(2n/\delta)}{\epsilon^2}. \tag{2.1}$$

This suggests that one has to sample more rows for SQ if Q has high coherence (μ close to 1), if one wants a low condition number bound (small ϵ), or if one wants a high success probability (small δ).

Numerical experiments (Section 2.4.2) illustrate that the bounds are informative for matrices with sufficiently low coherence μ and sufficiently high aspect ratio m/n. Our bounds have the following advantages (Section 2.4.3):

1. They are tighter than those in [1, Theorem 3.2] because they are non-asymptotic, with all constants explicitly specified.

2. They apply to three different sampling methods.

3. They imply a lower bound, of $\Omega\left(m\mu \ln n\right)$, on the required number of sampled rows.

4. They are realistic and informative – even for matrices of small dimension and the stringent requirement of a 99 percent success probability.

Condition number bounds based on leverage scores, for uniform sampling with replacement (Section 2.5)

The goal is to tighten the coherence-based bounds from Section 2.4 by making use of all the row norms of Q, instead of just the largest one. To this end we introduce *leverage scores* (Section 2.5.1), which are the squared row norms of Q,

$$\ell_j = \|e_j^T Q\|_2^2, \qquad 1 \leq j \leq m.$$

We use them to derive a bound for uniform sampling with replacement (Theorem 11 in Section 2.5.2). Then we present a more easily computable bound, in terms of just a few of the largest leverage scores (Section 2.5.3). It implies the following lower bound, not necessarily tight, on the number of samples.

PREVIEW OF COROLLARY 15 *Given a failure probability* $0 < \delta < 1$*, a tolerance* $0 \leq \epsilon < 1$*, and a labeling of leverage scores in non-increasing order,*

$$\mu = \ell_{[1]} \geq \cdots \geq \ell_{[m]}.$$

To achieve the condition number bound $\kappa(SQ) \leq \sqrt{\frac{1+\epsilon}{1-\epsilon}}$*, the number of rows from* Q*, sampled uniformly with replacement, should be at least*

$$c \geq \tfrac{2}{3} m \, (3\tau + \epsilon\mu) \, \frac{\ln(2n/\delta)}{\epsilon^2}, \tag{2.2}$$

where $t \equiv \lfloor 1/\mu \rfloor$ *and* $\tau \equiv \mu \, \sum_{j=1}^{t} \ell_{[j]} + (1 - t\,\mu)\, \ell_{[t+1]}$*.*

We show (Section 2.5.4) that (2.2) is indeed tighter than (2.1). This is confirmed by numerical experiments (Section 2.5.5). The difference becomes more drastic for matrices Q with widely varying non-zero leverage scores, and can be as high as ten percent. Hence (Section 2.5.6), when it comes to lower bounds for the number of rows sampled uniformly with replacement, we recommend (2.2) over (2.1).

Algorithms for generating matrices with prescribed coherence and leverage scores (Section 2.6)

The purpose is to make it easy to investigate the efficiency of the sampling methods in Section 2.3, and test the tightness of the bounds in Sections 2.4 and 2.5.

To this end we present algorithms for generating matrices with prescribed leverage scores and coherence (Section 2.6.1), and for generating particular leverage score distributions with prescribed coherence (Section 2.6.2). Furthermore we present two classes of structured matrices with prescribed coherence that are easy and fast to generate (Section 2.6.3). The basis for the algorithms is the following majorization result.

PREVIEW OF THEOREM 25 *Given integers* $m \geq n$ *and a vector* ℓ *with* m *elements that satisfy* $0 \leq \ell_j \leq 1$ *and* $\sum_{j=1}^{m} \ell_j = n$*, there exists a* $m \times n$ *matrix* Q *with orthonormal columns that has leverage scores* $\|e_j^T Q\|_2^2 = \ell_j$*,* $1 \leq j \leq m$*, and coherence* $\mu = \max_{1 \leq j \leq m} \ell_j$*.*

Bound for two-norms of diagonally scaled matrices (Section 2.8)

The bound (2.2) is based on a special case of the following general bound for the two-norm of diagonally scaled matrices.

Preview of Theorem 22 *Let Z be a $m \times n$ matrix with* $\operatorname{rank}(Z) = n$ *and largest squared row norm* $\mu_z \equiv \max_{1 \le j \le m} \|e_j^T Z\|_2^2$*. Let D be a $m \times m$ non-negative diagonal matrix, and a labeling of diagonal elements in non-increasing order,*

$$\|D\|_2 = d_{[1]} \geq \cdots \geq d_{[m]} \geq 0.$$

If $t \equiv \left\lfloor (\|Z^\dagger\|_2^2 \, \mu_z)^{-1} \right\rfloor$*, then either*

$$\|DZ\|_2^2 \leq \mu_z \sum_{j=1}^{t} d_{[j]}^2 + \left(\|Z\|_2^2 - t\,\mu_z\right) \, d_{[t+1]}^2 \qquad \text{if } \|Z\|_2^2 - t\,\mu_z \leq \mu_z$$

or

$$\|DZ\|_2^2 \leq \mu_z \sum_{j=2}^{t+1} d_{[j]}^2 + \left(\|Z\|_2^2 - t\,\mu_z\right) \, d_{[1]}^2. \qquad \text{if } \|Z\|_2^2 - t\,\mu_z > \mu_z.$$

Proofs (Sections 2.7, 2.8 and 2.9)

All proofs, except those for Sections 2.2 and 2.3, have been relegated to these three sections.

Section 2.7 contains the proofs for Sections 2.4 and 2.5, which are based on two matrix concentration inequalities: A Chernoff bound (Section 2.7.1), and a Bernstein bound (Section 2.7.4).

Section 2.8 contains the proofs for the easily computable bounds in Sections 2.5.3 and 2.5.4, together with the majorization results (Section 2.8.1) required for the proofs.

The majorization results in Section 2.9 represent the foundation for the algorithms in Section 2.6.

2.1.3 Literature

Existing randomized least squares methods are based on randomized projections. This means, conceptually they multiply A by a random matrix F, and then sample a few rows from FA.

The algorithms in [4, 12, 15] solve a smaller sampled problem by a direct method. Like *Blendenpik* [1], the algorithm in [33] computes a preconditioner from the QR factorization of a sampled submatrix, but then solves the preconditioned problem by applying the conjugate gradient method to the normal equations. The parallel solver *LSRN* [27] computes a preconditioner from the SVD of a sampled submatrix, and then solves the preconditioned problem with an iterative method. This solver applies to general matrices rather than just those of full column rank.

As for randomized algorithms in general, the excellent surveys [19, 25] provide clear analyses and good intuition.

2.1.4 Notation

The norm $\|\cdot\|_2$ denotes the Euclidean two-norm, and the two-norm *condition number* with respect to inversion of a real $m \times n$ matrix Z with $\operatorname{rank}(Z) = n$ is denoted by $\kappa(Z) \equiv \|Z\|_2 \|Z^\dagger\|_2$, where $Z^\dagger$ is the Moore-Penrose inverse. The $k \times k$ *identity matrix* is $I_k = \begin{pmatrix} e_1 & \dots & e_k \end{pmatrix}$, and its columns are the *canonical vectors* e_j, $1 \leq j \leq k$.

The *probability* of an event $\mathcal{X}$ is denoted by $\mathbf{Pr}[\mathcal{X}]$, and the *expected value* of a random variable X is denoted by $\mathbf{E}[X]$.

2.2 The Blendenpik algorithm, and coherence

We describe the *Blendenpik* algorithm for solving least squares problems (Section 2.2.1), and present the notion of coherence (Section 2.2.2).

2.2.1 Algorithm

The *Blendenpik* algorithm [1, Algorithm 1] solves full column rank least squares problems with the Krylov space method *LSQR* [30] and a randomized preconditioner. Algorithm 2.2.1 presents a conceptual sketch of *Blendenpik*. The subscript "s" denotes quantities associated with the sampled matrix.

Algorithm 2.2.1 Sketch of *Blendenpik* [1]

```
Input: $m \times n$ matrix $A$ with $m \geq n$ and $\operatorname{rank}(A) = n$, $m \times 1$ vector $b$
       $m \times m$ random unitary matrix $F$
       $k \times n$ sampling matrix $S$ with $k \geq n$
Output: Solution of $\min_x \|Ax - b\|_2$

  $M = FA$          {Improve coherence}
  $M_s = SM$          {Sample for preconditioner}
  Thin QR factorization $M_s = Q_s R_s$          {Generate preconditioner}
  Determine solution $y$ to $\min_z \|AR_s^{-1}z - b\|_2$          {Solve preconditioned problem}
  Solve $R_s \hat{x} = y$          {Recover solution to original problem}
```

The matrix F is the product of a random diagonal matrix with ±1 entries, and a unitary transform, such as a Walsh Hadamard transform, or a discrete Fourier, Hartley or cosine transform [1, Section 3.2]. The transformed matrix $M = FA$ is $m \times n$ with $m \geq n$ and $\operatorname{rank}(M) = n$.

The sampling matrix S selects $k \geq n$ rows from the transformed matrix M. We discuss different types of sampling matrices in Section 2.3. The $k \times n$ sampled matrix M_s has a thin QR decomposition $M_s = Q_s R_s$ where Q_s is $k \times n$ with orthonormal columns and R_s is $n \times n$ upper triangular.

The basis for the analysis is the thin QR decomposition $M = QR$, where Q is $m \times n$ with orthonormal columns and R is $n \times n$ upper triangular. This QR decomposition is *not* computed. The next result links the condition number of the preconditioned matrix to that of the matrix SQ, see also [1, Section 3.1] and [33, Theorem 1].

Lemma 1 *With the notation in Algorithm 2.2.1, if* $\operatorname{rank}(M_s) = n$*, then*

$$\kappa(AR_s^{-1}) = \kappa(SQ).$$

Proof. From $FA = M = QR$ and the fact that the 2-norm is invariant under premultiplication by matrices with orthonormal columns, it follows that

$$\begin{aligned} \kappa(AR_s^{-1}) &= \kappa(MR_s^{-1}) = \kappa(RR_s^{-1}) = \kappa(R_s R^{-1}) = \kappa(M_s R^{-1}) = \kappa(SMR^{-1}) \\ &= \kappa(SQ). \end{aligned}$$

$\square$

In Sections 2.4 and 2.5 we derive bounds for the condition number of the preconditioned matrix, $\kappa(AR_s^{-1})$. Our bounds are tighter than those in [1, Theorem 3.2], because they have all constants explicitly specified, and apply to three different sampling strategies. Since Lemma 1 implies $\kappa(AR_s^{-1}) = \kappa(SQ)$, we state the bounds for $\kappa(SQ)$ only. An important ingredient in these bounds is the coherence of Q.

2.2.2 Coherence

Coherence gives information about the localization or "uniformity" of the elements in an orthonormal basis. The more general concept of *mutual coherence* between two orthonormal bases was introduced in [10, §VII], in the context of signal processing and computational harmonic analysis, to describe a condition for the existence of sparse representations of signals. What we use here is a special case, and can be viewed as a measure for how close an orthonormal basis is to sharing a vector with a canonical basis.

Definition 2 (Definition 3.1 in [1], Definition 1.2 in [7]) *Let Q be a real $m \times n$ matrix with orthonormal columns, $Q^T Q = I_n$, then the* coherence *of Q is*

$$\mu \equiv \max_{1 \leq j \leq m} \|e_j^T Q\|_2^2.$$

If the columns of Q are an orthonormal basis for the column space of a matrix M, then the coherence of M is μ.

The second part of Definition 41 emphasizes that coherence is really a property of the column space, hence basis-independent. In other words, if $\hat{Q} = QV$, where V is a real $n \times n$ orthogonal matrix, then $\hat{Q}$ and Q have the same coherence.

The range for coherence is $\frac{n}{m} \leq \mu \leq 1$. If Q is a $m \times n$ submatrix of the $m \times m$ Hadamard matrix, then $\mu = n/m$. If a column of Q is a canonical vector, then $\mu = 1$. Hence an orthonormal basis has high coherence if it shares a vector with a canonical basis.

There are other definitions of coherence that differ from the above by factors depending on the matrix dimensions [32, Definition 1], [35, Definition 1]. However, the notion of *statistical coherence* in Bayesian analysis [24] appears to be unrelated.

2.3 Sampling Methods

We present three different types of sampling methods: Sampling without replacement (Section 2.3.1), sampling with replacement (Section 2.3.2), and Bernoulli sampling (Section 2.3.3). We show that Bernoulli sampling can be viewed as a form of sampling without replacement (Section 2.3.4). The numerical experiments illustrate that there is little difference among the three methods for small to moderate amounts of sampling (Section 2.3.5). Hence we recommend sampling with replacement for Algorithm 2.2.1 (Section 2.3.6).

The sampling matrices S in all three methods are scaled so that $S^T S$ is an unbiased estimator of the identity matrix.

2.3.1 Sampling without replacement

The obvious sampling strategy, in Algorithm 2.3.1, picks the requested number of rows, so that the sampling matrix S is just a scaled submatrix of a permutation matrix.

Uniform sampling without replacement can be implemented via *random permutations*[3]. A permutation $\pi_1, \ldots, \pi_m$ of the integers $1, \ldots, m$ is a *random permutation*, if it is equally likely to be one of $m!$ possible permutations [29, pages 41 and 48].

The following lemma presents the probability that sampling without replacement picks a particular row.

Lemma 3 *If Algorithm 2.3.1 samples c out of m indices, then the probability that a particular index is picked equals c/m.*

[3] We thank an anonymous reviewer for this advice.

Algorithm 2.3.1 Uniform sampling without replacement [16, 18]

Input: Integers $m \geq 1$ and $1 \leq c \leq m$
Output: $c \times m$ sampling matrix S with $\mathbf{E}[S^T S] = I_m$

```
  Let $k_1, \ldots, k_m$ be a random permutation of $1, \ldots, m$
  $S = \sqrt{\frac{m}{c}} \begin{pmatrix} e_{k_1} & \ldots & e_{k_c} \end{pmatrix}^T$
```

Proof. The probability that some index, say r, is not sampled in the first trial is $1-\frac{1}{m} = \frac{m-1}{m}$. Now there are only $m-1$ indices left. So the probability that index r is not sampled in the second trial is $1-\frac{1}{m-1} = \frac{m-2}{m-1}$. Repeating this argument shows that with probability $\prod_{t=1}^{c} \frac{m-t}{m-t+1} = \frac{m-c}{m}$ index r is not sampled in c trials.

The complementary event, the probability that index r is sampled, equals $1 - \frac{m-c}{m} = \frac{c}{m}$. □

2.3.2 Sampling with replacement

This is the sampling strategy that appears to be analyzed in [1]. It samples exactly the requested number of rows, but with replacement, which means a row may be sampled more than once. Algorithm 2.3.2 is the same as the *EXACTLY(c)* algorithm [15, Algorithm 3] with uniform probabilities, which is also used in the *BasicMatrixMultiplication Algorithm* [11, Fig. 2].

Algorithm 2.3.2 Uniform sampling with replacement [11, 15]

Input: Integers $m \geq 1$ and $1 \leq c \leq m$
Output: $c \times m$ sampling matrix S with $\mathbf{E}[S^T S] = I_m$

```
  for $t = 1 : c$ do
    Sample $k_t$ from $\{1, \ldots, m\}$ with probability $1/m$,
    independently and with replacement
  end for
  $S = \sqrt{\frac{m}{c}} \begin{pmatrix} e_{k_1} & \ldots & e_{k_c} \end{pmatrix}^T$
```

Sampling with replacement (Algorithm 2.3.2) is often easier to analyze and implement than sampling without replacement (Algorithm 2.3.1), and it can also be more robust to errors [29, §1.2].

2.3.3 Bernoulli sampling

The sampling strategy in Algorithm 2.3.3 is implemented in *Blendenpik* [1, Algorithm 1]. Following [18, Section A], we use the term *Bernoulli sampling*, because the strategy treats each row as an independent, identically distributed Bernoulli random variable. Each row is either sampled or not, with the same probability for each row. Algorithm 2.3.3 produces a $m \times m$ square matrix S – in contrast to Algorithms 2.3.1 and 2.3.2, which produce $c \times m$ matrices.

```
Algorithm 2.3.3 Bernoulli sampling [1, 16, 18]
Input: Integers $m \geq 1$ and $1 \leq c \leq m$
Output: $m \times m$ sampling matrix $S$ with $\mathbf{E}[S^T S] = I_m$

  $S = 0_{m \times m}$
  for $t = 1 : m$ do
      $S_{tt} = \sqrt{\frac{m}{c}} \begin{cases} 1 & \text{with probability } \frac{c}{m} \\ 0 & \text{with probability } 1 - \frac{c}{m} \end{cases}$
  end for
```

The number of sampled rows, which is equal to the number of non-zero diagonal elements in S, is not known a priori, but the expected number of sampled rows is c. The lemma below shows that the actual number of rows picked by Bernoulli sampling is characterized by a binomial distribution [34, Section 2.2.2].

Lemma 4 *If Algorithm 2.3.3 samples from m indices with probability $\gamma \equiv c/m$, then the probability that it picks exactly k indices equals $\binom{m}{k} \gamma^k (1-\gamma)^{m-k}$.*

Proof. Determining the diagonal elements of the $m \times m$ sampling matrix S in Algorithm 2.3.3 can be viewed as performing m independent trials, where trial t is a success ($S_{tt} \neq 0$) with probability γ, and a failure ($S_{tt} = 0$) with probability $1 - \gamma$. The probability of k successes is given by the binomial distribution $\binom{m}{k} \gamma^k (1-\gamma)^{m-k}$. □

2.3.4 Relating Bernoulli sampling and sampling without replacement

We show that Bernoulli sampling (Algorithm 2.3.3) is the same as first determining the number of samples with a binomial distribution (motivated by Lemma 4), and then sampling without replacement (Algorithm 2.3.1). This is described in Algorithm 2.3.4 below.

Below we describe the sense in which Algorithm 2.3.4 “behaves like” Bernoulli sampling in Algorithm 2.3.3.

Algorithm 2.3.4 Simulating Algorithm 2.3.3 with Algorithm 2.3.1

```
Input: Integers $m \geq 1$ and $1 \leq c \leq m$
Output: $\tilde{c} \times m$ sampling matrix $S$ with $\mathbf{E}[S^T S] = I_m$
        that "behaves like" a sampling matrix generated by Algorithm 2.3.3

  $\gamma \equiv c/m$
  Sample $\tilde{c}$ from $\{1, \ldots, m\}$ where $\mathbf{Pr}[\tilde{c} = k] = \binom{m}{k} \gamma^k (1-\gamma)^{m-k}$
  Use Algorithm 2.3.1 to sample $\tilde{c}$ indices $k_1, \ldots, k_{\tilde{c}}$ uniformly and without replacement
  $S = \sqrt{\frac{m}{\tilde{c}}} \begin{pmatrix} e_{k_1} & \ldots & e_{k_{\tilde{c}}} \end{pmatrix}^T$
```

Lemma 5 *The probability that Algorithm 2.3.4 picks a particular index equals $\gamma = c/m$.*

Proof. Motivated by Lemma 4, the actual number of samples k in Algorithm 2.3.4 is given by a binomial distribution. Once a specific k has emerged, one applies Lemma 3 to conclude that the probability that Algorithm 2.3.1 picks some index r is k/m.

Now the probability that Algorithm 2.3.4 picks some index r is obtained by conditioning [34, Section 3.5] on the number of samples, k, and equals

$$\begin{aligned}
&\sum_{k=0}^{m} \mathbf{Pr}\left[k \text{ indices sampled}\right] \mathbf{Pr}\left[\text{index } r \text{ sampled} \middle| \, k \text{ indices sampled}\right] \\
&\quad = \sum_{k=1}^{m} \binom{m}{k} \gamma^k \ (1-\gamma)^{m-k} \ \frac{k}{m} \\
&\quad = \gamma \sum_{k=0}^{m-1} \binom{m-1}{k} \gamma^k \, (1-\gamma)^{m-1-k} = \gamma \left(\gamma + (1-\gamma)\right)^{m-1} = \gamma,
\end{aligned}$$

where the first equality follows from the zero summand for $k = 0$. □

Finally, we can conclude that sampling with Algorithm 2.3.4 is the same as sampling with Algorithm 2.3.3.

Theorem 6 *Both, Algorithms 2.3.4 and 2.3.3 pick a particular set of indices $i_1, \ldots, i_c$ with probability $\gamma^c (1-\gamma)^{m-c}$.*

Proof. The probability that Algorithm 2.3.3 samples indices $i_1, \ldots, i_c$ is equal to $\gamma^c (1-\gamma)^{m-c}$.

We show that the same is true for Algorithm 2.3.4. The choice of the sampling distribution in Algorithm 2.3.4 implies that it samples $\tilde{c} = c$ indices with probability $\binom{m}{c} \gamma^c \, (1-\gamma)^{m-c}$. Since there are $\binom{m}{c}$ ways to sample c out of m indices, the probbility that the particular index set $i_1, \ldots, i_c$ is picked, given that c indices are being sampled is $1/\binom{m}{c}$. Thus, the probability that Algorithm 2.3.4 picks indices $i_1, \ldots, i_c$ equals

$$\frac{1}{\binom{m}{c}} \binom{m}{c} \gamma^c (1-\gamma)^{m-c} = \gamma^c (1-\gamma)^{m-c}.$$

□

2.3.5 Numerical experiments

We present two representative comparisons of the three sampling strategies, with two plots for each strategy: The condition numbers of full-rank sampled matrices SQ, and the failure percentage, that is the percentage of sampled matrices SQ that are numerically rank deficient (as determined by the Matlab command `rank`).

The experiments are limited to very tall and skinny matrices (with many more rows than columns, $m \gg n$), because that's when the sampling strategies are most efficient. In particular, since $c \geq n$ is required for SQ to have full column rank, sampling methods are inefficient when n is not much smaller than m, in which case a deterministic algorithm would be preferable.

