Data transformation and Dimension Reduction Lecture

Robert M. Gower*

November 8, 2019

Objective: When your data matrix $X \in \mathbb{R}^{d\times n}$ is too large to store on memory, or perform basic operations, dimension reduction tools become essential. Furthermore when we have far more features than data, dimension reduction tools can help avoid overfitting and thus decrease generalization/test error. Much of these notes are based on Chapter 5 of [6].

Contents

1 Dimension Reduction Maps

Suppose we are given training data $(x_i, y_i) \in \mathbb{R}^{d+1}$ for $i = 1, \ldots, n$. Let $X = [x_1, \ldots, x_n]$. What happens if not all the data fits onto your computer? One way to deal with large data is to sample rows or columns or compressed versions of the data. Here we study dimension reduction tools that compress the number of features.

Definition 1.1. We say that $f : \mathbb{R}^d \to \mathbb{R}^r$ is a dimension reduction map if $r < d$ and if the mapped data

$$Y = [f(x_1), \ldots, f(x_n)] \in \mathbb{R}^{r\times n},$$

is still useful for some application.

Being "useful" needs to be defined, and this will depend what you need the data for.

Here we consider two applications. In Section 2 we consider dimension reductions under which we can still recover the approximate solution to a regression/classification problem. In Section 3 we consider dimension reductions that preserve the pairwise distances between data points. That is, maps that are approximate isometries. This is relevant for clustering and kernal based methods.

*gowerrobert@gmail.com

2 Classification and PCA

2.1 Motivational Problem

When compressing data, some information will be lost. So we should choose what will be lost and what properties of the data we wish to preserve based on how we want to use this data. To start, let us consider the task of fitting a generalized linear model to data. We will then investigate under which dimension reductions is this task invariant. That is, which maps of the data can we perform so that the fit remains the same. We will then use this to motivate dimension reduction tools under which the fit is *approximately* invariant.

Consider the regularized empirial risk minimization problem with a linear hypothesis class:

$$\min_{w\in\mathbb{R}^d} F(w) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} \ell_i(\langle x_i, w\rangle), \tag{1}$$

where x_i is the given ith data point for $i = 1, \dots, n$, $\ell_i : \mathbb{R} \to \mathbb{R}_+$ is a convex loss function with respect to the ith label. For example in ridge regression we would have $\ell_i(y) = \frac{1}{2}(y - y_i)^2$ where y_i is the label of x_i. For simplicity we will not consider a regularizor.[1]

Here we look into what maps can we apply to the data matrix

$$X \stackrel{\text{def}}{=} [x_1, \dots, x_n] \in \mathbb{R}^{d\times n}, \tag{2}$$

that either make the problem (1) easier to solve, have better generalization properties or easier to vizualize.

First we focus on coordinate transformations, that are transformations under which we loose no information and thus can still find the exact solution to (1) even after applying a transformation to X.

2.2 Coordinate Transformations

First consider the transformation $w \to Pw$ where $P \in \mathbb{R}^{d\times d}$ and the resulting problem

$$\hat{w}^* = \arg\min_{w\in\mathbb{R}^d} \hat{F}(w) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} \ell_i(\langle x_i, Pw\rangle). \tag{3}$$

Exercise 2.1. Show that transforming the data using $X \to P^\top X$ is equivalent to applying the coordinate change $w \to Pw$.

Since $\hat{F}(w)$ is the composition of a convex function with a linear function, we know that $\hat{F}(w)$ is convex, and thus the first order optimality conditions are necessary and sufficient. That is, using the chain rule, the solution $\hat{w}^*$ to (3) is such that

$$\nabla\hat{F}(\hat{w}^*) = P^\top \nabla F(P\hat{w}^*) = 0. \tag{4}$$

How can we now recover the solution w^* to (1)? If P is invertible then right multiplying the above by P^{-1} gives $\nabla F(P\hat{w}^*) = 0$, which shows that $w^* = P\hat{w}^*$ is the solution to (1).

