Randomized Least Squares
MATH-453 Computational Linear Algebra Project

Maximo Cravero Baraja
Ecole Polytechnique Federale de Lausanne
maximo.craverobaraja@epfl.ch

Abstract

This report aims to provide an overview of the Blendenpik algorithm, which is a randomized version of a least squares solver geared towards large scale, highly overdetermined systems of the form $\mathbf{Ax} = \mathbf{b}$. Theoretical and practical results will be used to illustrate the importance of coherence and row-mixing to enable a uniform sampling of the rows to create a good preconditioner with high probability.

1 Background

The setting which is considered throughout this report is that of a highly overdetermined system $Ax = b$, with $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, and $m \gg n$. The aim is to obtain the optimal value of x, namely x^*, in the least squares sense i.e. $x^* = argmin_{x \in \mathbb{R}^n} ||b - Ax||_2$. Traditional least squares solvers such as `LAPACK` make use of a QR decomposition (1), where we can write:

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} = [Q_1 \quad Q_2] \begin{bmatrix} R \\ 0 \end{bmatrix} \quad Q \in \mathbb{R}^{m \times m}, Q_1 \in \mathbb{R}^{m \times n}, Q_2 \in \mathbb{R}^{m \times m-n}, R \in \mathbb{R}^{n \times n}$$

Q is an orthogonal matrix (or unitary in case A is complex), i.e. its rows and columns are orthonormal: $QQ^T = Q^TQ = I \implies Q^T = Q^{-1}$. R is an upper triangular matrix (in this case because $m > n$ and because we assume A is full rank). The linear system can thus be rewritten by using $Ax = b \implies Q^TAx = Q^Tb$, meaning:

$$||x - Ab||_2 = ||Q^Tb - Q^TAx||_2 = \left|\left| \begin{bmatrix} Q_1^Tb - Rx \\ Q_2^Tb \end{bmatrix} \right|\right|_2$$

This holds because $Q_1^TA = R$. Thus minimizing in the least squares sense using the QR decomposition boils down to solving $x = R^{-1}Q_1^Tb$. The total cost associated with solving the least squares problem in this manner consists of the cost of a QR decomposition of A, and the matrix inverse and multiplications to compute x. Since m is assumed to be large in these settings, especially given the increasing prevalence of big data applications, this can be extremely costly.

The objective of the Blendenpik algorithm is to apply uniform random sampling and random projections to define a preconditioner which reduces the number of iterations and thus computational cost associated with solving large scale systems using methods such as MINRES or LSQR. This requires careful consideration of the implications of random row sampling, as well as preprocessing steps to ensure a good solution can still be obtained with high probability.

A natural attempt at reducing the cost of solving an overdetermined system would be to sample a certain number of rows $\mathcal{R}$ and obtain the solution $x^*_{\mathcal{R}} = argmin_{x \in \mathbb{R}^n} ||A_{\mathcal{R}}x - b_{\mathcal{R}}||$. Under certain conditions on A and $\Omega(n \log(m) \log(n \log(m)))$ uniformly randomly sampled rows, this leads to a residual such that $||Ax_{\mathcal{R}} - b||_2 / ||Ax_{opt} - b||_2 \leq 1 + \epsilon$. This has two problems (2):

1. Standard stability analysis of linear least squares algorithms is done in terms of backward error.
2. Running time depends on ϵ^{-1}

An alternative approach is to compute a preconditioner based on the sampled matrix $A_{\mathcal{R}}$, namely R from $A_{\mathcal{R}} = QR$, and use this in conjunction with a standard iterative method such as LSQR or MINRES. However, depending on the matrix A, the resulting subsampled matrix may be ill-conditioned or rank deficient, leading to slow convergence or failure in the iterative subspace method. Such an approach works *only* if the coherence of the matrix is small, which is a measure of how well distributed the importance of rows in a matrix is. This can be enforced via a randomized row-mixing preprocessing phase. This is the direction the Blendenpik algorithm takes. Before further elaborating on Blendenpik and the underlying randomized sampling and row-mixing approaches, it is necessary to consider the key concepts which enable an effective use of the algorithm. This will then allow us to understand the ideas behind Blendenpik and derive bounds on the condition number of the preconditioned overdetermined system.