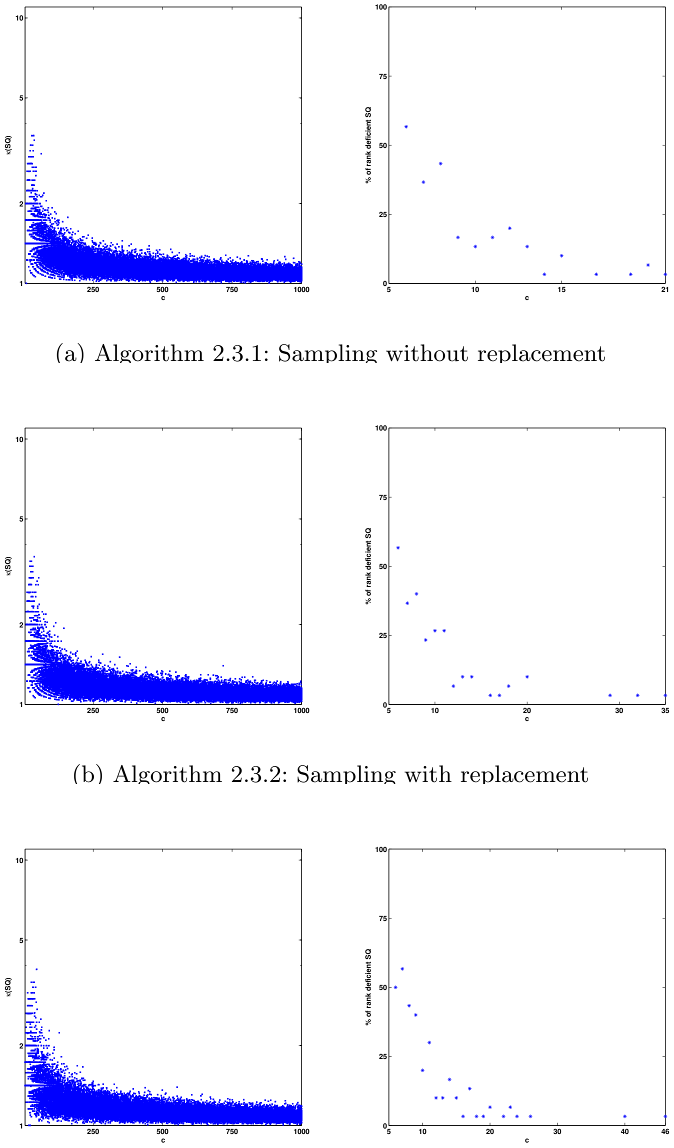
Experimental setup

The $m \times n$ matrices Q with orthonormal columns have $m = 10^4$ rows and $n = 5$ columns. The condition numbers and failure percentages are plotted against various sampling amounts c, with 30 runs for each c. For the failure percentages we display only those sampling amounts c that give rise to rank-deficient matrices, in these particular 30 runs. For Algorithm 2.3.3 the horizontal axis represents the numerator c in the probability, that is, the expected number of sampled rows. All three strategies sample from the same matrix.

We consider two different types of matrices: Matrices with low coherence $\mu = 1.5n/m$ in Figure 2.1; and matrices with higher coherence $\mu = 150n/m$ and many zero rows in Figure 2.2. Our numerical experiments indicate that these coherence values are representative, in the sense that different values of coherence would not produce any other interesting effects.

Figure 2.1

Shown are condition numbers and percentage of rank deficient matrices for a matrix Q with low coherence $\mu = 1.5n/m$ generated by Algorithm 2.6.2. At most 10 percent of the rows are sampled. The three strategies exhibit almost identical behavior: The sampled matrices SQ of full rank are very well conditioned, with $\kappa(SQ) \leq 5$. Numerically rank-deficient matrices SQ occur only for sampling amounts $c \leq 47$.

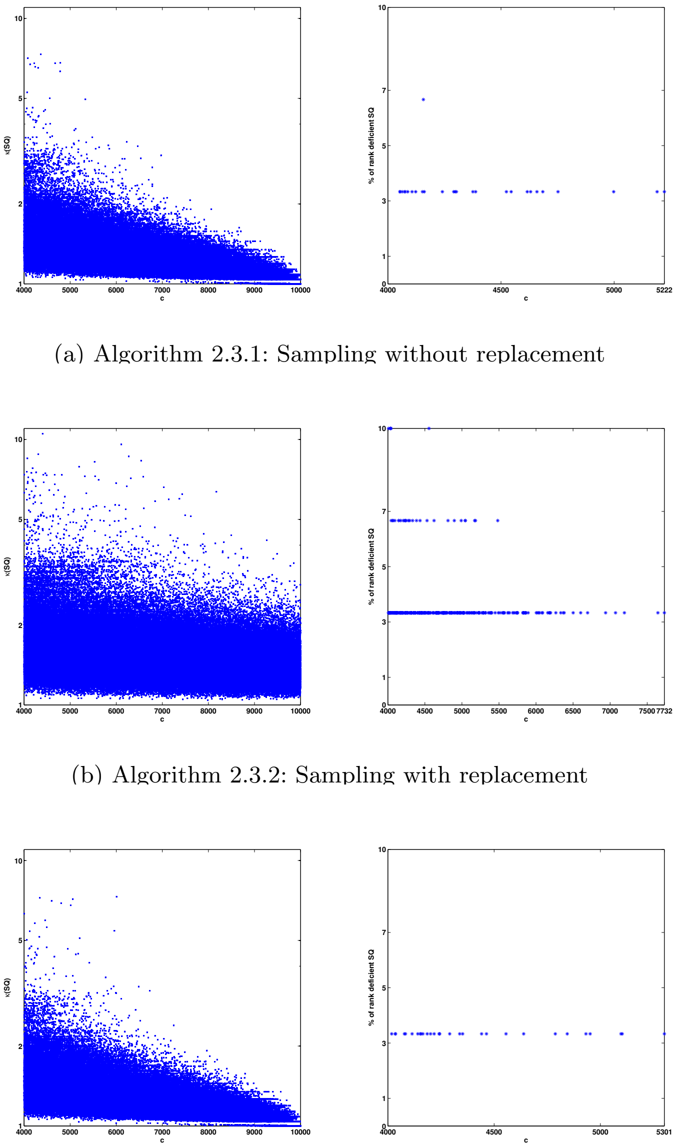


(a) Algorithm 2.3.1: Sampling without replacement

(b) Algorithm 2.3.2: Sampling with replacement

(c) Algorithm 2.3.3: Bernoulli sampling

Figure 2.1: Condition numbers and percentage of rank-deficiency for matrices with low coherence and small amounts of sampling. Here Q is $m \times n$ with orthonormal columns, $m = 10,000$, $n = 5$, coherence $\mu = 1.5n/m$, and generated with Algorithm 2.6.2. Left panels: Horizontal coordinate axes represent amounts of sampling $n \leq c \leq 1,000$. Vertical coordinate axes represent condition numbers $\kappa(SQ)$; the maximum is 10. Right panels: Horizontal coordinate axes represent amounts of sampling that give rise to numerically rank deficient matrices SQ. Vertical coordinate axes represent percentage of numerically rank deficient matrices.



(a) Algorithm 2.3.1: Sampling without replacement

(b) Algorithm 2.3.2: Sampling with replacement

(c) Algorithm 2.3.3: Bernoulli sampling

Figure 2.2: Condition numbers and percentage of rank-deficiency for matrices with higher coherence and large amounts of sampling. Here Q is $m \times n$ with orthonormal columns, $m = 10,000$, $n = 5$, coherence $\mu = 150n/m$, and generated with Algorithm 2.6.3. Left panels: Horizontal coordinate axes represent amounts of sampling $4,000 \leq c \leq m$. Vertical coordinate axes represent condition numbers $\kappa(SQ)$; the maximum is 10. Right panels: Horizontal coordinate axes represent amounts of sampling that give rise to numerically rank deficient matrices SQ. Vertical coordinate axes represent percentage of numerically rank deficient matrices SQ; the maximum is 10 percent.

Figure 2.2

Shown are condition numbers and percentage of rank deficient matrices for a matrix Q, generated by Algorithm 2.6.3, with coherence $150n/m$ and many zero rows. The number of sampled rows ranges from $c = 4000$ to m. The sampled matrices SQ of full rank are very well conditioned, with $\kappa(SQ) \leq 10$. Even for $c = 4,000$, as many 10 percent of the sampled matrices can still be rank-deficient. All three algorithms have to sample more than half of the rows of Q in order to always produce matrices SQ with full column rank. Specifically in these particular runs, Algorithms 2.3.1 and 2.3.3 need to sample $c \geq 5,222$ and $c \geq 5,301$ rows, respectively, while Algorithm 2.3.1 needs $c \geq 7732$.

Note that the condition numbers of matrices from Algorithms 2.3.1 and 2.3.3 approach 1 as more and more rows are sampled. This is because no row is sampled more than once; and for $c = m$ all rows are sampled.

Again, the three strategies exhibit almost identical behavior: The sampled matrices SQ of full rank are very well conditioned, with $\kappa(SQ) \leq 10$. However, due to the higher coherence, numerically rank-deficient matrices occur more frequently.

2.3.6 Conclusions for Section 2.3

The numerical experiments illustrate that the three sampling strategies behave almost identically, in particular for small to moderate sampling amounts, and that sampled matrices of full rank tend to be very well-conditioned[4]. Furthermore, Section 2.3.4 shows that Bernoulli sampling can be viewed as a form of sampling without replacement, and the numerical experiments confirm the similarity in behavior.

Among the three strategies, we recommend sampling with replacement (Algorithm 2.3.2) for small to moderate amounts of sampling in Algorithm 2.2.1. It is fast and easy to implement in both.

2.4 Condition number bounds based on coherence

We derive bounds for the condition numbers of matrices produced by the sampling strategies in section 2.3, in terms of coherence. These bounds are based on a specific concentration inequality and imply a, not necessarily tight, lower bound for the number of sampled rows (Section 2.4.1). Numerical experiments illustrate that the bounds are informative (Section 2.4.2). We end this section by summarizing the main features of the bounds (Section 2.4.3).

[4] We have not been able to show rigorously why the condition numbers tend to be less than 10.

2.4.1 Bounds

We show that the three sampling strategies in Section 2.3 all have the same condition number bound, in terms of coherence.

Theorem 7 below is based on a matrix Chernoff concentration inequality (Section 2.7.1). We chose this particular inequality because extensive numerical experiments with our Matlab toolbox `kappaSQ_v3` [22] suggest that it tends to produce the tightest bound.

Theorem 7 *Let Q be a real $m \times n$ matrix with $Q^TQ = I_n$ and coherence μ. Let S be a sampling matrix produced by Algorithms 2.3.1, 2.3.2, or 2.3.3 with $n \leq c \leq m$. For $0 < \epsilon < 1$ and $f(x) \equiv e^x(1+x)^{-(1+x)}$ define*

$$\delta \equiv n \left(f(-\epsilon)^{c/(m\mu)} + f(\epsilon)^{c/(m\mu)} \right).$$

If $\delta < 1$, then with probability at least $1 - \delta$ we have $\operatorname{rank}(SQ) = n$ and

$$\kappa(SQ) \leq \sqrt{\frac{1+\epsilon}{1-\epsilon}}.$$

Proof. The proof is based on results from [16, 36, 37] and is relegated to Section 2.7.2. □

Since $0 < f(\pm\epsilon) < 1$ for $0 < \epsilon < 1$, Theorem 7 implies that the sampling strategies in Section 2.3 are more likely to produce full-rank matrices as the number c of sampled rows increases. Furthermore, for a given total number of rows m, matrices Q with fewer columns n and lower coherence μ are more likely to give rise to sampled matrices SQ that have full rank.

Theorem 7 implies the following lower bound on the number of samples, but we make no claims about the tightness of this bound.

Corollary 8 *Under the assumptions of Theorem 8,*

$$c \geq 3m\mu \, \frac{\ln(2n/\delta)}{\epsilon^2}$$

samples are sufficient to achieve $\kappa(SQ) \leq \sqrt{\frac{1+\epsilon}{1-\epsilon}}$ with probability at least $1 - \delta$.

Proof. See Section 2.7.3. □

Corollary 8 implies that the sampling strategies in Section 2.3 should sample at least $c = \Omega\left(m\mu \ln n\right)$ rows to produce a full rank, well-conditioned matrix. In particular, if Q has minimal coherence $\mu = n/m$, then Corollary 8 implies that the number of sampled rows should be at least

$$c \geq 3n \, \frac{\ln(2n/\delta)}{\epsilon^2}, \tag{2.3}$$

that is $c = \Omega\left(n \ln n\right)$.

To achieve $\kappa(SQ) \leq 10$ with probability at least .99 requires that the number of sampled rows be at least

$$c \geq 3.2\, m\mu \ \left(\ln(2n) + 4.7\right). \tag{2.4}$$

Here we chose $\epsilon_0 = 99/101$, so that the condition number bound equals $\sqrt{\frac{1+\epsilon_0}{1-\epsilon_0}} = 10$.

Remark 9 *Theorem 7 is informative only for sufficiently low coherence values.*

For instance, consider the higher coherence matrices from Figure 2.2 in Section 2.3.5 with $m = 10,000$, $n = 5$ and coherence $\mu = 150n/m$. Choose $\epsilon = 99/101$ so that $\kappa(SQ) \leq 10$, and a failure probability $\delta = .01$. Then Corollary 8 implies the lower bound $c \geq 12,408$, which means that the number of sampled rows would have to be larger than the total number of rows.

2.4.2 Numerical experiments

We compare the bound for the condition numbers of the sampled matrices (Theorem 7) with the true condition numbers of matrices produced by sampling with replacement (Algorithm 2.3.2).

There are several reasons why it suffices to consider only a single sampling strategy: The three sampling methods all have the same bound (Theorem 7); Bernoulli sampling is a form of sampling without replacement (Section 2.3.4); and all three sampling methods exhibit very similar behavior for matrices of low coherence (Sections 2.3.5 and 2.3.6). Furthermore, this allows a clean comparison with the bounds in Section 2.5 which apply only to Algorithm 2.3.2.

Experimental setup

The $m \times n$ matrices Q with orthonormal columns have $m = 10^4$ rows and $n = 5$ columns. The left panels in Figure 2.3 show the condition numbers of the full-rank sampled matrices SQ produced by Algorithm 2.3.2 against different sampling amounts c, with 30 runs for each c. The right panels in Figure 2.3 show the percentage of rank deficient matrices SQ against different sampling amounts c. We display only those sampling amounts c that give rise to rank-deficient matrices, in these particular 30 runs.

The left panels in Figure 2.3 also show the condition number bound $\kappa_\epsilon \equiv \sqrt{\frac{1+\epsilon}{1-\epsilon}}$ from Theorem 7. For each value of c, we obtain ϵ as the solution of the nonlinear equation $F_c(x)^2 = 0$ associated with Theorem 7 and defined as

$$F_c(x) \equiv \delta - n\left(f(-x)^{c/(m\mu)} + f(x)^{c/(m\mu)}\right).$$

We impose the stringent requirement of $\delta = .01$, corresponding to a 99 percent success probability. Since an explicit expression seems out of reach, we use unconstrained nonlinear optimization (a Nelder-Mead simplex direct search) to solve $F_c(x)^2 = 0$. This is done in Matlab with a code equivalent to

$$\epsilon = \left|\texttt{fminsearch}(F_c(x)^2, 0, 10^{-30})\right|,$$

where `fminsearch` starts at the point 0, and terminates when $|F_c(\epsilon)|^2 \leq 10^{-30}$. If $0 < \epsilon < 1$ then κ_ϵ is plotted, otherwise nothing is plotted.

As explained in Remark 9, Theorem 7 is not informative for higher coherence values, so we consider matrices with the following properties: Minimal coherence $\mu = n/m$ in Figure 2.3(a); low coherence $\mu = 1.5n/m$ in Figure 2.3(b); slightly higher coherence $\mu = 15n/m$ with many zero rows in Figure 2.3(c). The matrices for Figures 2.3(a) and 2.3(b) were generated with Algorithm 2.6.2, while the matrix for Figure 2.3(c) was generated with Algorithm 2.6.3.

Figure 2.3

The left panels illustrate that Theorem 7, constrained to a 99 percent success probability, correctly predicts the magnitude of the condition numbers, i.e. $\kappa(SQ) \leq 10$. Hence Theorem 7 provides informative qualitative bounds for matrices with very low coherence, as well as for matrices with slightly higher coherence and many zero rows.

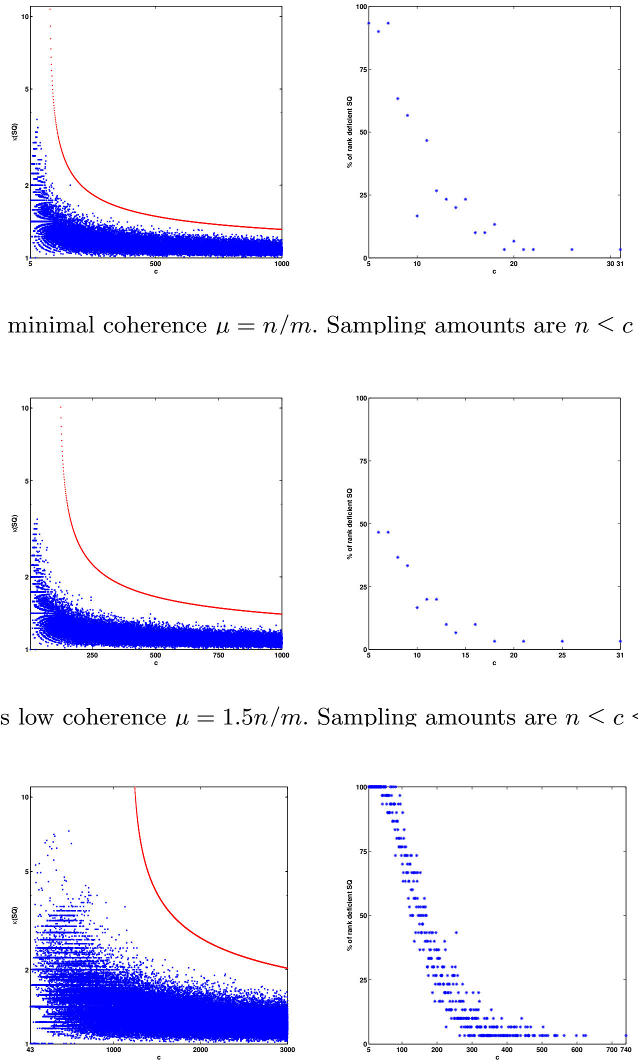
Table 2.1

This is a comparison of the numerical experiments in Figure 2.3 with the bounds from Theorem 7 and Corollary 8, both restricted to a 99 percent success probability.

The third column depicts the highest values of c for which a rank-deficient matrix occurs, during these particular 30 runs. It should be kept in mind that these values are highly dependent on the particular sampling runs. This column is to be compared to the fourth column which contains the lowest values of c where Theorem 7 starts to apply. Although there is a gap between the occurrence of the last rank deficiency and the onset of Theorem 7, the values have qualitatively the same order of magnitude.

The rightmost column in Table 2.1 contains the values of the lower bound (2.4), and is to be compared to the column with the starting values for Theorem 7. Although (2.4) is weaker than Theorem 7, its values are close to the starting values of Theorem 7, especially for lower coherence. Hence, the lower bound (2.4) captures the correct magnitude of the sampling amounts where Theorem starts to become informative.

Table 2.1 illustrates that, although Theorem 7 and Corollary 8 tend to become more pessimistic with increasing coherence, they still provide qualitative information for matrices with low coherence – even when restricted to a 99 percent success probability.



(a) Q has minimal coherence $\mu = n/m$. Sampling amounts are $n < c < 1,000$.

(b) Q has low coherence $\mu = 1.5n/m$. Sampling amounts are $n < c < 1,000$.

(c) Q has slightly higher coherence $\mu = 15n/m$ and many zero rows. Sampling sampling amounts are $n \leq c \leq 3,000$.

Figure 2.3: Condition numbers and bound from Theorem 7, and percentage of rank-deficiency. Here Q is $m \times n$ with orthonormal columns, $m = 10,000$ and $n = 5$. Left panels: The horizontal coordinate axes represent amounts of sampling c. The vertical coordinate axes represent condition numbers $\kappa(SQ)$; the maximum is 10. The dots at the bottom represent the condition numbers of matrices sampled with Algorithm 2.3.2, while the upper line represents the bound from Theorem 7. Right panels: The horizontal coordinate axes represent amounts of sampling that produce numerically rank deficient matrices SQ. The vertical coordinate axes represent the percentage of numerically rank deficient matrices SQ.

Table 2.1: Comparison of information from Figure 2.3, with Theorem 7 and Corollary 8.

Figure	coherence μ	last rank deficiency occurs at $c =$	Theorem 7 starts at $c =$	(2.4)
2.3(a)	n/m	31	81	83
2.3(b)	$1.5\, n/m$	31	121	125
2.3(c)	$15\, n/m$	740	1207	1241

2.4.3 Conclusions for Section 2.4

The bounds in Theorem 7 and Corollary 8 have the following advantages:

1. They are non-asymptotic bounds, where all constants have explicit numerical values, hence they are tighter than the bounds in [1, Theorem 3.2].
2. They apply to three different sampling methods.
3. They imply a lower bound, of $\Omega\,(m\mu \ln n)$, on the required number of sampled rows. Although we did not give a formal proof of tightness, numerical experiments illustrate that sampling only the required number of rows implied by the bound is realistic. numerical experiments illustrate that the bound is realistic.
4. Even under the stringent requirement of a 99 percent success probability, they are informative for matrices of small dimension because they correctly predict the magnitude of the condition numbers for the sampled matrices.

Note that the bounds in Theorem 7 and Corollary 8 are informative only for matrices that are tall and skinny ($m \gg n$) and have low coherence. The restriction to tall and skinny matrices is not an imposition, because it is required for the effectiveness of the sampling strategies, see Section 2.3.5.

In the next section we try to relax the restriction to low coherence matrices, by more thoroughly exploiting the information available from the row norms of Q.

2.5 Condition number bounds based on leverage scores, for uniform sampling with replacement

The goal is to tighten Theorem 7 by making use of all the row norms of Q, instead of just the largest one. To this end we introduce leverage scores (Section 2.5.1), which are the squared row norms of Q. We use them to derive a bound for uniform sampling with replacement (Section 2.5.2), and for more easily computable versions of the bound (Section 2.5.3). Analytical (Section 2.5.4) and experimental (Section 2.5.5) comparisons demonstrate that the implied lower bound on the number of sampled rows is better than the coherence-based bounds in Section 2.4. A review with some reflection ends this section (Section 2.5.6).

2.5.1 Leverage scores

So-called *statistical leverage scores* were first introduced in 1978 by Hoaglin and Welsch [20] to detect outliers when computing regression diagnostics, see also [8, 38]. Mahoney and Drineas pioneered the use of leverage scores for importance sampling strategies in randomized matrix computations [25].