2 Key Concepts

2.1 Coherence

Coherence defines a notion of the degree to which the importance of the rows is concentrated onto a small number of rows. This is undesirable in the case of uniform sampling as we would need to sample a large number of rows to ensure the important rows are picked. Therefore we prefer matrices with 'mixed' information, i.e. all rows are more or less equally important. Such matrices have a low coherence value (lower bounded by $\frac{n}{m}$), whereas badly behaved matrices have high coherence values (upper bounded by 1). Given a matrix $A \in \mathbb{R}^{m \times n}$ (where $m > n$) and a matrix U with orthonormal columns that forms a basis of the column space of A, the coherence is defined as:

$$\mu(A) := \max_{i \in [m]} ||U_{i,:}||_2^2$$

Lemma 1. *The coherence $\mu(A)$ of a matrix $A \in \mathbb{R}^{m \times n}$ has the following bounds:*

$$\frac{n}{m} \leq \mu(A) \leq 1$$

Proof. First we prove the upper bound. We can define a basis with orthonormal columns by using the QR decomposition, namely $A = QR = [Q_1 Q_2]R$. Since Q is orthogonal, we know that the row norms are all equal to 1. Further, we only require the first n columns from Q to span the column space of A, which corresponds to the Q_1 matrix. Consequently we see that the row norms are now *at most* 1, i.e. $\mu(A) \leq 1$. Now to prove the lower bound of the coherence, we consider the Frobenius norm of Q_1. The squared Frobenius norm of Q_1 is $||Q_1||_F^2 = \sum_{i \in [m], j \in [n]} Q_{1_{i,j}}^2 = n$, which follows from the fact that the vector 2-norm of each column is equal to 1. If $\mu(A) < \frac{n}{m}$, that is, the maximum squared row norm is less than $\frac{n}{m}$, then that implies that $||Q_1||_F^2 < \sum_{i \in [m]} \frac{n}{m} = n$, which is a contradiction as $||Q_1||_F^2 = n$. Thus we see that $\mu(A) \geq \frac{n}{m}$ and conclude that $\frac{n}{m} \leq \mu(A) \leq 1$. □

Generally speaking the coherence number of randomly generated matrices is quite low. In a basic experiment that was performed using $A_i \in \mathbb{R}^{1000}$, $i \in [1000]$, an average coherence of 0.072 was obtained, which is quite good seeing as the lower bound on such a matrix is $\frac{50}{1000} = 0.05$. The following code illustrates the procedure to compute this mean coherence.

```
rand('state', 42);
n_matrices = 1000; n_rows = 1000; n_cols = 50;

% b) average coherences
coherences = [];
for n=1:n_matrices
    A = rand(n_rows, n_cols);
    if ~(rank(A) == n_cols)
```

```
        fprintf("rank deficient random matrix with rank: %i\n", rank(A));
    end
    [Q, R] = qr(A, 0);
    coherence = max(sum(Q .^ 2, 2));
    coherences = [coherences, coherence];
end

mean_coherence = mean(coherences);
fprintf("Mean coherence of random matrices: %d\n", mean_coherence);
```

In order to obtain the worst possible coherence, one can simply define a rank deficient matrix. For instance, in the case of the random matrices $A_i \in \mathbb{R}^{1000\times 50}$, taking any one of these matrices and setting the first column to $[0, ..., 0]^T$ will yield a QR decomposition for which the first column of Q will be $[1, 0, ..., 0]^T$, meaning the coherence is automatically equal to 1 i.e. the worst possible value.

2.2 Randomized Row-Mixing

A randomized row-mixing preprocessing phase is necessary to ensure an incoherent matrix. This enables us to apply uniform sampling to the mixed matrix and more cheaply obtain a good preconditioner R from the QR decomposition. As shown above, random matrices are typically incoherent, but this is not always the case. Given such a row-mixing transform $\mathcal{F}$, we can guarantee with high probability that the coherence is within $\mathcal{O}(\log m)$ of the optimal coherence value after a single row-mixing step. Options for the unitary seed transforms include the DCT, WHT, or DHT. In this report only the DCT is considered.

What is critical to note about these unitary row-mixing procedures is that they reduce the coherence but *do not* affect the condition number, that is, a preconditioner R derived from the QR decomposition $QR = \mathcal{F}A$ is just as good for the original matrix A as the singular values of AR^{-1} and $\mathcal{F}AR^{-1}$ are the same. When matrices are coherent, we can apply row-mixing procedures repeatedly to further improve the coherence value. One final consideration is that any fixed unitary transform $\mathcal{F}$ will lead to high coherence numbers for certain matrices. To remedy this, the process is randomized by multiplying a fixed unitary seed transform F by a random diagonal matrix D with ± 1 diagonal entries.

3 Path to Blendenpik

Note that, given a preconditioner R, the system to solve becomes $AR^{-1}y = b$, where $y = Rx$. Thus the linear least squares problem can be stated as $y^* = argmin_{y\in\mathbb{R}^n}||AR^{-1}y - b||_2$, where $x^* = R^{-1}y^*$. In an ideal setting, we could use the QR decomposition of A directly and use R as a preconditioner, in which case we would have $AR^{-1} = Q$, meaning that the resulting condition number of the matrix $\kappa(AR^{-1}) = 1$ as the condition number of an orthogonal matrix is 1. However, as explained in the introduction, this can be extremely costly.

Consequently, we wish to find a cheaper alternative to derive a preconditioner in a similar manner using a reduced QR factorization of a smaller number of rows of the matrix A. This is where the coherence of A comes into play, which affects the efficacy of deriving a preconditioner based on a subsample of the rows of A, as will be shown theoretically shortly. A general overview of the approach and thought process is outlined below.

1. Given an overdetermined system $Ax = b$ with potentially redundant information which we want to take advantage of via some sampling procedure.
2. We want a good preconditioner e.g. R from $A = QR$, but this is too expensive. Thus we would like to generate a suitable preconditioner R more cheaply by using a subsample of the rows of A. For uniform sampling to be effective, the matrix must be incoherent.
3. To guarantee incoherence with a high probability, we apply a unitary and randomized seed transform $\mathcal{F} = FD$, where F is a unitary matrix and D is a diagonal matrix with IID Rademacher random variables. Note that we do not know (or do not want to compute) its coherence number in advance (as it is expensive).

4. Given our incoherent (low coherence), randomly row-mixed matrix $\mathcal{F}A$, we can now apply uniform random row sampling via $\mathcal{S}$ with probability $\gamma n/m$ and obtain a matrix $\tilde{A}$. Now we can compute the QR decomposition $\tilde{A} = QR$ and use R as a preconditioner for A with a standard method like MINRES or LSQR.

3.1 Blendenpik - Upper Bound Proof for $\kappa(\mathbf{AR^{-1}})$

Noting these definitions, we can formally prove bounds on a matrix A preconditioned via randomized techniques. Since the point of the Blendenpik algorithm is to efficiently design a suitable preconditioner, it is useful to define an upper bound on the condition number of the preconditioned matrix $\kappa(AR^{-1})$, which is guaranteed with high probability. This is summarized in the following theorem, which relates the sample size r, the probability of failure δ, $\mu(A)$, and $\kappa(AR^{-1})$.