Specifically, if M is a real $m \times n$ matrix with $\operatorname{rank}(M) = n$, then the $m \times m$ *hat matrix*

$$H \equiv M(M^T M)^{-1} M^T$$

is the orthogonal projector onto the column space of M, and its diagonal elements are called *leverage scores* [20, Section 2]. Hence, leverage scores are basis-independent. For our purposes, though, it suffices to define them in terms of a thin QR decomposition $M = QR$, so that the hat matrix can be expressed as $H = QQ^T$.

Definition 10 *If Q is a $m \times n$ matrix with $Q^T Q = I_n$, then its* leverage scores *are*

$$\ell_j \equiv \|e_j^T Q\|_2^2, \qquad 1 \leq j \leq m.$$

The $m \times m$ diagonal matrix of leverage scores is

$$L \equiv \operatorname{diag}(\ell_1, \ldots, \ell_m).$$

Note that the coherence is the largest leverage score,

$$\mu = \max_{1 \leq j \leq m} \ell_j = \|L\|_2.$$

2.5.2 Bounds

The bound in Theorem 11 below involves leverage scores and is based on a matrix Bernstein concentration inequality (Section 2.7.4), rather than on the matrix Chernoff concentration inequality (Section 2.7.1) for Theorem 7. Although the Bernstein inequality may not always be as tight, we did not see how to insert leverage scores into the Chernoff inequality.

Theorem 11 *Let Q be a $m \times n$ real matrix with $Q^TQ = I_n$, leverage scores ℓ_j, $1 \leq j \leq m$, and coherence μ. Let S be a sampling matrix produced by Algorithm 2.3.2 with $n \leq c \leq m$. For $0 < \epsilon < 1$ set*

$$\delta \equiv 2n \exp\left(-\tfrac{3}{2}\,\frac{c\epsilon^2}{m\,(3\|Q^TLQ\|_2 + \epsilon\mu)}\right).$$

If $\delta < 1$, then with probability at least $1 - \delta$ we have $\operatorname{rank}(SQ) = n$ *and*

$$\kappa(SQ) \leq \sqrt{\frac{1+\epsilon}{1-\epsilon}}.$$

Proof. The proof uses results from [2, 32] and is relegated to Section 2.7.5. □

Like Theorem 7, Theorem 11 implies that sampling with replacement is more likely to produce full-rank matrices as the number c of sampled rows increases. Furthermore, for a given total number of rows m, matrices Q with fewer columns n and lower coherence μ are more likely to yield sampled matrices SQ that have full rank. The dependence of $\|Q^TLQ\|_2$ on μ is discussed below.

Remark 12 *The norm $\|Q^TLQ\|_2$ has simple and tight bounds in terms of the coherence,*

$$\mu^2 \leq \|Q^TLQ\|_2 \leq \mu. \tag{2.5}$$

The lower bound follows from $\|Q^TLQ\|_2 = \|L^{1/2}Q\|_2^2$ and

$$\|L^{1/2}Q\|_2 \geq \|e_j^TL^{1/2}Q\|_2 = \ell_j^{1/2}\;\|e_j^TQ\|_2 = \ell_j, \qquad 1 \leq j \leq m,$$

which implies $\|L^{1/2}Q\|_2 \geq \mu$.

The bounds (2.5) are attained for extreme values of the coherence:

- *In case of minimal coherence $\mu = \ell_j$ for all $1 \leq j \leq m$, we have $L = \mu I_m$. Thus $\|Q^TLQ\|_2 = \mu\|Q^TQ\|_2 = \mu$, and the upper bound is attained.*
- *In case of maximal coherence $\mu = 1$, we have $\mu^2 = \mu$. Thus $\|Q^TLQ\|_2 = \mu^2 = \mu$, and both, lower and upper bounds are attained.*

2.5.3 Computable bounds

We present easily computable bounds for $\|Q^TLQ\|_2$, based on coherence and several of the largest leverage scores.

To this end, we use a labeling of the leverage scores in non-increasing order,

$$\mu = \ell_{[1]} \geq \cdots \geq \ell_{[m]}.$$

Corollary 13 *Under the assumptions of Theorem 11, if* $t \equiv \lfloor 1/\mu \rfloor$*, then*

$$\|Q^T L Q\|_2 \leq \mu \sum_{j=1}^{t} \ell_{[j]} + (1 - t\,\mu)\, \ell_{[t+1]} \leq \mu.$$

If, in addition, t *is an integer, then* $\|Q^T L Q\|_2 \leq \mu \sum_{j=1}^{t} \ell_{[j]}$.

Proof. See Section 2.8.2. □

The number of large leverage scores appearing in Corollary 13 depends on the coherence: Few leverage scores for high coherence, but more for low coherence. Henceforth we will use the approximation from Corollary 13 instead of the true value $\|Q^T L Q\|_2$, for two reasons: First, numerical experiments show that the approximation tends to be very accurate. Second, the approximation is convenient, because it requires only a leverage score distribution rather than a full-fledged matrix Q.

Remark 14 *Corollary 13 is tight for the extreme cases of minimal and maximal coherence.*

- *In case of minimal coherence* $\mu = \ell_j$ *for all* $1 \leq j \leq m$*, Remark 12 implies* $\|Q^T L Q\|_2 = \mu$*. The bound in Corollary 13 is* $\|Q^T L Q\|_2 \leq \mu$*, thus tight.*
- *In case of maximal coherence* $\mu = 1$*, Remark 12 implies* $\|Q^T L Q\|_2 = \mu^2 = \mu$*. Corollary 13 holds with* $t = 1$ *and gives the bound* $\|Q^T L Q\|_2 \leq \mu$*, which is tight as well.*

Inserting this approximation for $\|Q^T L Q\|_2$ into the expression for δ in Theorem 11 yields a, not necessarily tight, lower bound on the number of samples.

Corollary 15 *Under the assumptions of Theorem 11,*

$$c \geq \tfrac{2}{3} m \,(3\tau + \epsilon\mu)\, \frac{\ln(2n/\delta)}{\epsilon^2},$$

where $\tau \equiv \mu \sum_{j=1}^{t} \ell_{[j]} + (1 - t\,\mu)\, \ell_{[t+1]}$*, samples are sufficient to achieve* $\kappa(SQ) \leq \sqrt{\frac{1+\epsilon}{1-\epsilon}}$ *with probability at least* $1 - \delta$*.*

In particular, if Q has minimal coherence $\mu = n/m$, then Corollary 15 implies that the number of sampled rows should be at least

$$c \geq 3n\, \frac{\ln(2n/\delta)}{\epsilon^2}.$$

This is the same as the coherence-based lower bound (2.3).

To achieve $\kappa(SQ) \leq 10$ with probability at least .99 requires that the number of sampled rows be at least

$$c \geq m\,(2.1\tau + .7\mu)\,\left(\ln(2n) + 4.7\right). \tag{2.6}$$

2.5.4 Analytical comparison of the bounds in Sections 2.4.1 and 2.5.2

An analytical comparison between Theorems 7 and 11 is not obvious, because they are based on different concentration inequalities. Instead we compare the implied lower bounds for the number of sampled rows, and show that the leverage-score based bound in Corollary 15 is at least as tight as the coherence-based bound in Corollary 8.

Corollary 16 *Under the assumptions of Theorem 11 and Corollary 13,*

$$\tfrac{2}{3}m\,(3\tau + \epsilon\mu)\,\frac{\ln(2n/\delta)}{\epsilon^2} \leq 3m\mu\,\frac{\ln(2n/\delta)}{\epsilon^2}.$$

Hence Corollary 15 is at least as tight as Corollary 8.

Proof. See Section 2.8.3. □

2.5.5 Experimental comparison of the bounds in Sections 2.4.1 and 2.5.2

We present numerical experiments to compare the lower bounds for the number of sampled rows in Corollaries 8 and 15, for different values of coherence. This gives quantitative insight into the comparison in Corollary 16, and illustrates the reduction in the number of sampled rows from Corollary 15, as compared to Corollary 8.

Experimental setup

As in previous sections, we use $m \times n$ matrices with $m = 10^4$ rows and $n = 5$ columns. The success probability is .99; and $\epsilon = 99/101$, so that the bound for $\kappa(SQ)$ is equal to 10. Hence the bounds in Corollaries 8 and 15 amount to (2.4) and (2.6), respectively.

We consider two different leverage scores distributions: A distribution generated by Algorithm 2.6.2 with one large leverage score in Table 2.2; and a distribution generated by Algorithm 2.6.3 with as many zeros as possible in Table 2.3.

Table 2.2

This table shows the lower bounds on the number of sampled rows, for a leverage score distribution generated with Algorithm 2.6.2 that consists of one large leverage score, equal to

Table 2.2: Lower bounds for number of sampled rows in Corollaries 8 and 15 and approximation τ, for different values of coherence μ. The first value represents minimal coherence $\mu = n/m$. Here $m = 10,000$, $n = 5$, $\delta = .01$, $\epsilon = 99/101$, with leverage scores generated by Algorithm 2.6.2.

$\mu/(n/m)$	1	5	10	15	20	25	50	100
Cor. 8	108	540	1,079	1,618	2,157	2,697	5,393	10,786
Cor. 15	96	191	310	432	556	682	1,3343	2,777
$\tau/(n/m)$	1.00	1.01	1.04	1.10	1.19	1.30	2.22	9.95

the coherence, and all remaining leverage scores being non-zero and identical. The bounds, as well as the approximation τ to $\|Q^T LQ\|_2$, are displayed for eight different values of coherence, ranging from minimal coherence $\mu = n/m$ to $\mu = 100n/m$.

Table 2.2 illustrates that with increasing coherence, the number of sampled rows implied by Corollary 15 is only about 20 percent of that from Corollary 8. This is because τ increases much more slowly than μ. For instance, $\tau \approx \mu/10$ when $\mu = 100n/m$.

Table 2.3

This table shows the lower bounds on the number of sampled rows. The corresponding leverage score distribution is generated with Algorithm 2.6.3 and consists of as many zeros as possible. All non-zero leverage scores, expect possibly one, are equal to the coherence μ, so that $\tau \approx \mu$. The bounds are displayed for eight different values of coherence, ranging from minimal coherence $\mu = n/m$ to $\mu = 100n/m$.

The bounds for Corollary 8 are the same as in Table 2.2, because the coherence values are the same. Since $\tau = \mu$, the difference between Corollaries 8 and 15 is not as drastic as in Table 2.2, yet it increases with increasing coherence. For $\mu = 100n/m$, Corollary 15 remain informative, while Corollary 8 does not.

2.5.6 Conclusions for Section 2.5

The goal of this section was to derive condition number bounds that are based on leverage scores rather than just coherence, when rows are sampled uniformly with replacement (Algorithm 2.3.2). Corollary 16 and the numerical experiments illustrate that the lower bound on the number of sampled rows implied by Corollary 15 is smaller than that from Corollary 8.

Although the coherence based bound in Theorem 7 is derived from a stronger concentration inequality than the one for Theorem 11, this difference disappears in the weakening necessary to obtain lower bounds for the amount of sampling. Even in cases when the leverage score

Table 2.3: Lower bounds for number of sampled rows in Corollaries 8 and 15, for different values of coherence μ. The first value represents minimal coherence $\mu = n/m$. Here $m = 10,000$, $n = 5$, $\delta = .01$, $\epsilon = 99/101$, with leverage scores generated by Algorithm 2.6.3.

$\mu/(n/m)$	1	5	10	15	20	25	50	100
Cor. 8	108	540	1,079	1,618	2,157	2,697	5,393	10,787
Cor. 15	96	477	954	1,431	1,908	2,385	4,770	9,539

measure τ is the same as the coherence, Corollary 15 still retains a small advantage, which can increase with increasing coherence. Hence Corollary 15 tends to remain informative for larger values of coherence, even when Corollary 8 fails.

The difference in implied sampling amounts becomes more drastic in the presence widely varying non-zero leverage scores, and can be as high as ten percent. This is because the coherence-based bound in Corollary 8 cannot take advantage of the distribution of the leverage scores.

Hence, when it comes to lower bounds for the number of rows sampled uniformly with replacement, we recommend Corollary 15.

We have yet to derive leverage score based bounds for the other two sampling strategies, uniform sampling without replacement (Algorithm 2.3.1) and Bernoulli sampling (Algorithm 2.3.3).

2.6 Algorithms for generating matrices with prescribed coherence and leverage scores

In order to investigate the efficiency of the sampling methods in Section 2.3, and test the tightness of the bounds in Sections 2.4 and 2.5, we need to generate matrices with orthonormal columns that have prescribed leverage scores and coherence. The algorithms are implemented in the Matlab package *kappa_SQ_v3* [22].

We present algorithms for generating matrices with prescribed leverage scores and coherence (Section 2.6.1), and for generating particular leverage score distributions with prescribed coherence (Section 2.6.2). Such distributions can then, in turn, serve as inputs for the algorithm in Section 2.6.1. Furthermore we present two classes of structured matrices with prescribed coherence that are easy and fast to generate (Section 2.6.3).

2.6.1 Matrices with prescribed leverage scores

We present an algorithm that generates matrices with orthonormal columns that have prescribed leverage scores. In Section 2.9 we prove an existence result to show that this is always possible.

Algorithm 2.6.1 is a transposed version of [9, Algorithm 3]. It repeatedly applies $m \times m$ Givens rotations G_{ij} that rotate two rows i and j, and are computed from numerically stable expressions [9, section 3.1]. At most $m - 1$ such rotations are necessary. Since each rotation affects only two rows, Algorithm 2.6.1 requires $\mathcal{O}(mn)$ arithmetic operations.

Algorithm 2.6.1 Generating a matrix with prescribed leverage scores [9]

```
Input: Integers $m$ and $n$ with $m \geq n \geq 1$
       Vector $\ell$ with elements $0 \leq \ell_1 \leq \cdots \leq \ell_m \leq 1$ and $\sum_{j=1}^{m} \ell_j = n$
Output: $m \times n$ matrix $Q$ with $Q^T Q = I_n$ and leverage scores $\|e_j^T Q\|_2^2 = \ell_j$, $1 \leq j \leq m$

  $Q = \begin{pmatrix} I_n & 0_{n\times(m-n)} \end{pmatrix}^T$        {Initialization}
  repeat
    Determine indices $i < k < j$ with
    $\|e_i^T Q\|_2^2 < \ell_i$, $\|e_k^T Q\|_2^2 = \ell_k$, $\|e_j^T Q\|_2^2 > \ell_j$
    if $\ell_i - \|e_i^T Q\|_2^2 \leq \|e_j^T Q\|_2^2 - \ell_j$ then
      Apply rotation $G_{ij}$ to rows $i$ and $j$ so that $\|e_i^T G_{ij}\, Q\|_2^2 = \ell_i$
    else
      Apply rotation $G_{ij}$ to rows $i$ and $j$ so that $\|e_j^T G_{ij}\, Q\|_2^2 = \ell_j$
    end if
    $Q = G_{ij}\, Q$        {Update}
  until no more such indices exist
```

2.6.2 Leverage score distributions with prescribed coherence

We present algorithms that generate leverage score distributions for prescribed coherence. The resulting distributions then serve as inputs for Algorithm 2.6.1. These particular leverage score distribution help to distinguish the effect of coherence, which is the largest leverage score, from that of the remaining leverage scores.

One large leverage score

Given a prescribed coherence μ, Algorithm 2.6.2 generates a distribution consisting of one large leverage score equal to μ and the remaining leverage scores being identical and non-zero.

Algorithm 2.6.2 Generating a leverage score distribution with prescribed coherence: One large leverage score

```
Input:  Integers $m$ and $n$ with $m \geq n \geq 1$
        Real number $\mu$ with $n/m \leq \mu \leq 1$
Output: Vector $\ell$ with elements $\ell_1 = \mu$, $0 < \ell_j \leq 1$ and $\sum_{j=1}^{m} \ell_j = n$

  $\ell_1 = \mu$
  for $j = 2 : m$ do
    $\ell_j = \frac{n-\mu}{m-1}$
  end for
```

In the special case of minimal coherence $\mu = n/m$, Algorithm 2.6.2 generates m identical leverages equal to μ, which is the only possible leverage score distribution in this case.

Many zero leverage scores

Given a prescribed coherence, Algorithm 2.6.3 generates a distribution with as many zero leverage scores as possible. This serves as an "adversarial" distribution for the sampling algorithms in Section 2.3.

Given a prescribed coherence μ, Algorithm 2.6.3 first determines the smallest number of rows m_s that can realize this coherence, sets $m_s - 1$ leverage scores equal to μ, assigns another leverage score to to take up the possibly non-zero slack, and sets the remaining leverage scores to zero.

Algorithm 2.6.3 Generating a leverage score distribution with prescribed coherence: Many zero leverage scores

```
Input:  Integers $m$ and $n$ with $m \geq n \geq 1$
        Real number $\mu$ with $n/m \leq \mu \leq 1$
Output: Vector $\ell$ with elements $\ell_1 = \mu$, $0 \leq \ell_j \leq 1$ and $\sum_{j=1}^{m} \ell_j = n$

  $m_s = \lceil n/\mu \rceil$        {Number of nonzero rows}
  for $j = 1 : m_s - 1$ do
    $\ell_j = \mu$
  end for
  $\ell_{m_s} = n - (m_s - 1)\,\mu$
  for $j = m_s + 1 : m$ do
    $\ell_j = 0$
  end for
```

2.6.3 Structured matrices with prescribed coherence

We present two classes of structured matrices with orthonormal columns that have prescribed coherence. Although the structure puts constraints on the matrix dimensions, the generation of these matrices is faster than running Algorithm 2.6.1. Note that the matrices produced by Algorithm 2.6.1 also have structure, but it is not easily characterized.

Stacks of diagonal matrices Given matrix dimensions m and n, where $s = m/n$ is an integer, and prescribed coherence μ. The $m \times n$ matrix Q below has orthonormal columns and coherence μ, and consists of s stacks of $n \times n$ diagonal matrices,

$$Q = \begin{pmatrix} \sqrt{\mu}\, I_n \\ \phi\, I_n \\ \vdots \\ \phi\, I_n \end{pmatrix} \qquad \textit{where} \qquad \phi \equiv \sqrt{\frac{1-\mu}{\frac{m}{n}-1}}.$$

Matrices with Hadamard structure Given matrix dimensions m and n, where $m = 2^k$ and $n < m$ is also a power of two, and prescribed coherence μ. The $m \times n$ matrix

$$Q = D_k \begin{pmatrix} I_n \\ 0 \end{pmatrix}$$

has orthonormal columns and coherence μ, and is defined recursively as follows. For

$$\alpha \equiv \sqrt{\frac{\mu - \frac{n-1}{m-1}}{1 - \frac{n-1}{m-1}}}, \qquad \beta \equiv \sqrt{\frac{1-\alpha^2}{m-1}}$$

define square matrices B_j of dimension 2^j and square matrices D_j of dimension 2^{j+1} as follows,

$$B_0 = \beta, \qquad B_{j+1} = \begin{pmatrix} -B_j & B_j \\ B_j & B_j \end{pmatrix} \qquad 0 \le j \le k-1$$

$$D_1 = \begin{pmatrix} \alpha & -\beta \\ \beta & \alpha \end{pmatrix}, \qquad D_{j+1} = \begin{pmatrix} D_j & -B_j \\ B_j & D_j \end{pmatrix}.$$

Note that only the final matrix Q has orthonormal columns and coherence μ while, in general, the intermediate matrices B_j and D_j do not. We omit the messy induction proof, because it does not provide much insight.

2.7 Proofs for Sections 2.4 and 2.5.2

For the coherence-based bounds in Section 2.4 we first present a matrix concentration inequality (Section 2.7.1), and then the proofs of Theorem 7 (Section 2.7.2) and Corollary 8 (Section 2.7.3).

For the bound based on leverage scores in Section 2.5.2, we first present a matrix concentration inequality (Section 2.7.4), and then the proof of Theorem 11 (Section 2.7.5).

2.7.1 Matrix Chernoff Concentration inequality

The matrix concentration inequality below is the basis for Theorem 7 and Corollary 8.

Denote the eigenvalues of a Hermitian matrix Z by $\lambda_j(Z)$, and the smallest and largest eigenvalues by $\lambda_{min}(Z) \equiv \min_j \lambda_j(Z)$ and $\lambda_{max}(Z) \equiv \max_j \lambda_j(Z)$, respectively.

Theorem 17 (Corollary 5.2 in [37]) *Let X_j be a finite number of independent random $n \times n$ Hermitian positive semidefinite matrices with $\max_j \|X_j\|_2 \leq \tau$. Define*

$$\omega_{min} \equiv \lambda_{min}\left(\sum_j \mathbf{E}[X_j]\right) \qquad \omega_{max} \equiv \lambda_{max}\left(\sum_j \mathbf{E}[X_j]\right),$$

and $f(x) \equiv e^x (1+x)^{-(1+x)}$. Then for any $0 \leq \epsilon < 1$

$$\mathbf{Pr}\left[\lambda_{min}\left(\sum_j X_j\right) \leq (1-\epsilon)\,\omega_{min}\right] \leq n\, f(-\epsilon)^{\omega_{min}/\tau},$$

and for any $\epsilon \geq 0$

$$\mathbf{Pr}\left[\lambda_{max}\left(\sum_j X_j\right) \geq (1+\epsilon)\,\omega_{max}\right] \leq n\, f(\epsilon)^{\omega_{max}/\tau}.$$

2.7.2 Proof of Theorem 7

We present a separate proof for each sampling method.

Algorithm 2.3.1: Sampling without replacement The proof follows directly from [36, Lemma 3.4].

Algorithm 2.3.2: Sampling with replacement The proof is based on Theorem 17, and turns out to be somewhat similar to that of [36, Lemma 3.4].

Set $X_t \equiv \frac{m}{c}\, Q^T e_{k_t} e_{k_t}^T Q$, $1 \leq t \leq c$. Then X_t is $n \times n$ Hermitian positive semidefinite and $\|X_t\|_2 \leq \frac{m}{c} \|e_{k_t}^T Q\|_2^2 \leq \frac{m\mu}{c}$. Hence we set $\tau = m\mu/c$. Furthermore,

$$\mathbf{E}[X_t] = \sum_{j=1}^{m} \tfrac{1}{m} \, \left(\tfrac{m}{c}\, Q^T e_j e_j^T Q\right) = \tfrac{1}{c} \, \sum_{j=1}^{m} Q^T e_j e_j^T Q = \tfrac{1}{c} I_n.$$

Hence the eigenvalues of the sum are $\lambda_j \left(\sum_{t=1}^{c} \mathbf{E}[X_t]\right) = \lambda_j(I_n) = 1$, $1 \leq j \leq n$, and we set $\omega_{min} = \omega_{max} = 1$. Applying Theorem 17 to $\sum_{t=1}^{c} X_t = Q^T S^T S Q$ gives

$$\begin{aligned} \mathbf{Pr}\left[\lambda_{min}\left(Q^T S^T S Q\right) \leq 1 - \epsilon\right] \leq &\quad n f(-\epsilon)^{c/(m\mu)} \\ \mathbf{Pr}\left[\lambda_{max}\left(Q^T S^T S Q\right) \geq 1 + \epsilon\right] \leq &\quad n f(\epsilon)^{c/(m\mu)}. \end{aligned}$$

The result follows from Boole's inequality [34, p. 16].