Theorem 2. *Let A be an $m \times n$ full rank matrix, and let $\mathcal{S}$ be a random sampling operator that samples $r \geq n$ rows from A uniformly. Let $\tau = C\sqrt{m\mu(A)\log(r)/r}$, where C is a constant. Further assume $\delta^{-1}\tau < 1$. With probability at least $1 - \delta$, the sampled matrix $\mathcal{S}A$ is full rank, and if $\mathcal{S}A = QR$ is a reduced QR factorization of $\mathcal{S}A$, we have*

$$\kappa(AR^{-1}) \leq \sqrt{\frac{1 + \delta^{-1}\tau}{1 - \delta^{-1}\tau}}$$

To prove this, we first consider Theorem 7 from (3), which provides a theoretical bound on the solution quality of an approximate matrix multiplication technique using a randomized sampling procedure.

Theorem 3. *(3) Suppose $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{n\times p}$, and $r \leq n$. Construct C and R with Algorithm 6, using the `EXPECTED(r)` algorithm. If the sampling probabilities $\{p_i\}_{i=1}^n$ used by the algorithm are of the form (44) or (45), then*

$$\mathbb{E}\left[\|AB - CR\|_F\right] \leq \frac{1}{\sqrt{\beta c}}\|A\|_F\|B\|_F$$

If, in addition, $B = A^T$, then

$$\mathbb{E}\left[\left\|AA^T - CC^T\right\|_2\right] \leq O(1)\sqrt{\frac{\log r}{\beta r}}\|A\|_F\|A\|_2$$

The `EXPECTED(r)` algorithm is a column sampling procedure. Given an input matrix $A \in \mathbb{R}^{m\times n}$ and a set of non-negative probabilities p_i such that $\sum_{i=1}^n p_i = 1$, columns in A are sampled with probability p_i and rescaled by a factor $1/min\{1, \sqrt{rp_i}\}$, leading to an output matrix $C = A\tilde{S}D$, where $\tilde{S}$ is the sampling matrix and D the rescaling matrix. We further note that the sampling probabilities p_i in 3 satisfy the following inequality:

$$p_i \geq \beta\frac{||A^{(i)}||_2^2}{||A||_F^2} \iff \beta \leq p_i\frac{||A||_F^2}{||A^{(i)}||_2^2} \tag{1}$$

We proceed by setting $A = U^T$, where $U \in \mathbb{R}^{m\times n}$ is a basis (with orthonormal columns) for the column space of A. Setting $p_i = 1/m$ and noting that the entries of the scaling matrix D are $1/\sqrt{rp_i} = \sqrt{m/r}$, we have that $C = \sqrt{m/r}U_S^T$ in 3, where U_S denotes the subsampled U matrix. By fixing $\beta = p_i\frac{||U^T||_F^2}{\mu(A)} = \frac{||U^T||_F^2}{m\mu(A)}$, Equation 1 is still satisfied as the coherence $\mu(A)$ is the maximum squared row norm. Hence we obtain:

$$
\begin{aligned}
\mathbb{E}\left[\left\|I_{n\times n} - \frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}}\right\|_2\right] &\leq O(1)\sqrt{\frac{\log r}{\beta r}}\|U^T\|_F\|U^T\|_2 \\
&= O(1)\sqrt{\frac{m\mu(A)\log r}{r||U^T||_F^2}}\|U^T\|_F\|U^T\|_2 \\
&= O(1)\sqrt{\frac{m\mu(A)\log r}{r}}\|U^T\|_2 \\
&= O(1)\sqrt{\frac{m\mu(A)\log r}{r}} \\
&= \tau
\end{aligned}
\tag{2}
$$

Here we use the fact that $||U^T||_2 = ||U||_2 = \max\{||Ux||_2 : ||x||_2 = 1\} = 1$ since $(Ux)^T(Ux) = x^T U^T U x = x^T x = 1$, as the columns of U are orthonormal. We can now bound the norm in probability by applying Markov's inequality, which states that $\mathbb{P}(X \geq a) \leq \mathbb{E}[X]/a$, where X is a non-negative random variable and $a > 0$ a non-negative constant. In particular, we consider the random variable $||I_{n\times n} - \frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}}||_2$, set $a = \delta^{-1}\tau$, and obtain the following:

$$
\mathbb{P}(||I_{n\times n} - \frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}}||_2 \geq \delta^{-1}\tau) \leq \frac{\mathbb{E}[||I_{n\times n} - \frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}}||]}{\delta^{-1}\tau} \leq \delta \tag{3}
$$