Algorithm 2.3.3: Bernoulli sampling The proof is similar to the one above, and a special case of [16, Theorem 6.1].

Set

$$X_j \equiv \tfrac{m}{c} \begin{cases} Q^T e_j e_j^T Q & \text{with probability } \tfrac{c}{m} \\ 0_{n\times n} & \text{with probability } 1 - \tfrac{c}{m} \end{cases}, \qquad 1 \leq j \leq m.$$

Then X_j is $n \times n$ Hermitian positive semidefinite, $\|X_j\|_2 \leq \frac{m}{c} \|e_j^T Q\|_2^2 \leq \frac{m\mu}{c}$. As above, we set $\tau = m\mu/c$. Furthermore,

$$\mathbf{E}[X_j] = \tfrac{c}{m} \cdot \tfrac{m}{c} Q^T e_j e_j^T Q + (1 - \tfrac{c}{m}) \cdot 0_{n\times n} = Q^T e_j e_j^T Q,$$

which implies $\sum_{j=1}^{m} \mathbf{E}[X_j] = \sum_{j=1}^{m} Q^T e_j e_j^T Q = I_n$. Now proceed as in the above proof for Algorithm 2.3.2, and apply Theorem 17 to $\sum_{j=1}^{m} X_j = Q^T S^T S Q$.

2.7.3 Proof of Corollary 8

First we simplify the bound in Theorem 7 based on the inequality $f(-x) \leq f(x)$ for $0 < x < 1$. This implies for Theorem 7 that

$$\delta \equiv n \left(f(-\epsilon)^{c/(m\mu)} + f(\epsilon)^{c/(m\mu)}\right) \delta \leq 2n\, f(\epsilon)^{c/(m\mu)}.$$

Solving for c gives

$$c \geq m\mu \, \frac{\ln(2n/\delta)}{-\ln f(\epsilon)}.$$

If we can show that $-\ln f(\epsilon) > \epsilon^2/3$, then the above lower bound for c definitely holds if

$$c \geq 3m\mu \, \frac{\ln(2n/\delta)}{\epsilon^2}.$$

To show $-\ln f(\epsilon) > \epsilon^2/3$ for $0 < \epsilon < 1$, apply the definition $f(x) = e^x(1+x)^{-(1+x)}$ so that $h(x) \equiv -\ln f(x) = (1+x)\ln(1+x) - x$. Expand into the power series $\ln(1+x) = \sum_{j=1}^{\infty}(-1)^{j+1}\frac{x^j}{j}$. For $0 < x < 1$ this yields $h(x) = \frac{1}{2}x^2 - \frac{1}{6}x^3 + E(x)$, where

$$E(x) \equiv \sum_{j=4}^{\infty}(-1)^j \, \frac{x^j}{(j-1)j} = \sum_{j=2}^{\infty}\left(\frac{2j+1-(2j-1)x}{(2j-1)2j(2j+1)}\right) \, x^{2j} > 0,$$

since each summand is positive for $0 < x < 1$. Thus for $0 < x < 1$ we obtain

$$h(x) > \frac{1}{2}x^2 - \frac{1}{6}x^3 = \frac{3-x}{6} \, x^2 \geq \frac{x^2}{3}.$$

2.7.4 Matrix Bernstein concentration inequality

The matrix concentration inequality below is the basis for Theorem 11. It is a version specialized to square matrices of [32, Theorem 4]. In numerical experiments we found it to be tighter than [15, Theorem 4] and the Frobenius norm bound [11, Theorem 2].

Theorem 18 (Theorem 4 in [32]) *Let X_j be m independent random $n \times n$ matrices with $\mathbf{E}[X_j] = 0_{n\times n}$, $1 \leq j \leq m$. Let $\rho_j \equiv \max\{\|\mathbf{E}[X_j X_j^T]\|_2, \|\mathbf{E}[X_j^T X_j]\|_2\}$ and $\max_{1\leq j\leq m}\|X_j\|_2 \leq \tau$. Then for any $\epsilon > 0$*

$$\mathbf{Pr}\left[\|\sum_{j=1}^{m} X_j\|_2 > \epsilon\right] \leq 2n \, \exp\left(-\tfrac{3}{2}\,\frac{\epsilon^2}{3\sum_{j=1}^{m}\rho_j + \tau\epsilon}\right).$$

2.7.5 Proof of Theorem 11

The proof is similar to that of [2, Lemma 3]. Represent the outcome of uniform sampling with replacement in Algorithm 2.3.2 by $Q^T S^T S Q = \sum_{t=1}^{c} Y_t$, where $Y_t \equiv \frac{m}{c}\,Q^T e_{k_t} e_{k_t}^T Q$ are $n \times n$ matrices, $1 \leq t \leq c$, with expected value

$$\mathbf{E}[Y_t] = \sum_{j=1}^{m}\frac{1}{m}\,\frac{m}{c} Q^T e_j e_j^T Q = \frac{1}{c}\,\sum_{j=1}^{m} Q^T e_j e_j^T Q = \frac{1}{c} I_n.$$

Thus, the zero mean versions are $X_t \equiv Y_t - \frac{1}{c}I_n$. To apply Theorem [32, Theorem 4] to the X_t we need to verify that they fulfill the required conditions. First, by construction, $\mathbf{E}[X_t] = 0$,

$1 \leq t \leq c$. Second, since Y_t and I_n are symmetric positive semidefinite,

$$\|X_t\|_2 \leq \max\{\|Y_t\|_2, \|\frac{1}{c} I_n\|_2\} = \frac{1}{c} \max\{m \, \|e_{k_t}^T Q\|_2^2, 1\} \leq \frac{m\mu}{c},$$

where the last inequality follows from the definition of μ, and $\mu \geq n/m$. Hence we set $\tau = m\mu/c$. Third, since X_t is symmetric,

$$X_t^T X_t = X_t X_t^T = X_t^2 = Y_t^2 - \frac{2}{c} Y_t + \frac{1}{c^2} I_n.$$

From $\mathbf{E}[Y_t] = \frac{1}{c} I_n$ follows

$$\mathbf{E}[X_t^2] = \mathbf{E}[Y_t^2] - \frac{2}{c}\, \mathbf{E}[Y_t] + \frac{1}{c^2} I_n = \mathbf{E}[Y_t^2] - \frac{1}{c^2} I_n. \tag{2.7}$$

Since $Y_t^2 = \frac{m^2}{c^2}\, \ell_{k_t}\, Q^T e_{k_t} e_{k_t}^T Q$, we obtain

$$\mathbf{E}[Y_t^2] = \sum_{j=1}^{m} \frac{1}{m} \frac{m^2}{c^2}\, \ell_j Q^T e_j e_j^T Q = \frac{m}{c^2}\, Q^T \left(\sum_{j=1}^{m} \ell_j e_j e_j^T \right) Q = \frac{m}{c^2}\, Q^T L Q.$$

Substituting this into (2.7) yields

$$\mathbf{E}[X_t^2] = \tfrac{1}{c^2} \left(m \, Q^T L Q - I_n \right).$$

Positive semi-definiteness gives

$$\|\mathbf{E}[X_t^2]\|_2 \leq \frac{1}{c^2} \, \max\{m \, \|Q^T L Q\|_2, \, 1\} = \frac{m}{c^2} \, \|Q^T L Q\|_2.$$

We set $\rho_t = \frac{m}{c^2} \, \|Q^T L Q\|_2$. Applying [32, Theorem 4] to

$$\sum_{t=1}^{c} X_t = \sum_{t=1}^{c} \left(Y_t - \tfrac{1}{c} I_n \right) = (SQ)^T (SQ) - I_n$$

shows that $\| \sum_{t=1}^{c} X_t \|_2 \leq \epsilon$ with probability at least $1 - \delta$.

2.8 Two-norm bound for scaled matrices, and proofs for Sections 2.5.3 and 2.5.4

We derive a bound for the two-norm of diagonally scaled matrices (Section 2.8.1), which leads immediately to the proofs of Corollary 13 (Section 2.8.2), and Corollary 16 (Section 2.8.3).

2.8.1 Bound

We present two majorization bounds for Hadamard products of vectors (Lemmas 21 and 20), and use them to derive a bound for the two-norm of diagonally scaled matrices (Theorem 22).

Definition 19 (Definition 4.3.41 in [21]) *Let a and b be vectors with m real elements. The elements, labelled in algebraically decreasing order, are $a_{[1]} \geq \cdots \geq a_{[m]}$ and $b_{[1]} \geq \cdots \geq b_{[m]}$. The vector a* weakly majorizes *the vector b, if*

$$\sum_{j=1}^{k} a_{[j]} \geq \sum_{j=1}^{k} b_{[j]}, \qquad 1 \leq k \leq m.$$

The vector a majorizes *the vector b, if a weakly majorizes b and also $\sum_{j=1}^{m} a_{[j]} = \sum_{j=1}^{m} b_{[j]}$.*

The first lemma follows from a stronger majorization inequality for functions that are monotone and lattice superadditive.

Lemma 20 (Theorem II.4.2 in [3]) *If b and x are vectors with m non-negative elements, then*

$$\sum_{j=1}^{k} b_j \, x_j \leq \sum_{j=1}^{k} b_{[j]} \, x_{[j]}, \qquad 1 \leq k \leq m.$$

The second lemma is a variant of a well-known majorization result for Hadamard products of vectors [21, Lemma 4.3.51]. Since the version below is slightly different, we include a proof from first principles.

Lemma 21 *Let x, a and b be vectors with m non-negative elements. If a weakly majorizes b, then*

$$\sum_{j=1}^{k} a_{[j]} \, x_{[j]} \geq \sum_{j=1}^{k} b_{[j]} \, x_{[j]}, \qquad 1 \leq k \leq m.$$

Proof. The following arguments hold for $1 \leq k \leq m-1$. Start out with the upper bound, and separate the last summand,

$$\sum_{j=1}^{k+1} a_{[j]} \, x_{[j]} \quad = \quad \sum_{j=1}^{k} a_{[j]} \, x_{[j]} + a_{[k+1]} \, x_{[k+1]}. \tag{2.8}$$

Re-writing the right sum and applying $x_{[j]} \geq x_{[k+1]} \geq 0$, $1 \leq j \leq k$, gives

$$\begin{aligned} \sum_{j=1}^{k} a_{[j]}\, x_{[j]} &= \sum_{j=1}^{k} b_{[j]}\, x_{[j]} + \sum_{j=1}^{k} (a_{[j]} - b_{[j]})\, x_{[j]} \\ &\geq \sum_{j=1}^{k} b_{[j]}\, x_{[j]} + \sum_{j=1}^{k} (a_{[j]} - b_{[j]})\, x_{[k+1]} \\ &= \sum_{j=1}^{k} b_{[j]}\, x_{[j]} + \left(\sum_{j=1}^{k} a_{[j]} - \sum_{j=1}^{k} b_{[j]} \right) x_{[k+1]}. \end{aligned}$$

Insert this into (2.8) and gather common terms,

$$\begin{aligned} \sum_{j=1}^{k+1} a_{[j]}\, x_{[j]} &\geq \sum_{j=1}^{k} b_{[j]}\, x_{[j]} + \left(\sum_{j=1}^{k+1} a_{[j]} - \sum_{j=1}^{k} b_{[j]} \right) x_{[k+1]} \\ &\geq \sum_{j=1}^{k} b_{[j]}\, x_{[j]} + b_{[k+1]}\, x_{[k+1]} = \sum_{j=1}^{k+1} b_{[j]}\, x_{[j]}, \end{aligned}$$

where the second inequality follows from the majorization $\sum_{j=1}^{k+1} a_{[j]} \geq \sum_{j=1}^{k+1} b_{[j]}$. □

Now we are ready to bound the two norm of a row scaled matrix DZ, where Z is $m \times n$ of full column rank, and $D = \operatorname{diag}\begin{pmatrix} d_1 & \dots & d_m \end{pmatrix}$ is a non-negative $m \times m$ diagonal matrix. The obvious bound is

$$\|DZ\|_2 \leq \|D\|_2\, \|Z\|_2 = d_{[1]}\, \|Z\|_2. \tag{2.9}$$

However, the bound in Theorem 22 below, which incorporates the largest row norm of Z and several of the largest (in magnitude) diagonal elements of D, turns out to be tighter.

Theorem 22 *Let Z be a real $m \times n$ matrix with $\operatorname{rank}(Z) = n$, smallest singular value $\sigma_z = 1/\|Z^\dagger\|_2$, and largest squared row norm $\mu_z \equiv \max_{1 \leq j \leq m} \|e_j^T Z\|_2^2$. If $t \equiv \lfloor \sigma_z^2/\mu_z \rfloor$, then*

$$\|DZ\|_2^2 \leq \begin{cases} \mu_z \sum_{j=1}^{t} d_{[j]}^2 + \left(\|Z\|_2^2 - t\, \mu_z\right)\, d_{[t+1]}^2 & \text{if } \|Z\|_2^2 - t\, \mu_z \leq \mu_z \\ \mu_z \sum_{j=2}^{t+1} d_{[j]}^2 + \left(\|Z\|_2^2 - t\, \mu_z\right)\, d_{[1]}^2 & \text{otherwise.} \end{cases}$$

Proof. Let z be a $n \times 1$ vector with $\|z\|_2 = 1$ and $\|DZ\|_2 = \|DZz\|_2$. Furthermore let $z_j \equiv e_j^T Zz$, $1 \leq j \leq m$, be the elements of Zz, so that $\|Zz\|_2^2 = \sum_{j=1}^{m} z_j^2$.

Apply Lemma 20 Since $d_j^2 \geq 0$ and $z_j^2 \geq 0$, $1 \leq j \leq m$, we can apply Lemma 20 with $x_j = d_j^2$ and $b_j = z_j^2$, to obtain

$$\|DZ\|_2^2 = \|DZz\|_2^2 = \sum_{j=1}^m d_j^2\, z_j^2 = \sum_{j=1}^m b_j\, x_j \leq \sum_{j=1}^m b_{[j]}\, x_{[j]}.$$

Verify assumptions of Lemma 21 In order to apply Lemma 21 with

$$a_j = \mu_z, \quad 1 \leq j \leq t, \qquad a_{t+1} = \|Zz\|_2^2 - t\,\mu_z, \qquad a_j = 0, \quad t+2 \leq j \leq m,$$

we need show that the assumptions are satisfied, meaning all vector elements are non-negative and the majorization condition holds. Clearly $a_j \geq 0$ for $1 \leq j \leq t$ and $t+2 \leq j \leq m$. This leaves a_{t+1}. From $\operatorname{rank}(Z) = n$ follows that $\sigma_z > 0$. The definition of t implies $0 \leq t \leq \sigma_z^2/\mu_z$, so that

$$0 \leq \sigma_z^2 - t\,\mu_z = \min_{\|y\|_2=1} \|Zy\|_2^2 - t\,\mu_z \leq \|Zz\|_2^2 - t\,\mu_z = a_{t+1}.$$

Thus, all vector elements are non-negative.

To show the majorization condition, start with the Cauchy-Schwartz inequality,

$$b_j = z_j^2 = (e_j^T Z\, z)^2 \leq \|e_j^T Z\|_2^2\, \|z\|_2^2 = \|e_j^T Z\|_2^2 \leq \mu_z, \qquad 1 \leq j \leq m.$$

This yields, regardless of whether $a_{t+1} = \|Zz\|_2^2 - t\,\mu_z \leq \mu_z$ or not,

$$\sum_{j=1}^k a_{[j]} \geq \sum_{j=1}^k \mu_z \geq \sum_{j=1}^k z_{[j]}^2 = \sum_{j=1}^k b_{[j]}, \qquad 1 \leq k \leq t.$$

Moreover, for $1 \leq k \leq m-t$,

$$\sum_{j=1}^{t+k} a_{[j]} = \sum_{j=1}^t \mu_z + (\|Zz\|_2^2 - t\,\mu_z) = \|Zz\|_2^2 \geq \sum_{j=1}^{t+k} z_{[j]}^2 = \sum_{j=1}^{t+k} b_{[j]}.$$

This gives the weak majorization condition $\sum_{j=1}^k a_{[j]} \geq \sum_{j=1}^k b_{[j]}$, $1 \leq k \leq m$.

Apply Lemma 21 Since the assumptions of Lemma 21 are satisfied, we can conclude that $\sum_{j=1}^m b_{[j]}\, x_{[j]} \leq \sum_{j=1}^m a_{[j]}\, x_{[j]}$. At last, substitute into this majorization relation the expressions for a and b. If $\|Z\|_2^2 - t\,\mu_z \leq \mu_z$, then

$$\sum_{j=1}^m a_{[j]}\, x_{[j]} = \mu_z \sum_{j=1}^t d_{[j]}^2 + (\|Zz\|_2^2 - t\,\mu_z)\, d_{[t+1]}^2 \leq \mu_z \sum_{j=1}^t d_{[j]}^2 + (\|Z\|_2^2 - t\,\mu_z)\, d_{[t+1]}^2,$$

otherwise

$$\sum_{j=1}^{m} a_{[j]}\, x_{[j]} = (\|Zz\|_2^2 - t\,\mu_z)\, d_{[1]}^2 + \mu_z \sum_{j=2}^{t+1} d_{[j]}^2 \leq (\|Z\|_2^2 - t\,\mu_z)\, d_{[1]}^2 + \mu_z \sum_{j=2}^{t+1} d_{[j]}^2.$$

□

Theorem 22 is tighter than (2.9) because $d_{[j]}^2 \leq \|D\|_2^2$ implies

$$\begin{aligned} \|DZ\|_2^2 \quad &\leq \quad \begin{cases} \mu_z \sum_{j=1}^{t} d_{[j]}^2 + \left(\|Z\|_2^2 - t\,\mu_z\right)\, d_{[t+1]}^2 & \text{if } \|Z\|_2^2 - t\,\mu_z \leq \mu_z \\ \mu_z \sum_{j=2}^{t+1} d_{[j]}^2 + \left(\|Z\|_2^2 - t\,\mu_z\right)\, d_{[1]}^2 & \text{otherwise} \end{cases} \\ &\leq \quad t\mu_z \|D\|_2^2 - \left(\|Z\|_2^2 - t\,\mu_z\right)\, \|D\|_2^2 = \|D\|_2^2 \|Z\|_2^2. \end{aligned}$$

Corollary 23 *Let Z be a real $m\times n$ matrix with $Z^TZ = I_n$, and coherence $\mu_z \equiv \max_{1\leq j\leq m} \|e_j^T Z\|_2^2$. If $t \equiv \lfloor 1/\mu_z \rfloor$, then*

$$\|DZ\|_2^2 \leq \mu_z \sum_{j=1}^{t} d_{[j]}^2 + (1 - t\,\mu_z)\; d_{[t+1]}^2.$$

Proof. Applying Theorem 22 and assuming $1 - t\,\mu_z \leq \mu_z$ gives

$$\|DZ\|_2^2 \leq \mu_z \sum_{j=1}^{t} d_{[j]}^2 + (1 - t\,\mu_z)\; d_{[t+1]}^2.$$

The assumption $1 - t\,\mu_z \leq \mu_z$ is justified because

$$1 - t\,\mu_z = 1 - \lfloor 1/\mu_z \rfloor\; \mu_z \leq 1 - (1/\mu_z - 1)\mu_z = \mu_z.$$

□

2.8.2 Proof of Corollary 13

Apply Corollary 23 with $D = L^{1/2}$, $Z = Q$, $\mu_z = \mu$, and $t = \lfloor 1/\mu \rfloor$ to prove the first inequality,

$$\|Q^T L Q\|_2 = \|L^{1/2} Q\|_2^2 \leq \mu \sum_{j=1}^{t} \ell_{[j]} + (1 - t\,\mu)\, \ell_{[t+1]}.$$

As for the second inequality, $\ell_{[j]} \leq \mu$ implies

$$\mu \sum_{j=1}^{t} \ell_{[j]} + (1 - t\,\mu)\, \ell_{[t+1]} \;\; \leq t\mu^2 + (1 - t\mu)\mu = \mu.$$

If, in addition, t is an integer, then $t = 1/\mu$ and $1 - t\,\mu = 0$.

2.8.3 Proof of Corollary 16

Define the common term $\phi \equiv m\,\ln(2n/\delta)/\epsilon^2$ in both bounds, and write Corollary 8 as $c \geq 3\mu\,\phi$, and Corollary 15 as $c \geq (2\tau + \frac{2}{3}\epsilon\,\mu)\,\phi$. From $\epsilon < 1$ and $\tau \leq \mu$ follows

$$2\tau + \tfrac{2}{3}\epsilon\,\mu \leq 3\mu.$$

2.9 Existence of matrices with prescribed coherence and leverage scores

This section is the basis for Algorithm 2.6.1. We review a well-known majorization result (Theorem 24). We use it to show (Theorem 25) that, given prescribed matrix dimensions and leverage scores, there always exists a matrix Q with orthonormal columns that has the required dimensions and (squared) row norms equal to the leverage scores.

Our approach is again based on majorization, see Definition 19, and in particular on the fact that the eigenvalues of a real symmetric matrix majorize its diagonal elements.

Theorem 24 (Theorem 4.3.48 in [21]) *Let a and λ be vectors with real elements a_j and λ_j, respectively, $1 \leq j \leq m$. If λ majorizes a, then there exists a $m \times m$ real symmetric matrix with eigenvalues λ_j and diagonal elements a_j, $1 \leq j \leq m$.*

With the help of Theorem 24 we show that there exists a matrix with orthonormal columns that has prescribed leverage scores and coherence.

Theorem 25 *Given integers m and n with $m \geq n \geq 1$; and a vector ℓ with m elements ℓ_j that satisfy $0 \leq \ell_j \leq 1$ and $\sum_{j=1}^m \ell_j = n$. Then there exists a $m \times n$ matrix Q with orthonormal columns that has leverage scores $\|e_j^T Q\|_2^2 = \ell_j$, $1 \leq j \leq m$, and coherence $\mu = \max_{1 \leq j \leq m} \ell_j$.*

Proof. Let λ be a vector with m elements that satisfy $\lambda_j = 1$ for $1 \leq j \leq n$, and $\lambda_j = 0$ for $n + 1 \leq j \leq m$. We are going to construct a matrix Q by applying Theorem 24 to λ and ℓ. To this end, we first need to show that λ majorizes ℓ.

Majorization We distinguish the cases $1 \leq k \leq n$ and $n + 1 \leq k \leq m$.

Case $1 \leq k \leq n$: From $\ell_j \leq 1$ follows

$$\sum_{j=1}^{k} \lambda_j = k \geq \sum_{j=1}^{k} \ell_{[j]}.$$

Case $n+1 \leq k \leq m$: From $\ell_j \geq 0$ and $\sum_{j=1}^m \ell_j = n$ follows

$$\sum_{j=1}^k \lambda_j = n = \sum_{j=1}^k \ell_{[j]} + \sum_{j=k+1}^m \ell_{[j]} \geq \sum_{j=1}^k \ell_{[j]}.$$

Hence

$$\sum_{j=1}^k \lambda_j \geq \sum_{j=1}^k \ell_{[j]}, \qquad 1 \leq k \leq m,$$

which means that λ weakly majorizes ℓ. Since also $\sum_{j=1}^m \lambda_j = n = \sum_{j=1}^m \ell_{[j]}$, we can conclude that λ majorizes ℓ.

Construction of Q Theorem 24 implies that there exists a real symmetric matrix W with eigenvalues λ_j and diagonal elements $W_{jj} = \ell_j$, $1 \leq j \leq m$. Since W has n eigenvalues equal to one, and all other eigenvalues equal to zero, it has an eigenvalue decomposition

$$W = \hat{Q} \begin{pmatrix} I_n & 0 \\ 0 & 0 \end{pmatrix} \hat{Q}^T = QQ^T,$$

where $\hat{Q}$ is a $m \times m$ real orthogonal matrix, and $Q \equiv \hat{Q} \begin{pmatrix} I_n & 0 \end{pmatrix}^T$ has n orthonormal columns. Therefore Q has leverage scores $\|e_j^T Q\|_2^2 = e_j^T QQ^T e_j = W_{jj} = \ell_j$ and coherence $\mu = \max_{1 \leq j \leq m} \ell_j$. □

Chapter 3

Sensitivity of Leverage Scores

3.1 Introduction

We provide a brief overview of leverage scores and principal angles.

Leverage Scores

Statistical leverage scores were introduced in 1978 by Hoaglin and Welsch [20] to detect outliers when computing regression diagnostics, see also [8, 38]. To be specific, consider the least squares problem $\min_x \|Ax-b\|_2$, where A is a real $m\times n$ matrix with $\operatorname{rank}(A)=n$. The so-called *hat matrix* $H=A(A^TA)^{-1}A^T$ is the orthogonal projector onto $\operatorname{range}(A)$, and determines the *fit*, $\hat{b}=Hb$. The diagonal elements of the hat matrix are called the leverage scores of A,

$$\ell_j(A)\equiv H_{jj}, \qquad 1\leq j\leq m,$$

because $\ell_j(A)$ reflects the leverage of the j^{th} point b_j on the corresponding fit $\hat{b}_j$. To see this, suppose that $\ell_k(A)=1$ for some k. Then $\hat{b}_k=b_k$. Because b_k has maximal leverage, it completely determines the corresponding element of the *fit*. That is, k^{th} canonical vector, e_k, is in the column space of A, and b_k can be *fitted* completely without affecting *fit* of the other elements of b. In contrast, if $\ell_k(A)=0$ then b_k has zero leverage on the *fit* $\hat{b}_k$ and $\hat{b}_k=0$. That is, e_k is perpendicular to the column space of A.

Leverage scores can be stably computed from a thin QR decomposition $A=QR$, where Q is $m\times n$ with orthonormal columns, via $\ell_j(A)=\|e_j^TQ\|_2^2$. Leverage scores can also be expressed in terms of the left singular vectors of A that are associated with the non-zero singular values. In fact, for any $n\times n$ orthogonal matrix W, $\|e_j^TQ\|_2=\|e_j^TQW\|_2$ which leads us to the following definition of the leverage scores of a matrix A.

Definition 26 *Given a $m \times n$ real matrix A with $m \geq n$ and full column rank, let Q be any orthonormal basis for the column space of A. Then, the leverage scores of A are defined as*

$$\ell_i(A) = \|e_i^T Q\|_2^2, \text{ for } 1 \leq i \leq m.$$

Leverage scores are the basis for many sampling strategies in randomized matrix computations [25], including low rank approximations [13], CUR decompositions [14], subset selection [6], Nyström approximations [35], least squares [12], and matrix completion [7].

One can also define the leverage scores of the space spanned by the k dominant left singular vectors. This type of leverage scores is useful when dealing with numerically low rank matrices [13] and low rank matrix approximations [14, 26]. We let $\ell_{i,k}$ denote the leverage scores computed in this manner and provide the following definition.

Definition 27 *Given a $m \times n$ real matrix A, let $\sigma_1(A), \ldots, \sigma_{\min(m,n)}(A)$ be the singular values of A in descending order and let Q_k be any orthonormal basis for the space spanned by the k dominant left singular vectors of A where $0 < k \leq \min(m, n)$. If $\sigma_k(A) - \sigma_{k+1}(A) > 0$, then the leverage scores of A computed from the k dominant left singular vectors are defined as*

$$\ell_{i,k}(A) = \|e_i^T Q_k\|_2^2,$$

for $1 \leq i \leq m$ and $1 \leq k \leq m - 1$.

Principal Angles

The *principal angles*, also called the canonical angles, between two matrices, A and B, describe the angles between their subspaces. They give a measure of the distance between the column space of A and the column space of B. These angles can be described in terms of a singular value decomposition.

Definition 28 (§12.4.3, [17]) *Given $m \times n$ real, full column rank matrices A and B, let Q and $\tilde{Q}$ be any bases of orthonormal columns for A and B, respectively. Let $Q^T\tilde{Q} = Y\Sigma Z^T$ be a thin SVD, where Y and Z are $n \times n$ orthogonal matrices containing the left and right singular vectors of $Q^T\tilde{Q}$, respectively. Define*

$$\theta_j = \arccos(\Sigma_{j,j}) \in [0, \pi/2],$$

for $1 \leq j \leq n$, to be the principal angles between A and B. The columns of QY and $\tilde{Q}Z$ are called the principal vectors of A and B, respectively. The principal angles are ordered such that $0 \leq \theta_1 \leq \ldots \leq \theta_n \leq \pi/2$.

We also use an alternative definition for θ_n from [28].

Definition 29 (§5.15, [28]) *Given two real $m \times n$ matrices, A and B, let P_A and P_B be orthogonal projectors onto the column space of A and B, respectively, and define*

$$\theta_n = \arcsin\left(\|(I - P_B)P_A\|_2\right)$$

be the maximal principal angle between the two subspaces.

For θ_n, these two definitions are essentially the same. To see the connection, observe that

$$\cos(\theta_n) = \|Q^T\tilde{Q}\|_2 = \|QQ^T\tilde{Q}\tilde{Q}^T\|_2 = \|P_BP_A\|_2.$$

3.2 Supplemental Results

Below we present two theorems and a lemma leading up to a leverage score perturbation bound. We consider two $m \times n$ matrices of full column rank, A and B, and bound $\ell_i(B)$ in terms of $\ell_i(A)$ and the principal angles between the column spaces of A and B. Next, we bound the largest principal angle in terms of $\|B - A\|_2$ and $\|A^\dagger\|_2$. Finally, we present a second upper bound on the largest principal angle. These results are used to obtain our main results in Section 3.3.

3.2.1 Leverage Score Perturbation in Terms of Principal Angles

Theorem 30 below uses principal angles and the triangle inequality to bound $\ell_i(B)$ in terms of $\ell_i(A)$.

Theorem 30 *Let A and B be $m \times n$ real, full column rank matrices, and let θ_j, for $1 \leq j \leq n$, be the principal angles between the column spaces of A and B. Then,*

$$\ell_i(B) \leq \left(\cos(\theta_1)\sqrt{\ell_i(A)} + \sin(\theta_n)\sqrt{1 - \ell_i(A)}\right)^2, \text{ for } 1 \leq i \leq m.$$

In addition, for each i, if $\sqrt{\ell_i(A)} - \sin(\theta_n) \geq 0$ and $\cos(\theta_1) > 0$, then

$$\left(\frac{\sqrt{\ell_i(A)} - \sin(\theta_n)}{\cos(\theta_1)}\right)^2 \leq \ell_i(B).$$

Proof. See Section 3.6.1. □

Theorem 31 below uses principal angles to bound $|\ell_i(A) - \ell_iB|$.

Theorem 31 *Let A and B be $m \times n$ real, full column rank matrices, and let θ_j, for $1 \leq j \leq n$, be the principal angles between the column spaces of A and B. Then,*

$$|\ell_i(B) - \ell_i(A)| \leq 2\sqrt{\ell_i(A)}\sqrt{1-\ell_i(A)}\sin(\theta_n) + \sin(\theta_n)^2$$

for $1 \leq i \leq m$.

Proof. See Section 3.6.2. □

Theorem 30 and Theorem 31 show that the leverage scores of A and B are close if the principal angles between their column spaces are small. If A and B have the same column space, then all of the principal angles are zero and Theorem 30 and Theorem 31 confirm that their leverage scores are equal. On the other hand, if A and B are completely orthogonal to each other, then Theorem 30 implies that $\ell_i(B) + \ell_i(A) \leq 1$.

3.2.2 Upper Bounds for the Largest Principal Angle

In Theorem 32, we show a bound by Wedin on the maximal principal angle between A and B in terms of $\|B - A\|_2$, and $\|A^\dagger\|_2$.

Theorem 32 *[41, Equation 4.4] Let A and B be $m \times n$ real, full column rank matrices with $m > n$. Then,*

$$\sin(\theta_n) \leq \|B - A\|_2 \|A^\dagger\|_2.$$

Proof. See Section 3.6.3. □

Theorem 32 shows that the largest principal angle is small if the difference between A and B and $\|A^\dagger\|_2$ are small.

In Theorem 33, we show a bound on the maximal angle between the space spanned by the top k left singular vectors of A, and the space spanned by the top k left singular vectors of B. We begin by setting up a little notation. For any $m \times n$ matrix A, let $\sigma_i(A)$, for $1 \leq i \leq \min(m, n)$, denote the i^{th} singular value of A where $\sigma_1(A) \geq \ldots \geq \sigma_{\min(m,n)}(A) \geq 0$.

This bound is a modification of [40, Equation 3.1].

Corollary 33 ([40]) *Let A and B be $m \times n$ matrices and let Q_k and $\tilde{Q}_k$ be the k dominant left singular vectors of A and B, respectively. If $2\|B - A\|_2 \leq \sigma_k(A) - \sigma_{k+1}(A)$, then*

$$\sin(\theta_k(Q_k, \tilde{Q}_k)) \leq \frac{\|B - A\|_2}{\sigma_k(A) - \sigma_{k+1}(A) - \|B - A\|_2} \leq \frac{2\|B - A\|_2}{\sigma_k(A) - \sigma_{k+1}(A)}$$

for $1 \leq k \leq \min(m, n) - 1$.

Proof. See section 3.6.4. □

Corollary 33 shows that the largest principal angle between Q_k and $\tilde{Q}_k$ is small if the relative difference between A and B is small compared to the singular value gap between the k^{th} and $k+1^{\text{th}}$ singular values of A.

In Lemma 34, we present a second upper bound on the largest principal angle between the column spaces of A and B. We use this lemma to remove the condition $\cos(\theta_1) > 0$ from Theorem 30 in the proof for Corollary 35.

Lemma 34 *Given $m \times n$, real, full column rank matrices A and B, let θ_j, for $1 \leq j \leq n$, be the principal angles between the columns spaces of A and B. If $\|B - A\|_2 \leq \|A^\dagger\|_2^{-1}$, then $\theta_n < \pi/2$.*

Proof. See Section 3.6.5. □

3.3 Leverage Score Perturbation in terms of Matrix Perturbation

Here we combine our previous results to obtain leverage score perturbation bounds. First, we give two sided bounds for both $\ell_i(B)$ and $\ell_{i,k}(B)$.

Corollary 35 *Given $m \times n$ real matrices A and B with $m > n$ such that A has full column rank and $\|B - A\|_2 < \|A^\dagger\|_2^{-1}$. Then,*

$$\ell_i(B) \leq \left(\sqrt{\ell_i(A)} + \|A^\dagger\|_2 \|B - A\|_2 \sqrt{1 - \ell_i(A)}\right)^2, \text{ for } 1 \leq i \leq m.$$

In addition, for each $1 \leq i \leq m$, if $\sqrt{\ell_i(A)} - \|A^\dagger\|_2 \|B - A\|_2 \geq 0$, then

$$\left(\sqrt{\ell_i(A)} - \|A^\dagger\|_2 \|B - A\|_2\right)^2 \leq \ell_i(B).$$

Proof. See Section 3.6.6. □

Corollary 36 *Given $m \times n$ real matrices A and B. If $2\|B - A\|_2 \leq \sigma_k(A) - \sigma_{k+1}(A)$, then*

$$\ell_{i,k}(B) \leq \left(\sqrt{\ell_{i,k}(A)} + \frac{\|B - A\|_2}{\sigma_k(A) - \sigma_{k+1}(A) - \|B - A\|_2} \sqrt{1 - \ell_{i,k}(A)}\right)^2$$

for $1 \leq i \leq m$ and $1 \leq k \leq \min(m, n) - 1$. In addition, for each $1 \leq i \leq m$, if $\sqrt{\ell_{i,k}(A)} - \frac{\|B-A\|_2}{\sigma_k(A)-\sigma_{k+1}(A)-\|B-A\|_2} \geq 0$, then

$$\left(\sqrt{\ell_{i,k}(A)} - \frac{\|B - A\|_2}{\sigma_k(A) - \sigma_{k+1}(A) - \|B - A\|_2}\right)^2 \leq \ell_{i,k}(B).$$

Proof. See Section 3.6.7. □

Corollary 35 shows that the leverage scores of A and B are close if $\|A^\dagger\|_2$ and the difference between A and B are small. Corollary 36 shows that if $\|B - A\|_2$ is small with respect to the k^{th} singular value gap of A, then the leverage scores of A and B, as computed by the top k left singular vectors, are close.

While trying to examine Corollary 35 and Corollary 36, we found that the results are somewhat difficult to plot. Even with logarithmically scaled axes, the upper and lower bounds are indistinguishable since their difference is much smaller than their magnitude. To aid in examining our results, we use Theorem 31 along with Theorem 32 and Corollary 33 to obtain bounds on $|\ell_i(A) - \ell_i(B)|$ and $|\ell_{i,k}(A) - \ell_{i,k}(B)|$. While the bounds below are slightly less descriptive than Corollary 35 and Corollary 36, they have the advantage that one can actually see the results in a plot.

Corollary 37 *Given $m \times n$ real matrices A and B with $m > n$ such that A has full column rank and $\|B - A\|_2 < \|A^\dagger\|_2^{-1}$. Then,*

$$|\ell_i(B) - \ell_i(A)| \leq 2\sqrt{\ell_i(A)}\sqrt{1-\ell_i(A)}\|A^\dagger\|_2\|B-A\|_2 + \Big(\|A^\dagger\|_2\|B-A\|_2\Big)^2, \text{ for } 1 \leq i \leq m.$$

and if $\ell_i(A) > 0$, then

$$\frac{|\ell_i(B) - \ell_i(A)|}{\ell_i(A)} \leq 2\sqrt{\frac{1-\ell_i(A)}{\ell_i(A)}}\|A^\dagger\|_2\|B-A\|_2 + \left(\frac{\|A^\dagger\|_2\|B-A\|_2}{\sqrt{\ell_i(A)}}\right)^2, \text{ for } 1 \leq i \leq m.$$

Proof. See Section 3.6.8. □

Corollary 38 *Given $m \times n$ real matrices A and B such that*

$$2\|B - A\|_2 < \sigma_k(A) - \sigma_{k+1}(A)$$

. Let

$$\gamma = \tfrac{\|B-A\|_2}{\sigma_k(A)-\sigma_{k+1}(A)-\|B-A\|_2} < \frac{2\|B-A\|_2}{\sigma_k(A)-\sigma_{k+1}(A)}.$$

Then,

$$|\ell_{i,k}(B) - \ell_{i,k}(A)| \leq 2\sqrt{\ell_{i,k}(A)}\sqrt{1-\ell_{i,k}(A)}\gamma + \gamma^2,$$

for $1 \leq i \leq m$ and $1 \leq k \leq \min(m,n) - 1$. And if $\ell_{i,k}(A) > 0$, then

$$\frac{|\ell_{i,k}(B) - \ell_{i,k}(A)|}{\ell_i(A)} \leq 2\sqrt{\frac{1-\ell_{i,k}(A)}{\ell_{i,k}(A)}}\gamma + \left(\frac{\gamma}{\sqrt{\ell_{i,k}(A)}}\right)^2, \text{ for } 1 \leq i \leq m.$$

Proof. See Section 3.6.9. □

3.3.1 Leverage score bound for givens rotation

In this subsection, we bound the leverage scores of a matrix $G_{i,j}A$ where $G_{i,j}$ is a Givens matrix that rotates row i with row j by ϕ radians.

Theorem 39 *Given the $m \times n$ real, full columns rank matrix A with $m > n$, let $G_{i,j}$ be the $m \times m$ Givens rotation matrix that rotates row i with row j by ϕ radians. Then,*

$$\ell_i(G_{i,j}A) \leq \cos(\phi)^2\ell_i(A) + \sin(\phi)^2\ell_j(A) + |\sin(2\phi)|\sqrt{\ell_i(A)\ell_j(A)}$$

and

$$\ell_i(G_{i,j}A) \geq \cos(\phi)^2\ell_i(A) + \sin(\phi)^2\ell_j(A) - |\sin(2\phi)|\sqrt{\ell_i(A)\ell_j(A)}.$$

Proof. See Section 3.6.10. □

This bound shows that the change in leverage score is small if the rotation angle of $G_{i,j}$ is small.

3.4 Experiments

In this section, we examine the tightness and behavior of Corollary 37 with a few carefully constructed examples. In all of the examples, we set $m = 500$ and $n = 15$ and we use Algorithm 2.6.1 to construct the unperturbed matrix A with orthonormal columns. We use this algorithm because it allows us to create a matrix with a wide distribution of leverage scores. It is important to note that since Corollary 37 depends on $\|B - A\|_2\|A^\dagger\|_2$, we would not gain any insight on the performance of the bound by considering matrices with different singular values. It is for this reason that we only consider matrices A with orthonormal columns.

In Figures 3.1, 3.2 and 3.3, the horizontal coordinate axis represents the leverage scores of A (the unperturbed leverage scores) and the vertical coordinate axis represents either the absolute or relative magnitude of the change in leverage scores. The dots represent the absolute difference between the leverage scores of A and B, $|\ell_i(B) - \ell_i(A)|$, and the black line represents the bound from Corollary 37.

Example 1

Here we present two plots to show how well Corollary 37 predicts the behavior of the leverage scores of $B = A + E$ where E is a matrix whose entries are independent realizations of a normal random variable.

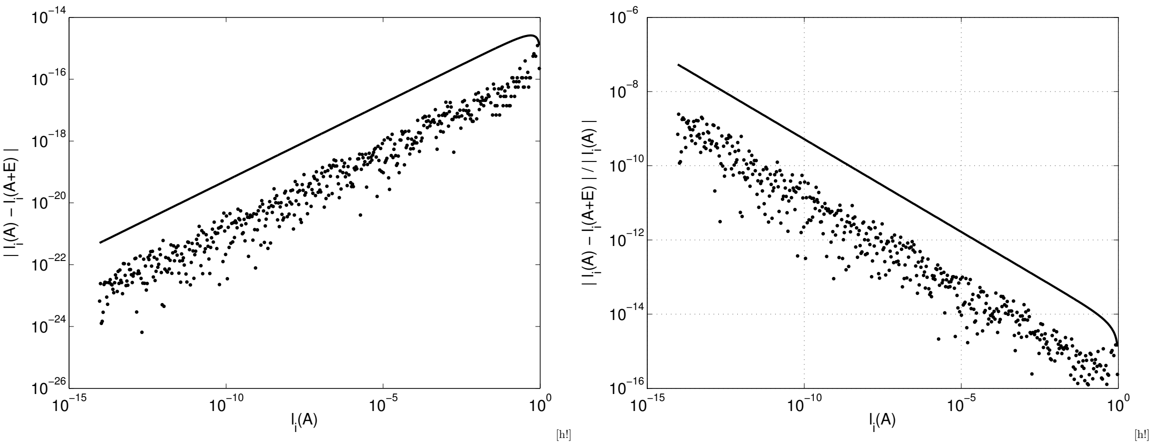


Figure 3.1: Here, A has orthonormal columns and thus $\|A\|_2 = \sigma_n(A) = 1$ and $\|E\|_2 \approx 2.6 * 10^{-15}$. The entries of E have mean 0 and variance 10^{-16}. On the left, we plot the absolute change in the leverage scores and the absolute bound from Corollary 37, and on the right we plot the relative change in the leverage scores against the relative bound from Corollary 37.

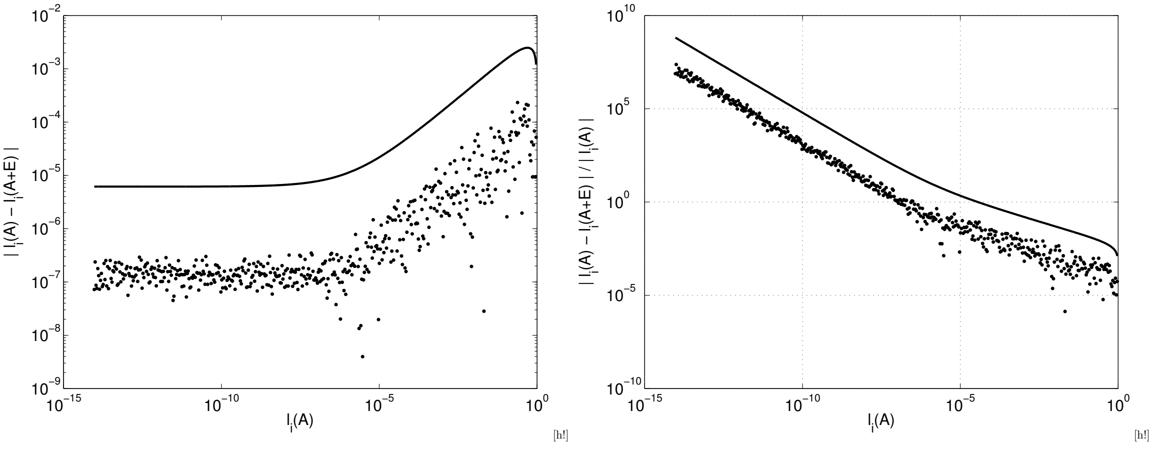


Figure 3.2: Here, A has orthonormal columns and thus $\|A\|_2 = \sigma_n(A) = 1$ and $\|E\|_2 \approx 2.5 * 10^{-3}$. The entries of E have mean 0 and variance 10^{-4}. On the left, we plot the absolute change in the leverage scores and the absolute bound from Corollary 37, and on the right we plot the relative change in the leverage scores against the relative bound from Corollary 37.