If $\mathbb{P}(X \geq a) \leq \mathbb{E}[X]/a$, this implies that $\mathbb{P}(X < a) = 1 - \mathbb{P}(X \geq a) \geq 1 - \mathbb{E}[X]/a$. Consequently we have that:

$$
\mathbb{P}(||I_{n\times n} - \frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}}||_2 < \underbrace{\delta^{-1}\tau}_{<1}) \geq 1 - \delta \tag{4}
$$

Thus we have two guarantees with probability $1 - \delta$, namely that the bound in Equation 4 holds and that the sampled matrix $A_{\mathcal{S}}$ is full rank. Given this bound, we can use a Rayleigh quotient argument to derive a bound on $\kappa(U_{\mathcal{S}})$. To relate this to $\kappa(AR^{-1})$, we consider Theorem 1 from (4).

Theorem 4. *(4) Suppose that l, m, and n are positive integers such that $m \geq l \geq n$. Suppose further that A is a full-rank $m \times n$ matrix, and that the SVD of A is*

$$A_{m\times n} = U_{m\times n}\Sigma_{n\times n}V_{n\times n*}$$

Suppose in addition that T is an $l \times m$ matrix such that the $l \times n$ matrix TU has full rank. Then, there exist an $n \times n$ matrix P, and an $l \times n$ matrix Q whose columns are orthonormal, such that

$$T_{l\times m}A_{m\times n} = Q_{l\times n}P_{n\times n}.$$

Furthermore, if P is any $n \times n$ matrix, and Q is any $l \times n$ matrix whose columns are orthonormal, such that P and Q satisfy the latter equation, then the condition numbers of AP^{-1} and TU are equal.

This tells us that $\kappa(AR^{-1}) = \kappa(U_{\mathcal{S}})$. Thus using the bound derived from the Markov inequality above, we can derive a bound on the condition number of the preconditioned matrix AR^{-1}. We are dealing with symmetric matrices, so we can use the fact that $\lambda_{max}(A) = max_{x\in\mathbb{R}^n:||x||=1}x^T Ax$, which holds analogously for the minimum eigenvalue. We want to use this to derive a bound on $\kappa(U_{\mathcal{S}})$. We proceed by using the Rayleigh quotient, namely $R(A, x) = \frac{x^T Ax}{x^T x}$, or equivalently $R(A, x) = x^T Ax$ subject to $||x||_2 = 1$ applied to $\frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}}$. We obtain the following:

$$
\begin{aligned}
R(\frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}}, x) &= \frac{x^T(\frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}})x}{x^T x} \\
&= \frac{x^T x + x^T(\frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}} - I_{n\times n})x}{x^T x} \\
&= 1 - R(\underbrace{I_{n\times n} - \frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}}}_{M}, x)
\end{aligned}
\tag{5}
$$

We note that M is symmetric, meaning that the Rayleigh quotient $R(M, x) \in [\lambda_{min}, \lambda_{max}]$, and since $||M||_2 < \delta^{-1}\tau$ (with probability $1-\delta$), it holds that $|R(M, x)| < \delta^{-1}\tau$, which implies that the eigenvalues of $\frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}}$ are contained in the interval $[1-\delta^{-1}\tau, 1+\delta^{-1}\tau]$. This allows us to bound the condition number of $U_{\mathcal{S}}$ and thus the preconditioned matrix AR^{-1}. In particular we have that:

$$\kappa(U_{\mathcal{S}}) = \frac{\sigma_{max}(U_{\mathcal{S}})}{\sigma_{min}(U_{\mathcal{S}})} = \sqrt{\frac{\lambda_{max}(U_{\mathcal{S}}^T U_{\mathcal{S}})}{\lambda_{min}(U_{\mathcal{S}}^T U_{\mathcal{S}})}} = \sqrt{\frac{\lambda_{max}(\frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}})}{\lambda_{min}(\frac{m}{r}U_{\mathcal{S}}^T U_{\mathcal{S}})}} \tag{6}$$