In Figures 3.1 and 3.2, we see that Corollary 37 accurately bounds the absolute difference between the leverage scores of A and B. In particular, we see the following behaviors.

- **Small leverage scores have a larger relative perturbation.** When $\|B-A\|_2$ is small, as in Figure 3.1, the relative bound on the leverage score perturbation is $\mathcal{O}\left(\sqrt{\frac{1-\ell_i(A)}{\ell_i(A)}}\right)$, which is larger for small $\ell_i(A)$. On the other hand, when $\|B-A\|_2$ is large, as in Figure 3.2, the relative perturbation bound on the small leverage scores is $\mathcal{O}\left(\left(\frac{\|B-A\|_2\|A^\dagger\|_2}{\ell_i(A)}\right)^2\right)$, which is again larger for small $\ell_i(A)$.

- **The absolute perturbation bound for Small leverage scores is flat when $\|B-A\|_2$ is large.** When $\|B-A\|_2$ is large, as in Figure 3.2, the $\left(\|A^\dagger\|_2\|B-A\|_2\right)^2$ term in Corollary 37 becomes dominant. Thus, the absolute perturbation bound for small leverage scores below a certain threshold is flat as it does not significantly depend on $\ell_i(A)$. The intuition for this is that even a zero leverage score can be increased by a certain amount for a given $\|B-A\|_2$ and this amount becomes significant for small leverage scores.

- **Large leverage scores have a small relative perturbation even when $\|B-A\|_2\|A^\dagger\|_2$ is large.** From Figure 3.2 we can see that even when $\|B-A\|_2\|A^\dagger\|_2 \approx 2.5*10^{-3}$, the relative perturbation of the large leverage scores remains small. This is because, for $\ell_i(A) \approx 1$, the relative perturbation bound is $\mathcal{O}(\|B-A\|_2\|A^\dagger\|_2)$.

At first glance, Corollary 37 does not look particularly tight as there is a large gap between the experimental data and the bound. This is because the elements of the perturbation, E, were sampled from a normal random variable and thus its affects are spread out among all of the leverage scores.

Example 2

By carefully constructing E, it is possible to focus its affects on a particular leverage score. In Figure 3.3, we have constructed E in a way that attempts to maximize $\frac{\ell_{100}(B)-\ell_{100}(A)}{\|B-A\|_2}$.[1]

Figure 3.3 shows that the bound from Corollary 37 can be very tight for particular leverage scores.

A note on Corollary 38. We don't show experiments for Corollary 38 since the plots look qualitatively identical to those shown above. To see why, recall that Corollary depends on

$$\gamma \leq 2\frac{\|B-A\|_2}{\sigma_k(A)-\sigma_{k+1}(A)}$$

[1] The algorithm used to construct this example is similar to a gradient descent method where we examine $\partial\ell_i(G_{i,j}A)/\partial\|G_{i,j}A\|_2$.

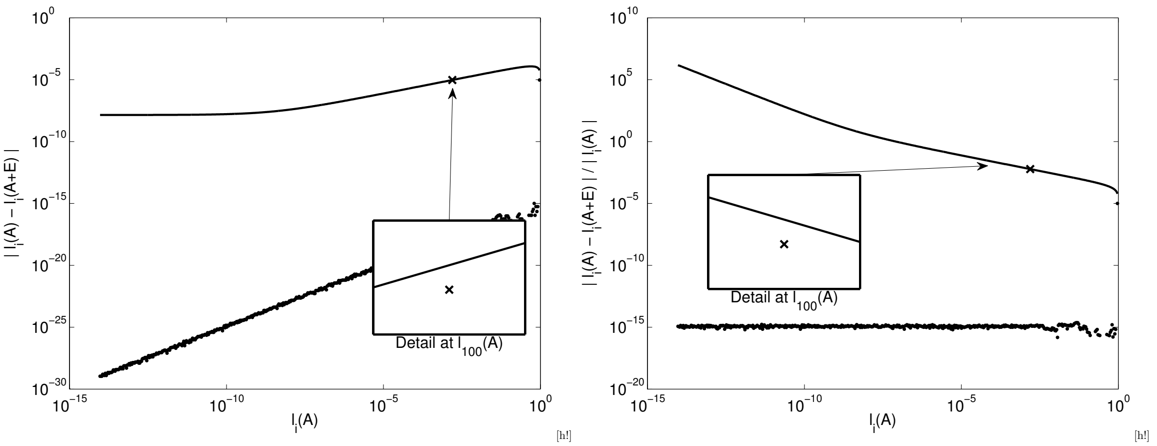


Figure 3.3: Here, A has orthonormal columns and thus $\|A\|_2 = \|A^{\dagger}\|_2 = 1$ and $\|E\|_2 \approx 1.2 * 10^{-4}$. The 'x' represents $|\ell_{100}(A) - \ell_{100}(B)|$ and is $\sim 5 * 10^{-10}$ below the absolute bound implied by Corollary 37. On the left, we plot the absolute change in the leverage scores and the absolute bound from Corollary 37, and on the right we plot the relative change in the leverage scores against the relative bound from Corollary 37.

and that Corollary 37 depends on

$$\frac{\|B - A\|_2}{\sigma_n(A)}.$$

Thus, one can see that the main difference between the two corollaries is that Corollary 37 depends on the smallest singular value of A whereas Corollary 38 depends on the k^{th} singular value gap.

3.5 Conclusion

We have proven multiple perturbation bounds for leverage scores. Theorem 30 shows that if the principal angles between A and B are small, then the leverage scores of A and B are close. Corollary 35 and Corollary 37 show the leverage scores of A and B are close if $\|A^{\dagger}\|_2$ and the two-norm difference between A and B are small. Finally, Corollary 36 and Corollary 38 show that if two-norm difference between A and B is small with respect to the k^{th} singular value gap, then the leverage scores of A and B, as computed from the top k left singular vectors, are close.

3.6 Proofs

3.6.1 Proof of Theorem 30

Let Q and $\tilde{Q}$ be the respective column spaces of A and B and let Y and Z be $n \times n$ matrices with orthonormal columns as described in Definition 28. Define the $m \times m$ orthogonal matrix $C = \begin{bmatrix} QY & Q_\perp \end{bmatrix}$. Then $C^T \tilde{Q} Z = \begin{bmatrix} \Sigma \\ D \end{bmatrix}$, where $D = Q_\perp^T \tilde{Q} Z$, hence $\tilde{Q} Z = C \begin{bmatrix} \Sigma \\ D \end{bmatrix} = QY\Sigma + Q_\perp D$. Since the leverage scores of B do not depend on the choice of basis (Definition 26),

$$\ell_i(B) = \|e_i^T \tilde{Q} Z\|_2^2 = \|e_i^T (QY\Sigma + Q_\perp D)\|_2^2.$$

The triangle and Cauchy-Schwarz inequality gives us

$$\ell_i(B) \leq \left(\|e_i^T Q\|_2 \|\Sigma\|_2 + \|e_i^T Q_\perp\|_2 \|D\|_2\right)^2 = \left(\sqrt{\ell_i(A)} \cos(\theta_1) + \|e_i^T Q_\perp\|_2 \|D\|_2\right)^2.$$

Since C is an orthogonal matrix,

$$1 = \|e_i^T C\|_2^2 = \|e_i^T Q\|_2^2 + \|e_i^T Q_\perp\|_2^2 = \ell_i(A) + \|e_i^T Q_\perp\|_2^2$$

implies

$$\ell_i(B) \leq \left(\sqrt{\ell_i(A)} \cos(\theta_1) + \sqrt{1 - \ell_i(A)} \|D\|_2\right)^2.$$

Also, since $C^T \tilde{Q}$ has orthonormal columns, we have $\Sigma^2 + D^T D = I$, and it follows that

$$\|D\|_2 = \sqrt{\|I - \Sigma^2\|_2} = \sqrt{1 - \cos(\theta_n)^2} = \sin(\theta_n).$$

Finally,

$$\ell_i(B) \leq \left(\cos(\theta_1)\sqrt{\ell_i(A)} + \sin(\theta_n)\sqrt{1 - \ell_i(A)}\right)^2.$$

Reversing the roles of A and B gives

$$\ell_i(A) \leq \left(\cos(\theta_1)\sqrt{\ell_i(B)} + \sin(\theta_n)\sqrt{1 - \ell_i(B)}\right)^2 \leq \left(\cos(\theta_1)\sqrt{\ell_i(B)} + \sin(\theta_n)\right)^2.$$

Rearranging the terms gives

$$\sqrt{\ell_i(B)} \geq \left(\frac{\sqrt{\ell_i(A)} - \sin(\theta_n)}{\cos(\theta_1)}\right).$$

In addition, for $1 \leq i \leq m$, if $\sqrt{\ell_i(A)} - \sin(\theta_n) \geq 0$,

$$\ell_i(B) \geq \left(\frac{\sqrt{\ell_i(A)} - \sin(\theta_n)}{\cos(\theta_1)} \right)^2 .$$

3.6.2 Proof of Theorem 31

Let Q and $\tilde{Q}$ be the respective column spaces of A and B and let Y and Z be $n \times n$ matrices with orthonormal columns as described in Definition 28. Define the $m \times m$ orthogonal matrix $C = \begin{bmatrix} QY & Q_\perp \end{bmatrix}$. Then $C^T \tilde{Q} Z = \begin{bmatrix} \Sigma \\ D \end{bmatrix}$, where $D = Q_\perp^T \tilde{Q} Z$, hence $\tilde{Q} Z = C \begin{bmatrix} \Sigma \\ D \end{bmatrix} = QY\Sigma + Q_\perp D$. Since the leverage scores of B do not depend on the choice of basis (Definition 26),

$$\ell_i(B) = \|e_i^T \tilde{Q} Z\|_2^2 = \|e_i^T (QY\Sigma + Q_\perp D)\|_2^2.$$

We begin with

$$|\ell_i(B) - \ell_i(A)| = |\|e_i^T (QY\Sigma + Q_\perp D)\|_2^2 - \|e_i^T Q\|_2^2|.$$

Note that $\|e_i^T QY\|_2 = \|e_i^T Q\|_2$. We now consider two cases.

- **Case 1:** $\|e_i^T (QY\Sigma + Q_\perp D)\|_2^2 \geq \|e_i^T Q\|_2^2$**.** Using the triangle inequality, and the fact that $\|\Sigma\|_2^2 \leq 1$, gives us

$$\begin{aligned} |\ell_i(B) - \ell_i(A)| &\leq \left(\|e_i^T Q\|_2 \|\Sigma\|_2 + \|e_i^T Q_\perp\|_2 \|D\|_2\right)^2 - \|e_i^T Q\|_2^2 \\ &\leq \|e_i^T Q\|_2^2 (\|\Sigma\|_2^2 - 1) + 2\|e_i^T Q\|_2 \|e_i^T Q_\perp\|_2 \|\Sigma\|_2 \|D\|_2 + \|e_i^T Q_\perp\|_2^2 \|D\|_2^2 \\ &\leq 2\|e_i^T Q\|_2 \|e_i^T Q_\perp\|_2 \|\Sigma\|_2 \|D\|_2 + \|e_i^T Q_\perp\|_2^2 \|D\|_2^2. \end{aligned}$$

- **Case 2:** $\|e_i^T (QY\Sigma + Q_\perp D)\|_2^2 < \|e_i^T Q\|_2^2$**.** Using the reverse triangle inequality and rearranging terms gives us

$$\begin{aligned} |\ell_i(B) - \ell_i(A)| &\leq \|e_i^T Q\|_2^2 - \|e_i^T (QY\Sigma + Q_\perp D)\|_2^2 \\ &\leq \|e_i^T Q\|_2^2 - |\|e_i^T QY\Sigma\|_2 - \|e_i^T Q_\perp D\|_2|^2 \\ &\leq \|e_i^T Q\|_2^2 - (\|e_i^T QY\Sigma\|_2 - \|e_i^T Q_\perp D\|_2)^2 \\ &\leq \|e_i^T Q\|_2^2 - \|e_i^T QY\Sigma\|_2^2 + 2\|e_i^T QY\Sigma\|_2 \|e_i^T Q_\perp D\|_2 - \|e_i^T Q_\perp D\|_2^2 \\ &\leq \|e_i^T Q\|_2^2 - \|e_i^T QY\Sigma\|_2^2 \|\Sigma^{-1}\|_2^2 \|\Sigma^{-1}\|_2^{-2} + 2\|e_i^T QY\Sigma\|_2 \|e_i^T Q_\perp D\|_2 \\ &\leq \|e_i^T Q\|_2^2 (1 - \|\Sigma^{-1}\|_2^{-2}) + 2\|e_i^T Q\|_2 \|\Sigma\|_2 \|e_i^T Q_\perp\|_2 \|D\|_2. \end{aligned}$$

  Note that $\|B - A\|_2 < \|A^\dagger\|_2^{-1}$ ensures that Σ is invertible.

Thus,

$$|\ell_i(B) - \ell_i(A)| \leq \|e_i^T Q\|_2^2(1 - \|\Sigma^{-1}\|_2^{-2}) + 2\|e_i^T Q\|_2\|e_i^T Q_\perp\|_2\|\Sigma\|_2\|D\|_2 + \|e_i^T Q_\perp\|_2^2\|D\|_2^2.$$

This simplifies to

$$|\ell_i(B) - \ell_i(A)| \leq \ell_i(A)(1 - \cos(\theta_n)^2) + 2\sqrt{\ell_i(A)}\|e_i^T Q_\perp\|_2 \cos(\theta_1)\|D\|_2 + \|e_i^T Q_\perp\|_2^2\|D\|_2^2.$$

Since C is an orthogonal matrix,

$$1 = \|e_i^T C\|_2^2 = \|e_i^T Q\|_2^2 + \|e_i^T Q_\perp\|_2^2 = \ell_i(A) + \|e_i^T Q_\perp\|_2^2$$

implies that

$$|\ell_i(B) - \ell_i(A)| \leq \ell_i(A)(1 - \cos(\theta_n)^2) + 2\sqrt{\ell_i(A)}\sqrt{1-\ell_i(A)}\cos(\theta_1)\|D\|_2 + (1-\ell_i(A))\|D\|_2^2.$$

Since $C^T\tilde{Q}$ has orthonormal columns, we have $\Sigma^2 + D^T D = I$, and it follows that

$$\|D\|_2 = \sqrt{\|I - \Sigma^2\|_2} = \sqrt{1 - \cos(\theta_1)^2} = \sin(\theta_1).$$

This gives,

$$|\ell_i(B) - \ell_i(A)| \leq \ell_i(A)(1 - \cos(\theta_n)^2) + 2\sqrt{\ell_i(A)}\sqrt{1-\ell_i(A)}\cos(\theta_1)\sin(\theta_n) + (1-\ell_i(A))\sin(\theta_1)^2.$$

Finally, since $\sin(\theta_1) < \sin(\theta_n)$ and $\sin(\theta_n)^2 + \cos(\theta_n)^2 = 1$ we have,

$$|\ell_i(B) - \ell_i(A)| \leq 2\sqrt{\ell_i(A)}\sqrt{1-\ell_i(A)}\sin(\theta_n) + \sin(\theta_n)^2.$$

3.6.3 Proof of Theorem 32 from [41]

Here, for completeness, we restate the proof found in [41, Page 278]. Start with Definition 29 for θ_n. Observe that [41, Equation 4.2] impies

$$(I - P_B)(B - A) = (I - P_B)P_A A$$

because $P_A = AA^\dagger$. Multiplying on the right by $A^\dagger$ gives

$$(I - P_B)(B - A)A^\dagger = (I - P_B)P_A.$$

Thus,

$$\sin(\theta_n(A,B)) = \|(I-P_B)P_A\|_2 = \|(I-P_B)(B-A)A^\dagger\|_2 \leq \|B-A\|_2\|A^\dagger\|_2.$$

3.6.4 Proof of Corollary 33

Let Q_k and $\tilde{Q}_k$ be bases of orthonormal columns for the column space of A_k and B_k, respectively. Using [40, Inequality 3.1], one obtains the following statement. *If there exists $\alpha \geq 0$ and $\delta > 0$ such that,*

$$\sigma_k(B) \geq \alpha + \delta \qquad \textit{and} \qquad \sigma_{k+1}(A) \leq \alpha,$$

then

$$\theta_n(Q_k, \tilde{Q}_k) = \theta_n(A_k, B_k) \leq \frac{\epsilon}{\delta} \leq \frac{\|B-A\|_2}{\delta}.$$

Choose $\alpha = \sigma_{k+1}(A)$ and $\delta = \sigma_k(A) - \sigma_{k+1}(A) - \|B-A\|_2$. The inequality

$$|\sigma_i(A) - \sigma_i(B)| \leq \|B-A\|_2,$$

for $1 \leq i \leq \min(m,n)$, from [17, Corollary 8.6.2] proves that $\sigma_k(B) \geq \alpha+\delta$ holds for our choice of δ. This gives the desired result.

Note that the result still holds true for $\|B-A\|_2 < \sigma_k(A) - \sigma_{k+1}(A)$, however

$$\frac{\|B-A\|_2}{\sigma_k(A) - \sigma_{k+1}(A) - \|B-A\|_2} \leq 1$$

is only true for $2\|B-A\|_2 \leq \sigma_k(A) - \sigma_{k+1}(A)$.

3.6.5 Proof of Lemma 34

We prove the contrapositive; if $\theta_n = \pi/2$, then $\|B-A\|_2 > \sigma_n(A)$.
If $\theta_n = \pi/2$, then $\Sigma_{n,n} = \cos(\theta_n) = 0$. This implies that $Q^T\tilde{Q}$ has a zero singular value and thus, A^TB also has a zero singular value. Thus, there exists a vector x, where $\|x\|_2 = 1$, such that $x^TB^TAx = 0$. Additionally, since A and B are full column rank, Ax and Bx are non-zero. Hence, Pythagoras implies the strict inequality, $\|(B-A)x\|_2 = \|Bx - Ax\|_2 > \|Ax\|_2$. Using $\|B-A\|_2 \geq \|(B-A)x\|_2$ and $\|Ax\|_2 \geq \sigma_n(A)$ gives us our desired result,

$$\|B-A\|_2 > \sigma_n(A).$$

3.6.6 Proof of Corollary 35

Start with Theorem 30. We begin by removing the condition that $\cos(\theta_1) > 0$ since Lemma 34 ensures that this is always true when $\|B-A\|_2 \leq \sigma_n(A)$. Finally, Theorem 32 allows us to substitute $\arcsin(\|A^\dagger\|_2\|B-A\|_2)$ for θ_n.

3.6.7 Proof of Corollary 36

Start with Theorem 30 with Q_k in place of A and $\tilde{Q}_k$ in place of B where Q_k and $\tilde{Q}_k$ are bases of orthonormal columns for the column space of A_k and B_k, respectively. Use Corollary 33 to substitute

$$\frac{\|B-A\|_2}{\sigma_k(A)-\sigma_{k+1}(A)-\|B-A\|_2}$$

for θ_n and note that $2\|B-A\|_2 < \sigma_k(A)-\sigma_{k+1}(A)$ implies that $\cos(\theta_1) \geq \cos(\theta_n) > 0$.

3.6.8 Proof of Corollary 37

Start with Theorem 31 and use Theorem 32 to substitute $\arcsin(\kappa(A)\varepsilon)$ for θ_n.

3.6.9 Proof of Corollary 38

Start with Theorem 31 with Q_k in place of A and $\tilde{Q}_k$ in place of B where Q_k and $\tilde{Q}_k$ are bases of orthonormal columns for the column space of A_k and B_k, respectively. Use Theorem 33 to substitute γ for θ_n.

3.6.10 Proof of Theorem 39

Let Q be a basis of orthonormal columns for A. Then, $G_{i,j}Q$ is a basis for $G_{i,j}A$ and

$$G_{i,j}Q = \cos(\phi)e_i^TQ + \sin(\phi)e_j^tQ.$$

From here we can compute the i^{th} leverage score of $G_{i,j}Q$,

$$\ell_i(G_{i,j}Q) = \|\cos(\phi)e_i^TQ+\sin(\phi)e_j^tQ\|_2^2 = \cos(\phi)^2\|e_i^TQ\|_2^2+\sin(\phi)^2\|e_j^TQ\|_2^2+2\cos(\phi)\sin(\phi)e_i^TQQ^Te_j.$$

Using the double angle formula for sin and recognizing that the row norms are leverage scores yields,

$$\ell_i(G_{i,j}Q) = \cos(\phi)^2\ell_i(Q) + \sin(\phi)^2\ell_j(Q) + \sin(2\phi)e_i^TQQ^Te_j.$$

Using the Cauchy-Schwarz inequality gives us our upper bound,

$$\begin{aligned}
\ell_i(G_{i,j}Q) &\leq \cos(\phi)^2\ell_i(Q) + \sin(\phi)^2\ell_j(Q) + |\sin(2\phi)|\,|e_i^TQQ^Te_j| \\
&\leq \cos(\phi)^2\ell_i(Q) + \sin(\phi)^2\ell_j(Q) + |\sin(2\phi)|\,\|e_i^TQ\|_2\|e_j^TQ\|_2 \\
&= \cos(\phi)^2\ell_i(Q) + \sin(\phi)^2\ell_j(Q) + |\sin(2\phi)|\,\sqrt{\ell_i(Q)\ell_j(Q)}.
\end{aligned}$$

A similar process gives us our lower bound,

$$\begin{aligned}
\ell_i(G_{i,j}Q) &\geq \cos(\phi)^2\ell_i(Q) + \sin(\phi)^2\ell_j(Q) - |\sin(2\phi)|\,|e_i^TQQ^Te_j| \\
&\geq \cos(\phi)^2\ell_i(Q) + \sin(\phi)^2\ell_j(Q) - |\sin(2\phi)|\,\|e_i^TQ\|_2\|e_j^TQ\|_2 \\
&= \cos(\phi)^2\ell_i(Q) + \sin(\phi)^2\ell_j(Q) - |\sin(2\phi)|\,\sqrt{\ell_i(Q)\ell_j(Q)}.
\end{aligned}$$

Chapter 4

kappa_SQ

4.1 Introduction

We wrote the kappa_SQ software package to assist us with our research on various algorithms for uniform row sampling. In our research, a $m \times n$ matrix Q with orthonormal columns and $m \geq n$ is sampled by a row sampling matrix S to create the $c \times n$ sampled matrix SQ. We then address the question, given $\eta > 0$, what is the probability that $\operatorname{rank}(SQ) = n$ and the two-norm condition number $\kappa(SQ) = \|SQ\|_2 \|(SQ)^\dagger\|_2 \leq 1 + \eta$? This question is important due to its applications to randomized least squares solvers such as LSRN [27] and, in particular, the *Blendenpik* algorithm [1].