Note that the last inequality holds as scaling a matrix A by some constant c leads to the same eigenvalues but scaled by c i.e. $\lambda_i(cA) = c\lambda_i(A)$, which does not influence the ratio of the maximum and minimum eigenvalues. In addition we have used the fact that $\sigma_i^2(U_{\mathcal{S}}) = \lambda_i(U_{\mathcal{S}}^T U_{\mathcal{S}})$, which we have just shown to be contained in the interval $[1-\delta^{-1}\tau, 1+\delta^{-1}\tau]$ (when including the m/r factor). Finally, we can conclude the bound on the condition number of the preconditioned matrix.

$$\kappa(AR^{-1}) = \kappa(U_{\mathcal{S}}) \leq \sqrt{\frac{1+\delta^{-1}\tau}{1-\delta^{-1}\tau}}$$

One key takeaway from this proof is the fact that the condition number $\kappa(AR^{-1})$ is independent of $\kappa(A)$. It is dependent on the number of sampled rows and the coherence $\mu(A)$.

3.2 Algorithm

We apply the Blendenpik algorithm using two different iterative solvers, namely MINRES and LSQR. In the case of MINRES we need to use the normal equations $\tilde{A}^T\tilde{A}y = \tilde{A}^T b$ as a symmetric, square matrix is required, where $\tilde{A} = AR^{-1}$. For LSQR you can directly apply right preconditioning to the original system, but we consider the normal equations in the experiments conducted in section 4.

Algorithm 1 Blendenpik Algorithm (DCT)

$x =$ **blendenpik**$(A \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^n, \gamma)$
$\tilde{m} \leftarrow \lceil m/1000 \rceil \times 1000$
$M \leftarrow \begin{bmatrix} A \\ 0 \end{bmatrix} \in \mathbb{R}^{\tilde{m}\times n}$
while true

1. $M \leftarrow DCT(DM)$, where $D \in \mathbb{R}^{\tilde{m}\times\tilde{m}}$ is diagonal with $D_{ii} = \pm 1$ with equal probability
2. Define diagonal matrix $\mathcal{S} \in \mathbb{R}^{\tilde{m}\times\tilde{m}}$ with $S_{ii} = \begin{cases} 1 & \text{with probability } \gamma n/\tilde{m} \\ 0 & \text{with probability } 1-\gamma n/\tilde{m} \end{cases}$
3. Compute reduced QR decomposition $\mathcal{S}M = QR$ with $R \in \mathbb{R}^{n\times n}$
4. **if** $\kappa_{est}(R)^{-1} > 5\epsilon_{machine}$
   (a) $x \leftarrow$ MINRES/LSQR applied to linear system with preconditioner R
   (b) **return**
5. **endif**

end while

Note in addition that if the algorithm fails to produce a preconditioner R with a sufficiently small condition number for three iterations, the solution is simply returned using a standard LAPACK routine.

4 Experiments

In this section we conduct some experiments to analyze the convergence of the Blendenpik algorithm given different types of input matrices. We define an incoherent, ill-conditioned matrix A_1 with:

```
U = orth(rand(20000,400)); S = diag(linspace(1,1e5,400));
V = orth(rand(400)); A1 = U*S*V';
```

This matrix has a condition number $\kappa(A_1) = 10^5$, and a coherence (given the selected random seed) of $\mu(A_1) = 0.0236$, which confirms the incoherence of A_1 as the lower bound in this case is $n/m = 400/20000 = 0.02$. We also consider a coherent, ill-conditioned matrix A_2, given by:

```
A2 = [ diag(linspace(1,e5,400)); zeros(19600,400) ];
A2 = A2 + 1e-8*ones(20000,400);
```

This matrix has a condition number $\kappa(A_2) = 10^5$ and a coherence $\mu(A_2) = 1$, i.e. it is highly coherent. The following figures show the time required to converge for different values of γ ranging from 2 to 16, given a tolerance of 10^{-12}. The b vector is randomly initialized. Note that only one single row-mixing step is implemented, although it may be beneficial to implement more, particularly in the case of coherent matrices (5).