The *Blendenpik* algorithm uses randomized sampling to solve an overdetermined least-squares problem $\min_x \|Ax - b\|_2$ faster than LAPACK. It starts by finding the QR factorization, $Q_s R_s = SA$, of the randomly sampled matrix SA and then, if SA has full colun rank, solves the preconditioned least squares problem $\min_z \|AR_s^{-1} z - b\|_2$ via LSQR. The solution to the original least squares problem can be found by solving a much smaller linear system with coefficient matrix R_s. The key to this method is that if $\kappa(AR_s^{-1}) \approx 1$, then LSQR will converge quickly.

The connection between our work, kappa_SQ and the *Blendenpik* algorithm is that if SA is full rank, then $\kappa(SQ) = \kappa(AR_s^{-1})$. This means that sampling rows from A is, conceptually, the same as sampling rows from Q and that $\kappa(AR_s^{-1})$ depends only on the columns space of A (and the sampling matrix). Thus, it suffices to examine the behavior of $\kappa(SQ)$.

This code examines $\kappa(SQ)$ in two main ways. First, it can perform numerical experiments where $\kappa(SQ)$ is measured. And second, our code can plot bounds for $\kappa(SQ)$. In the literature, these bounds are often expressed in terms of two matrix properties that have been shown to be related to row sampling, leverage scores, and coherence.

Leverage scores were first introduced in 1978 by Hoaglin and Welsch [20] to detect outliers when computing regression diagnostics. They give a measurement of the distribution of the elements in an orthonormal basis. The leverage scores of a matrix A are defined in terms of any orthonormal basis, Q, for the column space of A.

Definition 40 *The leverage scores of the real $m \times n$ matrix A with $m \geq n$ are*

$$\ell_j(A) = \ell_j(Q) \equiv \|e_j^T Q\|_2^2, \qquad 1 \leq j \leq m.$$

Since leverage scores are simply row norms from matrices with orthonormal columns, the inequality $0 \leq \ell_i(A) \leq 1$ holds and $\sum_{i=1}^m \ell_i(A) = n$. If $\ell_i(A) = 1$ then the i'th row contains all of the information for a particular column. On the other hand, if $\ell_i(A) = 0$, then the i'th row of A is zero and contains no data. Thus, leverage scores give a quantification of the importance of each row with respect to sampling. We use leverage scores as an input to both generate test matrices and to bound the condition number of a sampled matrix. Our code for computing leverage scores is `leverageScores.m`.

In our work, coherence is simply the largest leverage score.

Definition 41 (Definition 3.1 in [1], Definition 1.2 in [7]) *The coherence of A is*

$$\mu(A) \equiv \max_{1 \leq j \leq m} \ell_j(Q) = \max_{1 \leq j \leq m} \|e_j^T Q\|_2^2.$$

Although coherence contains far less information about a matrix than the leverage scores, it can still be useful in bounding the condition number of a sampled matrix (see Bound 1) and may be easier to estimate than leverage scores Due to the properties of leverage scores, the inequality $n/m \leq \mu(A) \leq 1$ holds, and if $\mu(A) \approx n/m$, then $\ell_i(A) \approx n/m$. Our code for computing coherence is `coherence.m`.

4.2 kappa_SQ Design

Kappa_SQ was designed to perform all of the computations from Chapter 2 and output paper-ready plots. It can assist researchers in the following ways.

First, the GUI for kappa_SQ has been designed to assist the user set-up, perform and plot experiments on $\kappa(SQ)$. There are two types of experiments, the computation of (possibly probabilistic) bounds on $\kappa(SQ)$ and the computation of $\kappa(SQ)$ for a given or generated test matrix Q. The GUI has also been coded to allow a user to easily incorporate their own codes by simply placing a properly formatted `Matlab` function in the "boundsAndAlgorithms" directory.

Second, kappa_SQ includes a collection of codes for various algorithms and bounds pertaining to the field of randomized row sampling. These codes are all written as `Matlab` function files that can be used on their own or with the kappa_SQ GUI. The codes include functions for row sampling, test matrix generation, leverage score distribution generation and functions to compute bounds for $\kappa(SQ)$. The included codes are outlined in section 4.2.2.

The kappa_SQ codes can be broken up into two main groups, the GUI and "Algorithm Codes." Below, we describe these codes and their functions.

4.2.1 kappa_SQ GUI

The kappa_SQ GUI is designed to produce plots of both numerical experiments, where $\kappa(SQ)$ is actually measured, and of bounds on $\kappa(SQ)$. To perform a numerical experiment, kappa_SQ will perform sampling on a matrix and then measure the condition number of the sampled matrix, $\kappa(SQ)$. Occasionally, the sampled matrix, SQ, is not full rank. This event is termed a "failure" event and kappa_SQ also keeps track of these.

When kappa_SQ has completed its computations, it will output plots like those shown in Figures 4.1 and 4.2. For the moment, do not worry about the specifics of each plot other than the following. The triangles in figure 4.1 show the measured condition number of the sampled matrix, $\kappa(SQ)$, and the line plots a bound on $\kappa(SQ)$. For the sections of the domain where a line is not plotted, the bound does not apply. All of the bounds included with $\kappa(SQ)$ are probabilistic bounds and therefore only hold with probability at least $1 - \delta$. Therefore, at least $100(1 - \delta)\%$ of the measured $\kappa(SQ)$ should be below the line. In Figure 4.2, the "failure rate," the percent of numerical experiments that resulted in a failure event, is plotted. Despite the fact that for many experiments the failure rate will be 0% for most values, it is still an important quantity because figure 4.1 only plots the the "good" events (where SQ has full column rank).

When a user executes `kappaSQ.m`, he or she is presented with the kappa_SQ GUI (see Figure 4.3). It is broken up into sections; the first two correspond to inputs the user must provide to produce a plot and the last two are for plot editing and and to display important information. Below we describe these sections and, in the process, how to use the GUI to produce plots.

"Step 1: Select Bounds and/or Numerical Experiments." This section allows the user to select what he or she would like to plot. The first listbox contains possible bounds that the user can plot. The second listbox contains various sampling methods. In kappa_SQ, numerical experiments are first defined in terms of what sampling algorithm the user would like to use. By selecting a sampling method, the user is telling kappa_SQ that he or she wishes to do a numerical experiment with that sampling method. Selecting multiple sampling methods will perform multiple experiments.

"Step 2: Matrix Properties / Parameters." This section allows the user to provide the required inputs. Only inputs that are required will be visible. As an example, if the user chose to plot Bound 1, then this section would ask for the user to provide values for m, n, μ, δ and c as those values are required to compute Bound 1. In addition, either c or μ must be a

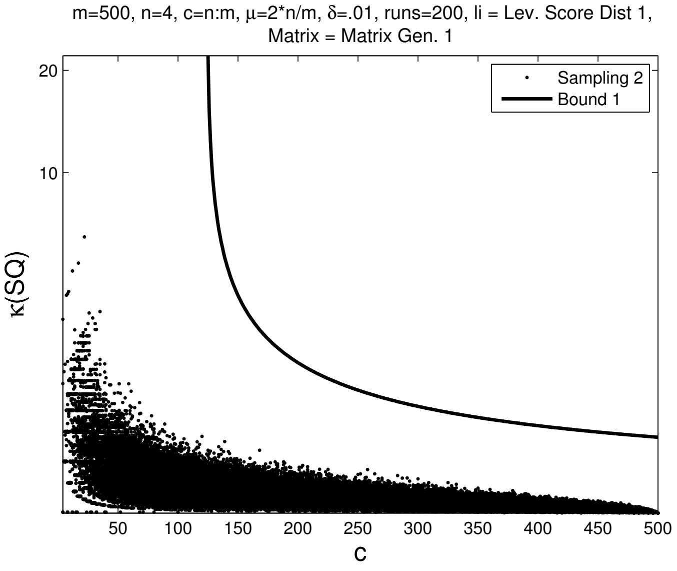


Figure 4.1: In this plot we show the results of a numerical experiment (triangles) and a bound on kappa_SQ (line) that holds with probability $1 - \delta$.

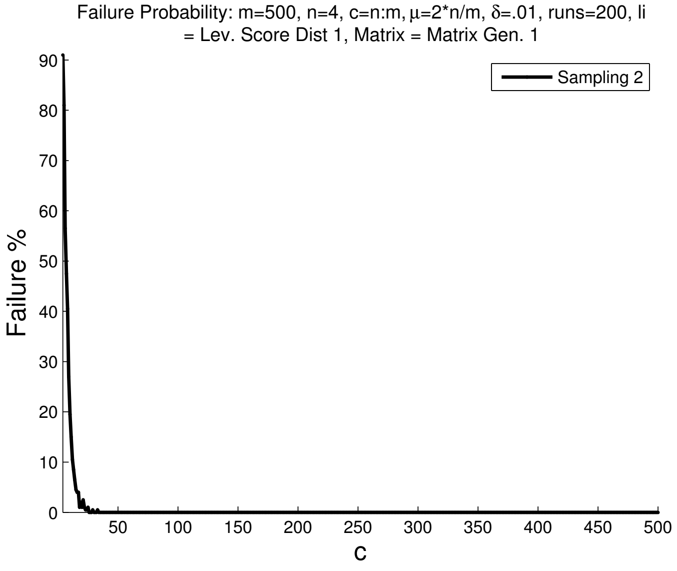


Figure 4.2: In this plot we show the failure rate of a numerical experiment on kappa_SQ.

vector and whichever is a vector will be placed on the x-axis.

Of particular interest in this section are the "Matrix Generation" and "li" (ℓ_i) inputs. When a test matrix is required (ex: when running a numerical experiment), kappa_SQ will generate a matrix using the algorithm specified in this listbox. Similarly, when a leverage score distribution is required, kappa_SQ will generate one by the method specified in the "li" listbox.

"Plot Button." Once the user has completed steps 1 and 2, he or she may click the plot button to run and plot the experiment.

"Help!" This button will open the kappa_SQ help file. This file includes a FAQ section and a list of included functions.

Adv. Features: "Batch Features." This section can be viewed by clicking on the "Adv. Features" button. After performing steps 1 and 2, the user can instead add the current experiment to a batch of jobs to be run later in serial. This is particularly useful if the chosen experiments require a long time to run, or if the user has many experiments to run. In addition, if the user keeps all of their experiments defined in a batch file, they can easily repeat all of the experiments for their work.

Adv. Features: "Other Features." This section contains a button called "Beautify Plots" which will open the plot editing window shown in Figure 4.4. This window assists the user with modifying many of the common plot settings and creating a script that will apply these settings to future plots. In addition, the plot editing window will generate a command which will apply these settings without the GUI. KappaSQ can be set to run this command for all future plots by checking the "beautify command" checcckbox and entering the command. Thus, the user only needs to set up their plots once. Finally, this section also has an option to plot a standard confidence interval for the failure probability.

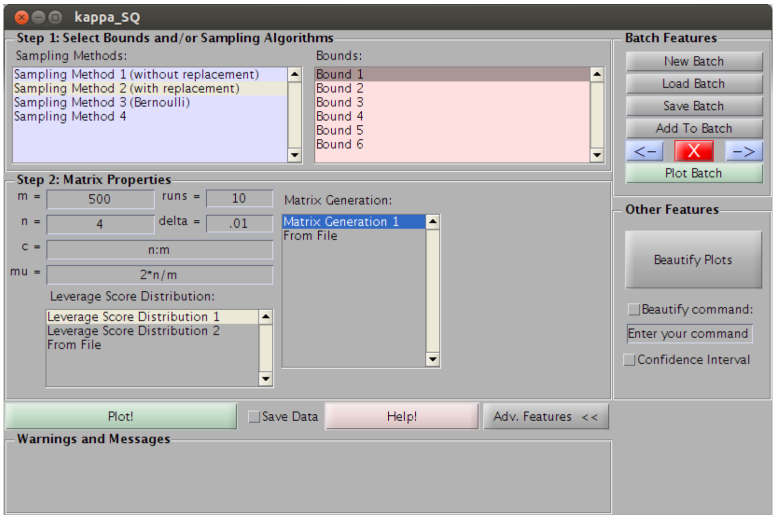
Below we will discuss the various included codes that the kappa_SQ GUI uses in the above sections to produce plots.

4.2.2 Algorithm Codes

In this section, we describe the various algorithms and bounds that are included in the kappa_SQ package. These algorithms can be broken up into four main groups, bounds, matrix generation, sampling methods and leverage score distributions.

Sampling methods

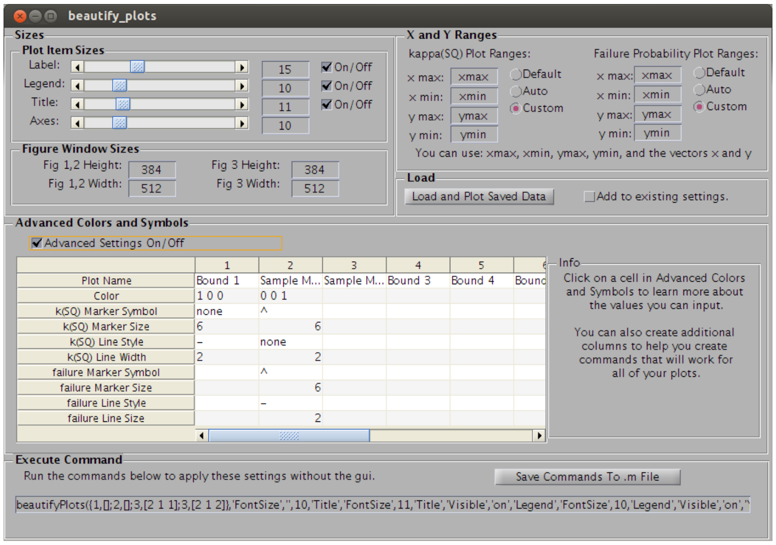
We include four different row sampling methods from the literature, Sampling without Replacement, Sampling with Replacement, Bernoulli Sampling, and Sampling Proportional to Leverage Scores. In Chapter 2, the first three of these sampling methods are described by constructing a sampling matrix S such that SQ is the sampled matrix. In the kappa_SQ package, we instead



m = 500
runs = 10
n = 4
delta = .01
c = n:m
mu = 2*n/m

□ Beautify command: Enter your command
□ Confidence Interval
□ Save Data

Figure 4.3: kappaSQ GUI with advanced features shown.



Label: 15 ☑ On/Off
Legend: 10 ☑ On/Off
Title: 11 ☑ On/Off
Axes: 10

Fig 1,2 Height: 384
Fig 1,2 Width: 512
Fig 3 Height: 384
Fig 3 Width: 512

kappa(SQ) Plot Ranges: x max: xmax; x min: xmin; y max: ymax; y min: ymin ○ Default ○ Auto ● Custom

Failure Probability Plot Ranges: x max: xmax; x min: xmin; y max: ymax; y min: ymin ○ Default ○ Auto ● Custom

You can use: xmax, xmin, ymax, ymin, and the vectors x and y

□ Add to existing settings.

☑ Advanced Settings On/Off

	1	2	3	4	5	6
Plot Name	Bound 1	Sample M...	Sample M...	Bound 3	Bound 4	Bound
Color	1 0 0	0 0 1				
k(SQ) Marker Symbol	none	^				
k(SQ) Marker Size	6	6				
k(SQ) Line Style	–	none				
k(SQ) Line Width	2	2				
failure Marker Symbol		^				
failure Marker Size		6				
failure Line Style		–				
failure Line Size		2				

Click on a cell in Advanced Colors and Symbols to learn more about the values you can input.

You can also create additional columns to help you create commands that will work for all of your plots.

Run the commands below to apply these settings without the gui.

```
beautifyPlots({1,[];2,[];3,[2 1 1];3,[2 1 2]},'FontSize','',10,'Title','FontSize',11,'Title','Visible','on','Legend','FontSize',10,'Legend','Visible','on','
```

Figure 4.4: Beautify Plots GUI.

code these algorithms to compute $B =\equiv SQ$ directly. Each sampling method inputs the initial matrix, Q, and the desired (or desired expected) number of rows to be sampled, c, and outputs the sampled matrix, SQ.

Sampling Method 4.2.1, Sampling Without Replacement. This algorithm samples exactly the desired number of rows such that no row is sampled more than once by sampling uniformly from the $m!/(m-c)!$ possible permutations of c rows. We implement this by first randomly permuting all of the rows and then sampling the first c rows. In the algorithm below we use the term random permutations. A permutation $\pi_1, \ldots, \pi_m$ of the integers $1, \ldots, m$ is a *random permutation*, if it is equally likely to be one of $m!$ possible permutations [29, pages 41 and 48]. The matlab command `randperm(m)` will generate a random permutation of the integers $1, \ldots, m$. Our code for this sampling method is `Sample_randperm.m`.

Algorithm 4.2.1 *Sampling Without Replacement*, [16, 18]

```
Input: A $m \times n$ matrix $Q$ and an integer $c$, such that $1 \leq c \leq m$.
Output: A $c \times m$ sampled matrix $B$.
$v = \texttt{randperm}(m)$  $s = v(1:c)$  $B = \sqrt{m/c}\, Q(s,:)$
```

Sampling Method 4.2.2, Sampling With Replacement (Exactly(c)). This algorithm samples exactly the desired number of rows with a uniform probability distribution and with replacement. Our code for implementing this sampling method is `Sample_exactlyC.m`.

Algorithm 4.2.2 *Sampling With Replacement, [11, 15]*

```
Input: A $m \times n$ matrix $Q$ and an integer $c$, such that $1 \leq c \leq m$.
Output: A $c \times m$ sampled matrix $B$.
Let $\pi_1, \ldots, \pi_c$ be integers uniformly sampled from$\{1, \ldots, m\}$ with replacement $s = \begin{bmatrix} \pi_1, & \ldots, & \pi_c \end{bmatrix}$ $B = \sqrt{m/c}\, Q(s,:)$
```

Sampling Method 4.2.3, Bernoulli Sampling. In this sampling method, each row is either sampled, with probability c/m, or not sampled with probability $1 - c/m$. Thus, whether or not each row is sampled is an independent Bernoulli trial and the expected total number of rows sampled is c. Our implementation of this algorithm differs slightly from Algorithm 2.3.3. In Algorithm 2.3.3, rows that are not sampled are set to 0, while in our code they are removed. Removing the zero rows is more memory efficient, avoids unnecessary matrix-matrix multiplications and does not affect $\kappa(SQ)$. Our code for this algorithm is `Sample_bernoulli.m`.

Algorithm 4.2.3 *Bernoulli Sampling, [1, 16, 18]*

```
Input: A $m \times n$ matrix $Q$ and an integer $c$, such that $1 \leq c \leq m$.
Output: A $c \times m$ sampled matrix $B$.
Let $\pi$ be a $m \times 1$ vector of $m$ independent realizations of a boolean random variable with success probability $c/m$
Let $s$ be a $\hat{c} \times 1$ vector containing the indices where $\pi_i = 1$, where $1 \leq i \leq m$ and $\hat{c} =$ the number of nonzero entries in $\pi$
$B = \sqrt{m/c}\, Q(s, :)$
```

Sampling Method 4.2.4, Sampling Proportional to Leverage Scores. In this sampling method, c rows are sampled with probability $\ell_i(Q)/n$ with replacement. Our code for implementing this sampling method is `Sample_leverageScores.m`

Algorithm 4.2.4 *Sampling Proportional to Leverage Scores*

```
Input: A $m \times n$ matrix $Q$ and an integer $c$, such that $1 \leq c \leq m$ and the leverage scores $\ell(Q)$.
Output: A $c \times m$ sampled matrix $B$.
Let $\pi_1, \ldots, \pi_c$ be integers sampled from $\{1, \ldots, m\}$ with probabilities $\{\ell_1(Q)/n, \ldots, \ell_m(Q)/n\}$ and replacement
$s = \begin{bmatrix} \pi_1, & \ldots, & \pi_c \end{bmatrix}$
$B = \sqrt{m/c}\, Q(s, :)$
```

Bounds

We include codes for the two probabilistic bounds for $\kappa(SQ)$ from Chapter 2 and four other weaker bounds that were included in the first version of our paper [23].

Bound 1, Coherence based bound. This bound is expressed in terms of coherence and comes from a matrix Chernoff concentration inequality [37, Corollary 5.2]. It applies to all of the sampling methods 4.2.2 except for Sampling Proportional to Leverage Scores (Sampling Method 4.2.4). Our code for this bound is `Bound_muBound.m`.

Bound 1 (Theorem 7) *Let Q be a real $m \times n$ matrix with $Q^TQ = I_n$ and coherence μ. Let SQ be a sampling matrix produced by Algorithms 4.2.1, 4.2.2, or 4.2.3 with $n \leq c \leq m$. For $0 < \epsilon < 1$ and $f(x) \equiv e^x(1+x)^{-(1+x)}$ define*

$$\delta \equiv n \left(f(-\epsilon)^{c/(m\mu)} + f(\epsilon)^{c/(m\mu)} \right). \tag{4.1}$$

If $\delta < 1$, then with probability at least $1-\delta$ we have rank$(SQ) = n$ and

$$\kappa(SQ) \leq \sqrt{\frac{1+\epsilon}{1-\epsilon}}.$$

Bound 2, Leverage score based bound. This bound is based on leverage scores and only applies to sampling without replacement (Sampling Method 4.2.1). It is based on a matrix Bernstein concentration inequality [32, Theorem 4]. Our code for this bound is `Bound_leverageScoresBound.m`.