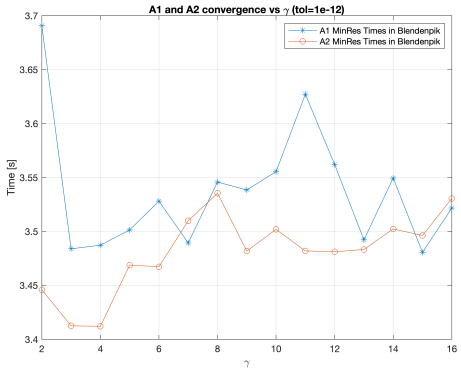
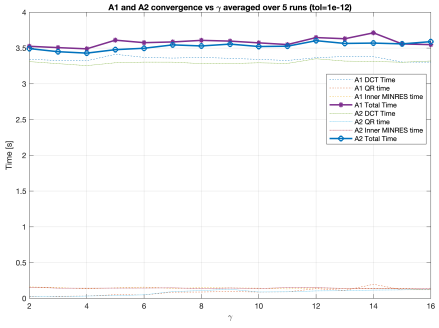
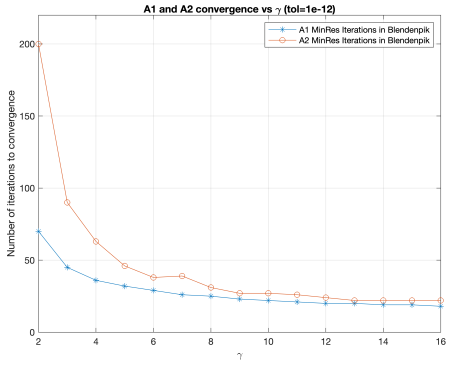


Figure 1: Convergence time of A_1 and A_2 as function of γ

From the above image it is not so clear to see a general trend in the running times as a function of the number of sampled rows as these vary every time (the random seed is not reset when selecting which rows to sample for different values of γ). Consequently we compute the average running time over 5 independent runs to get a clearer idea of a suitable γ, as shown in the figure below. Plotted alongside is the number of inner MINRES iterations required to converge to the desired tolerance for the tested values of γ.



(a) Breakdown of Blendenpik time to convergence of A_1 and A_2 averaged over 5 runs as a function of γ

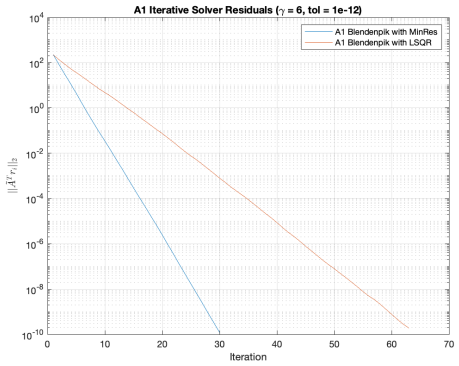


(b) Iterations to convergence of inner MINRES steps for A_1 and A_2 as function of γ

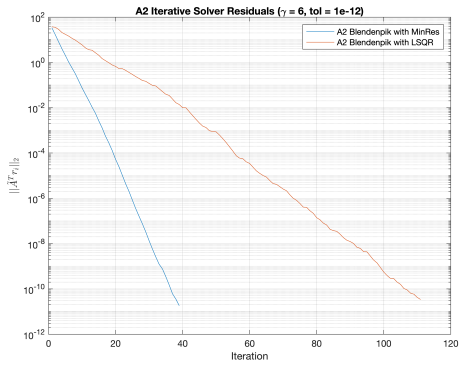
Based on the left figure, which presents a breakdown of the average running times of the main components of the Blendenpik algorithm, we see that a value of $\gamma = 6$ provides a good compromise between sampling a moderate amount of rows, namely $\gamma n = 2400$ rows in expectation, or 12% of the original number of rows, and obtaining performance improvements in the running time and inner MINRES steps (particularly for A_2), as shown in the right figure. It is also interesting to note that the DCT operation takes up the majority of the running time. We note that for small values of γ, Blendenpik takes more iterations to converge for A_2 as this is a coherent matrix, and as per the bound derived in 2, we observe that the larger coherence negatively affects the inner MINRES convergence rate due to a weaker bound on the condition number of the preconditioned matrix.

Furthermore we see that in both cases, increasing γ, i.e. increasing the number of sampled rows, leads to a reduced number of iterations to convergence. This corresponds to the inverse relationship in the theoretical bound between $\kappa(AR^{-1})$ and r, the number of sampled rows, which is equal to γn in expectation in our case. One final thing to note is that the improvement in the required number of iterations to convergence is smaller in the case of A_1, which is to be expected as incoherent matrices do not benefit as much from row-mixing as coherent matrices do.

In the following experiment we investigate the convergence of the Blendenpik algorithm using inner LSQR and MINRES steps for the two matrices defined above, using $\gamma = 6$. Note that in the following we consider the normal equations for both MINRES and LSQR, that is we consider the system $\tilde{A}^T\tilde{A}y = \tilde{A}^T b$, where $\tilde{A} = AR^{-1}$ and the solution of the original system is given by $x = R^{-1}y$.



(a) Residual norm $||\tilde{A_1}^T r_i||_2$ at each iteration using MINRES and LSQR inner steps for $\gamma = 5$



(b) Residual norm $||\tilde{A_2}^T r_i||_2$ at each iteration using MINRES and LSQR inner steps for $\gamma = 5$

Note that the norm we are plotting here is not exactly the same as in (5). Here we consider only the numerator, namely $||(AR^{-1})^T r_i||_2$. In both cases we observe that the convergence using MINRES is much faster than using LSQR when using the normal equations, taking less than half the number of iterations for both matrices.

5 Concluding Remarks

In this report we have considered the Blendenpik algorithm, which is a randomized algorithm for solving linear least squares problems via a random projection preconditioner. We have shown that the condition number of the preconditioned matrix AR^{-1}, where R comes from the QR decomposition of a subsample of A, is dependent on the coherence $\mu(A)$, motivating the use of randomized sampling and projection algorithms to guarantee incoherence with a high probability. In doing so, a good preconditioner can be efficiently obtained which is able to reduce the required number of iterations in the inner MINRES/ LSQR steps.

It is important to note that this algorithm performs best in the case of larger matrices, where it significantly outperforms standard LAPACK solvers. This is however not the case for small matrices, nearly square ones, and some sparse cases. Further it is important to consider that while the condition number of the preconditioned matrix $\kappa(AR^{-1})$ gives an upper bound on the convergence rate, in practice the convergence of the inner LSQR steps is dictated by the distribution of the singular values.

References

[1] S. Blackford. (1999) Qr factorization. [Online]. Available: https://www.netlib.org/lapack/lug/node40.html

[2] P. Drineas, M. W. Mahoney, S. Muthukrishnan, and T. Sarlós, "Faster least squares approximation," *CoRR*, vol. abs/0710.1435, 2007. [Online]. Available: http://arxiv.org/abs/0710.1435

[3] P. Drineas, M. Mahoney, and S. Muthukrishnan, "Relative-error cur matrix decompositions," *SIAM Journal on Matrix Analysis and Applications*, vol. 30, pp. 844–881, 05 2008.

[4] V. Rokhlin and M. Tygert, "A fast randomized algorithm for overdetermined linear least-squares regression," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, pp. 13 212–7, 09 2008.

[5] H. Avron and P. Maymounkov, "Blendenpik: Supercharging lapack's least-squares solver," *SIAM J. Scientific Computing*, vol. 32, pp. 1217–1236, 01 2010.