Bound 2 (Theorem 11) *Let Q be a $m \times n$ real matrix with $Q^TQ = I_n$, leverage scores $\ell_j(Q)$, $1 \leq j \leq m$, and coherence μ. Let L be a diagonal matrix such that $L_{j,j} = \ell_j(Q)$. Let S be a sampling matrix produced by Algorithm 4.2.2 with $n \leq c \leq m$. For $0 < \epsilon < 1$ set*

$$\delta \equiv 2n \exp\left(-\tfrac{3}{2}\,\frac{c\epsilon^2}{m\,(3\|Q^TLQ\|_2 + \epsilon\mu)}\right).$$

If $\delta < 1$, then with probability at least $1-\delta$ we have rank$(SQ) = n$ and

$$\kappa(SQ) \leq \sqrt{\frac{1+\epsilon}{1-\epsilon}}.$$

Bound 3, Weaker coherence based bound. This bound is based on a probabilistic two-norm bound for a Monte Carlo matrix multiplication algorithm that samples according to Sampling Method 4.2.2 [15, Theorem 4]. Our code for this bound is `weakerBound_1.m`.

Bound 3 ([23, Theorem 3.2]) *Given $0 < \epsilon < 1$ and $0 < \delta < 1$, let Q be a $m \times n$ real matrix with $Q^TQ = I_n$ and coherence μ. Let c be an integer so that*

$$\min\left\{n, \zeta \ln\left(\zeta/\sqrt{\delta}\right)\right\} \leq c \leq m, \qquad \textit{where} \qquad \zeta \equiv \frac{96m\,\mu}{\epsilon^2}.$$

If S is a $c \times m$ matrix produced by Sampling Method 4.2.2 with uniform probabilities $p_k = 1/m$, $1 \leq k \leq m$, then with probability at least $1-\delta$, we have $\operatorname{rank}(SQ) = \operatorname{rank}(M_s) = n$ *and*

$$\kappa(SQ) = \kappa(AR_s^{-1}) \leq \sqrt{\frac{1+\epsilon}{1-\epsilon}}.$$

Bound 4, Weaker coherence based bound. This bound is based on a special case of the noncommutative Bernstein inequality [32, Theorem 4] and applies to sampling Sampling Method 4.2.2. Our code for this bound is `weakerBound_3.m`.

Bound 4 ([23, Corollary 3.10]) *Given $c \geq n$ and $0 < \delta < 1$, let Q be a $m \times n$ real matrix with $Q^TQ = I_n$ and coherence μ. Let $\rho \equiv \frac{2}{3}\ln(2n/\delta)$ and*

$$\epsilon_1 \equiv \frac{\mu m}{2c}\left(\rho + \sqrt{\frac{12c\rho}{m\mu} + \rho^2}\right).$$

Let S be a $m \times m$ matrix produced by Algorithm 4.2.2. If $\epsilon_1 < 1$ then with probability at least $1 - \delta$, we have $\operatorname{rank}(SQ) = n$ *and*

$$\kappa(SQ) \leq \sqrt{\frac{1+\epsilon_1}{1-\epsilon_1}}.$$

Bound 5, Weaker coherence based bound. This bound is based on a Frobenius norm bound for a Monte Carlo matrix multiplication algorithm that samples according to Sampling Method 4.2.2 and applies to sampling Sampling Method 4.2.2. Our code for this bound is `weakerBound_4.m`.

Bound 5 ([23, Theorem 3.5]) *Given $0 < \delta < 1$ and $c \geq n$, let Q be a $m \times n$ real matrix with $Q^TQ = I_n$ and coherence μ. Let*

$$\epsilon_2 \equiv \sqrt{\frac{mn\,\mu}{c}} + m\,\mu\sqrt{\frac{8\log(1/\delta)}{c}}.$$

Let S be a $c \times m$ matrix produced by Algorithm 4.2.2 with uniform probabilities $p_k = 1/m$, $1 \leq k \leq m$. If $\epsilon_2 < 1$, then with probability at least $1 - \delta$, we have $\operatorname{rank}(SQ) = n$ *and*

$$\kappa(SQ) \leq \sqrt{\frac{1+\epsilon_2}{1-\epsilon_2}}.$$

Bound 6, Weaker coherence based bound. This bound is again based on the noncommutative Bernstein inequality in [32, Theorem 4] and applies to sampling Sampling Method 4.2.3. Our code for this bound is `weakerBound_6.m`.

Bound 6 ([23, Corollary 4.3]) *Given $m \geq n$, $0 < \gamma < 1$ and $0 < \delta < 1$, let Q be a $m \times n$ real matrix with $Q^TQ = I_n$ and coherence μ. Let $\rho \equiv \frac{2}{3}\ln(2n/\delta)$ and*

$$\hat{\epsilon}_3 \equiv \frac{\mu}{2}\left(\phi\rho + \sqrt{\frac{1-\gamma}{\gamma}\,12m\rho + \phi^2\rho^2}\right), \qquad \phi = \begin{cases} 1 & \textit{if } \gamma \geq 1-\gamma \\ \frac{1-\gamma}{\gamma} & \textit{if } 1-\gamma > \gamma \end{cases}$$

Let S be a $m \times m$ matrix produced by Algorithm 4.2.3. If $\hat{\epsilon}_3 < 1$ then with probability at least

$1 - \delta$, *we have* $\operatorname{rank}(SQ) = n$ *and*

$$\kappa(SQ) \le \sqrt{\frac{1+\hat{\epsilon}_3}{1-\hat{\epsilon}_3}}.$$

Leverage score distribution

We include code for two functions which define leverage score distributions.

Leverage Score Distribution 4.2.5, Good leverage score distribution. The first function is designed to be an ideal case for row sampling. It outputs a leverage score distribution with one leverage score is set equal to the coherence and the remaining leverage scores all identical. Thus, most rows are equally "important" and uniform row sampling should work well. The code for this algorithm is `liDist_oneBig.m`.

Algorithm 4.2.5 *Good leverage score distribution Algorithm 2.6.2*

```
Input: Integers $m$ and $n$ such that $m \ge n \ge 1$, and desired coherence $\mu$.
Output: A $m \times 1$ vector, $\ell$ of leverage scores such that $\max \ell = \mu$.
$\ell = [\mu; \text{ones}(m-1,1)(n-mu)/(m-1)]$
```

Leverage Score Distribution 4.2.6, Bad leverage score distribution. The second function is designed to be a particularly bad case for row sampling. It outputs a leverage score distribution with the maximal number of entries set equal to the coherence and at most one additional non-zero entry. Matrices with this leverage score distribution will have the maximal number of zero rows for the given coherence. Zero rows are bad for uniform row sampling since zero rows contain no information. The code for this algorithm is `liDist_manyBig.m`.

Algorithm 4.2.6 *Bad leverage score distribution Algorithm 2.6.3*

```
Input: Integers $m$ and $n$ such that $m \ge n \ge 1$, and desired coherence $\mu$.
Output: A $m \times 1$ vector, $\ell$ of leverage scores such that $\max \ell = \mu$.
$\tilde{m} = \lfloor n/\mu \rfloor$  if $\tilde{m} < m$ then
    $\ell = [\mu\text{ones}(\tilde{m},1); n - \tilde{m}\mu, \text{zeros}(m-\tilde{m},1)]$
else
    $\ell = \mu\text{ones}(\tilde{m},1)$
end
```

Test matrix generation

Included in kappa_SQ, we provide code for a deterministic matrix generation algorithm.

Matrix Generation 4.2.7, Deterministic matrix generation algorithm. This algorithm inputs the desired matrix dimensions and leverage scores and outputs a test matrix, Q, with these properties. To compute Q, the algorithm applies $m-1$ Givens rotations to the matrix $Q = \begin{bmatrix} I_n & \text{zeros}(n, m-n) \end{bmatrix}^T$. Each Givens rotation alters the leverage scores of two rows such that at least one of the two rows has the desired leverage score. The reason that only $m-1$ Givens rotations are required is that the leverage scores sum to n and thus the final leverage sore is determined by the other $m-1$ leverage scores. We note here that this algorithm is a transposed version of [9, Algorithm 3] and that the Givens rotations are computed from numerically stable expressions [9, section 3.1]. The code for this algorithm is `mtxGen_li.m`.

Algorithm 4.2.7 *Matrix Generation 4.2.7*

Input: Integers m and n such that $m \geq n \geq 1$, and a $m \times 1$ vector l of the desired leverage scores.

Output: A $m \times n$ matrix Q with orthonormal columns and the desired leverage scores.

```
$Q = \begin{bmatrix} I_n & \text{zeros}(n, m-n) \end{bmatrix}^T$  $[l, I]$=sort($l$);% Sort and store original order  $i = m - n$  $j = m - n + 1$  for $dummyVar = 1 : m - 1$ do
    if $\left| l_i - \|e_i^T Q\|_2 \right| < \left| l_j - \|e_j^T Q\|_2 \right|$ then
        Rotate rows $i$ and $j$ of $Q$ so that $\|e_i^T Q\|_2^2 = l_i$  $i = i - 1$
    else
        Rotate rows $i$ and $j$ of $Q$ so that $\|e_j^T Q\|_2^2 = l_j$  $j = j + 1$
    end
end
$Q(I, :) = Q$;% Undo sorting
```

4.2.3 Other Functions

We also include two simple functions to assist with choosing nice, aesthetically pleasing, ranges for c and μ named `logPoints.m` and `logPointsDouble.m`, respectively. These functions produce ranges that are more heavily weighted towards the smaller end of the desired range. Most of the interesting action in the final plots occurs near smaller c or μ values and, in addition, larger values of c are more computationally expensive. We describe these functions in the kappa_SQ help file which can be accessed by pressing the help button in the gui (see Section 4.2.1).

4.3 Examples

Here we show a few examples of some of the ways that the kappa_SQ GUI can be used.

4.3.1 Example 1

In this example, we show how to perform a basic experiment with the GUI. In this experiment we compare Bound 1 to a numerical experiment with sampling Sampling Method 4.2.2. Since Bound 1 applies to this sampling method, the results should show that at least $100(1-\delta)\%$ of the measured $\kappa(SQ)$ are less than the bound. In order for kappa_SQ to run a numerical experiment, it must have a test matrix to work on. Here, we will chose to generate a test matrix with Sampling Method 4.2.7 and leverage scores defined by Sampling Method 4.2.5.

To set up the experiment, start by selecting the desired bound and sampling method. Then, move on to step 2 and input the following values, `m=500`, `n=4` `c=n:m`, `mu=2n/m`, `runs=10`, and `delta=.01`. For the "Matrix Generation" listbox, select "Matrix Generation 4.2.7" (Sampling Method 4.2.7). This will cause the leverage score listbox to appear since this matrix generation algorithm requires a leverage score distribution. Select "Leverage Score Distribution 1" (Sampling Method 4.2.5) for the leverage score distribution. In figure 4.3 we show how the GUI should at this point. When ready, click the plot button to begin the experiment. We show the resulting plot in figure 4.5.

4.3.2 Example 2

Here we show how to set up a kappa_SQ batch to perform multiple experiments in serial. To create a new batch, first click the "Adv. Features" button to expand the GUI and then click on the "New Batch" button to start a new batch. Next, set up an experiment by performing steps 1 and 2 as described in the first example. Then, instead of clicking the plot button, click the "Add to Batch" button. This will add the current experiment to the batch. Repeat this process for the remaining experiments. The user may use the arrow buttons to navigate and the "X" button to delete previously entered experiments. When ready, click the "Save Batch" button to save the current experiments to a file and then the "Run Batch" button to have kappa_SQ run all of the experiments. Plot images will be saved automatically with a file name based on the batch file name and their job number.

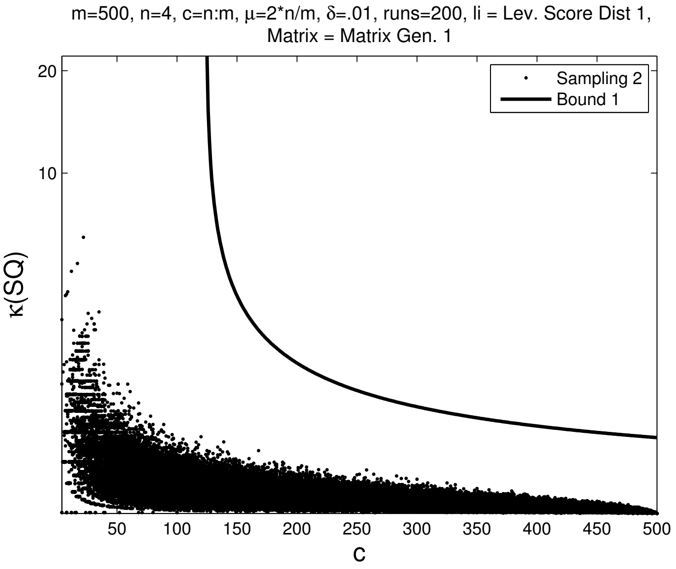


Figure 4.5: The solid line shows Bound 1 and the triangles show the results of the numerical experiments with sampling Sampling Method 4.2.2 and a matrix generated by Sampling Method 4.2.7.
Here Q is a matrix generated by Algorithm 4.2.7 with orthonormal columns, $m = 10,000$, $n = 4$, coherence $\mu = 20n/m$, . Left panel: Horizontal coordinate axes represent amounts of sampling $n \leq c \leq 10,000$. Vertical coordinate axes represent condition numbers $\kappa(SQ)$; the maximum is 10. Right panels: Horizontal coordinate axes represent amounts of sampling that give rise to numerically rank deficient matrices SQ. Vertical coordinate axes represent percentage of numerically rank deficient matrices.

4.3.3 Example 3

Here we show how the built in plot editing tools can be used to expedite plot editing, how to create a script that will apply these settings to future plots and set up kappa_SQ to run that script after every experiment. To start, run an experiment as described in example 1. Then, click the "Adv. Features" button to expand the GUI and then click on the "Beautify Plots" button. This will open the plot editing window. (See figure 4.4).

This window allows the user to easily edit many different plot settings. While using this window, any changes will instantly be applied to the plots, so we suggest positioning the plots and the GUI window such that they can all be seen. When done editing, press the button labeled "Save Commands To .m File" to create a `.m` file that will apply these settings to future plots. To have kappa_SQ apply these plot settings automatically to all new plots, write the command for this file in the box labeled "Enter your command here" in the main GUI window and check the "Beautify Command" checkbox. (See figure 4.3).

4.4 Conclusions

The kappa_SQ package is designed to assist researchers examine the behavior of $\kappa(SQ)$. The package includes codes for generating matrices with specific leverage score distributions, generating two specific leverage score distributions, four types of row sampling algorithms and computing bounds on $\kappa(SQ)$. These codes can be used on their own, or with the kappa_SQ GUI which is capable of setting up and running numerical experiments, computing bounds, and producing quality plots with the help of custom plot-editing tools. In addition, the GUI has been designed to detect properly formatted `Matlab` function files, which allows the user to incorporate their own codes into the GUI.

Chapter 5

Future Work

5.1 Future work

The following is a list of what we would like to accomplish next.

- Tightening of Corollary 8 so that it retains the strength of the Chernoff concentration inequality inherent in Theorem 11.
- Extension of the condition number bounds in Section 2.5 to uniform sampling without replacement (Algorithm 2.3.1) and Bernoulli sampling (Algorithm 2.3.3).
- Determination of a statistically significant number of runs for each sampling amount c, for two purposes:
    1. To assert, within a specific confidence interval, bounds on the condition numbers of the *actually sampled* matrices.
    2. To assert with a specific confidence that the probabilistic expressions in Sections 2.4 and 2.5 do indeed represent bounds.
- For uniform sampling, determine when one should use a randomized preprocessing transformation, such as the randomize Hadamard transformation [5, 1], rather than just sampling more rows.
- Derive bounds for the condition number of a sampled matrix if sampling is performed with leverage score probabilities as in Sampling Method 4.2.4.
- Derive bounds for the sensitivity of leverage scores where the perturbation is multiplicative.

REFERENCES

[1] Haim Avron, Petar Maymounkov, and Sivan Toledo. Blendenpik: supercharging Lapack's least-squares solver. *SIAM J. Sci. Comput.*, 32(3):1217–1236, 2010.

[2] Laura Balzano, Benjamin Recht, and Robert Nowak. High-dimensional matched subspace detection when data are missing. *CoRR*, abs/1002.0852, 2010.

[3] Rajendra Bhatia. *Matrix analysis*, volume 169 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1997.

[4] Christos Boutsidis and Petros Drineas. Random projections for the nonnegative least-squares problem. *Linear Algebra Appl.*, 431(5-7):760–771, 2009.

[5] Christos Boutsidis and Alex Gittens. Improved matrix algorithms via the subsampled randomized Hadamard transform. *SIAM J. Matrix Anal. Appl.*, 34(3):1301–1340, 2013.

[6] Christos Boutsidis, Michael W. Mahoney, and Petros Drineas. An improved approximation algorithm for the column subset selection problem. *CoRR*, abs/0812.4293, 2010.

[7] Emmanuel J. Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Found. Comput. Math.*, 9(6):717–772, 2009.

[8] Samprit Chatterjee and Ali S. Hadi. Influential observations, high leverage points, and outliers in linear regression. *Statist. Sci.*, 1(3):379–416, 1986. With discussion.

[9] Inderjit S. Dhillon, Robert W. Heath, Jr., Mátyás A. Sustik, and Joel A. Tropp. Generalized finite algorithms for constructing Hermitian matrices with prescribed diagonal and spectrum. *SIAM J. Matrix Anal. Appl.*, 27(1):61–71 (electronic), 2005.

[10] David L. Donoho and Xiaoming Huo. Uncertainty principles and ideal atomic decomposition. *IEEE Trans. Inform. Theory*, 47(7):2845–2862, 2001.

[11] Petros Drineas, Ravi Kannan, and Michael W. Mahoney. Fast Monte Carlo algorithms for matrices. I. Approximating matrix multiplication. *SIAM J. Comput.*, 36(1):132–157, 2006.

[12] Petros Drineas, Michael W. Mahoney, and S. Muthukrishnan. Sampling algorithms for l_2 regression and applications. In *Proc. 17th Ann. ACM-SIAM Symp. on Discrete Algorithms*, SODA '06, pages 1127–1136, New York, NY, USA, 2006. ACM.

[13] Petros Drineas, Michael W. Mahoney, and S. Muthukrishnan. Subspace sampling and relative-error matrix approximation: column-based methods. In *Approximation, randomization and combinatorial optimization*, volume 4110 of *Lecture Notes in Comput. Sci.*, pages 316–326. Springer, Berlin, 2006.

[14] Petros Drineas, Michael W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM J. Matrix Anal. Appl.*, 30(2):844–881, 2008.

[15] Petros Drineas, Michael W. Mahoney, S. Muthukrishnan, and Tamás Sarlós. Faster least squares approximation. *Numer. Math.*, 117(2):219–249, 2011.

[16] Alex Gittens and Joel A. Tropp. Tail bounds for all eigenvalues of a sum of random matrices, 2011.

[17] Gene H. Golub and Charles F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

[18] David Gross and Vincent Nesme. Note on sampling without replacing from a finite collection of matrices, 2010.

[19] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.*, 53(2):217–288, 2011.

[20] D. C. Hoaglin and R. E. Welsch. The Hat matrix in regression and ANOVA. *Amer. Statist.*, 32(1):17–22, 1978.

[21] Roger A. Horn and Charles R. Johnson. *Matrix analysis*. Cambridge University Press, Cambridge, second edition, 2013.

[22] I. C. F. Ipsen and T. Wentworth. kappaSQ_v3, 2013. http://www4.ncsu.edu/~ipsen/papers.html.

[23] Ilse C. F. Ipsen and Thomas Wentworth. The effect of coherence on sampling from matrices with orthonormal columns, and preconditioned least squares problems (v1). *CoRR*, abs/1203.4809v1, 2012.

[24] D. V. Lindley. The Bayesian approach. *Scand. J. Statist.*, 5(1):1–26, 1978. With discussion.

[25] Michael W. Mahoney. Randomized algorithms for matrices and data. *CoRR*, abs/1104.5557, 2011.

[26] Michael W. Mahoney and Petros Drineas. CUR matrix decompositions for improved data analysis. *Proc. Natl. Acad. Sci. USA*, 106(3):697–702, 2009. With supplementary material available online.

[27] Xiangrui Meng, Michael A. Saunders, and Michael W. Mahoney. LSRN: A Parallel Iterative Solver for Strongly Over- or Underdetermined Systems. *SIAM J. Sci. Comput.*, 36(2):C95–C118, 2014.

[28] Carl Meyer. *Matrix analysis and applied linear algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000. With 1 CD-ROM (Windows, Macintosh and UNIX) and a solutions manual (iv+171 pp.).

[29] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, New York, NY, USA, 2005.

[30] Christopher C. Paige and Michael A. Saunders. LSQR: an algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Software*, 8(1):43–71, 1982.

[31] Dimitris Papailiopoulos, Anastasios Kyrillidis, and Christos Boutsidis. Provable deterministic leverage score sampling. *CoRR*, abs/1404.1530, 2014.

[32] Benjamin Recht. A simpler approach to matrix completion. *J. Mach. Learn. Res.*, 12:3413–3430, 2011.

[33] Vladimir Rokhlin and Mark Tygert. A fast randomized algorithm for overdetermined linear least-squares regression. *Proc. Natl. Acad. Sci. USA*, 105(36):13212–13217, 2008.

[34] Sheldon M. Ross. *Introduction to Probability Models, Ninth Edition.* Academic Press, Inc., Orlando, FL, USA, 2006.

[35] Ameet Talwalkar and Afshin Rostamizadeh. Matrix coherence and the nystrom method. *CoRR*, abs/1004.2008, 2010.

[36] Joel A. Tropp. Improved analysis of the subsampled randomized Hadamard transform. *Adv. Adapt. Data Anal.*, 3(1-2):115–126, 2011.

[37] Joel A. Tropp. User-friendly tail bounds for sums of random matrices. *Found. Comput. Math.*, 12(4):389–434, 2012.

[38] P. F. Velleman and R. E. Welsch. Efficient computing of regression diagnostics. *Amer. Statist.*, 35(4):234–242, 1981.

[39] Shusen Wang. On the lower bounds of the nyström method. *CoRR*, abs/1303.4207, 2013.

[40] Per-Æke Wedin. Perturbation bounds in connection with singular value decomposition. *BIT Numerical Mathematics*, 12(1):99–111, 1972.

[41] Per-Æke Wedin. On angles between subspaces of a finite dimensional inner product space. In Bo Kågström and Axel Ruhe, editors, *Matrix Pencils*, volume 973 of *Lecture Notes in Mathematics*, pages 263–285. Springer Berlin Heidelberg, 1983.