We can still recover the solution to (1) even when P is not invertible. Indeed, writing out (4) explicitly we have that

$$\nabla\hat{F}(w) = \frac{1}{n}\sum_{i=1}^{n} \ell_i'(\langle x_i, Pw\rangle)P^\top x_i = \frac{1}{n}P^\top X \underbrace{[\ell_1'(\langle x_1, Pw\rangle), \dots, \ell_n'(\langle x_n, Pw\rangle)]^\top}_{\stackrel{\text{def}}{=} L(Pw)} = 0. \tag{5}$$

We can use the above to prove the following

[1] Add a regularizer and regularization parameter $\lambda R(w)$ complicates matter because λ is chosen as a function of the data, and here we will consider several maps of the data. Thus we would have to have an explicit model for the function $\lambda(X)$.

Theorem 2.2 (Range space preserving)**.** If

$$\mathbf{Range}\left(X^\top\right) = \mathbf{Range}\left(X^\top P\right), \tag{6}$$

then every $\hat{w}^*$ that solves (5) we have that

$$w^* = P\hat{w}^*, \tag{7}$$

solves (1).

Proof. From (5) we have that $P^\top XL(P\hat{w}^*) = 0$. For $w^* = P\hat{w}^*$ to be a solution to (1) we require that $XL(w^*) = 0$. Fortunately the assumption (6) guarantees that these two condition are equivalent. Indeed, a sufficient condition for (5) and $XL(w^*) = 0$ to be equivalent would be if for every $z \in \mathbb{R}^d$ we have that

$$P^\top Xz = 0 \quad \Leftrightarrow \quad Xz = 0.$$

That is if $\mathbf{Null}\left(P^\top X\right) = \mathbf{Null}\left(X\right)$. Taking the orthogonal component over $\mathbf{Range}\left(X^\top\right) = \mathbf{Range}\left(X^\top P\right)$ gives just that since $\mathbf{Range}\left(X^\top\right)^\perp = \mathbf{Null}\left(P^\top X\right)$ and $\mathbf{Range}\left(X^\top P\right)^\perp = \mathbf{Null}\left(X\right)$. □

In particular, using Lemma 4.1 we have that if there exists symmetric positive definite matrix $G \in \mathbb{R}^{d\times d}$ such that $P = GX$ then (6) holds. Turn to the exercise sheet and now prove that $\mathbf{Range}\left(X^\top\right) = \mathbf{Range}\left(X^\top GX\right)$ for $G \in \mathbb{R}^{d\times d}$ symmetric positive definite.

This shows that P need not be invertible. In particular, this suggests that P could be the projection operator onto $\mathbf{Range}\left(X\right)$, which motivates the use of projection based dimension reduction tools such as PCA, as we will see in Section 2.4.

2.3 Data transformations as a modelling choice

Though we have explored data transformation through the perspective of coordinate changes that preserve the original solution, data transformations can be a modelling choice, and need not preserve the original problem. For instance, if we knew before hand that the output labels y_i are all positive, we may choose to apply a log transform. That is, by applying $\hat{y}_i = \log(y_i)$ for $i = 1, \ldots, n$. Then accordingly, after fitting $\hat{w}^*$ to this transformed data, we can predict the label of a new data point x_i using $\exp(\langle x_i, \hat{w}^*\rangle)$. This enforces positive predictions, though there is no longer a one-to-one correspondence between the solution $\hat{w}^*$ of the transformed problem and the solution to the original problem.

2.4 Principal Component Analysis

Here we consider linear dimension reduction tools that fit the following format.

Definition 2.3. Let $X \in \mathbb{R}^{d\times n}$. We say that $P \in \mathbb{R}^{d\times r}$ is a linear dimension reducing operator and $Y = P^\top X \in \mathbb{R}^{r\times n}$ the resulting low dimensional transformed data.

How should we choose the transformation P? From Theorem 2.2 we would like $\mathbf{Range}\left(X^\top P\right) \approx \mathbf{Range}\left(X^\top\right)$. Using Lemma 4.2 in the appendix we have that $\mathbf{Range}\left(X^\top P\right) = \mathbf{Range}\left(X^\top PP^\top\right)$. This form is better because now $X^\top PP^\top$ has the same dimension as $X^\top$. So we can instead look for P such that

$$X^\top \approx X^\top PP^\top \quad \Leftrightarrow \quad X \approx PP^\top X.$$

Thus we can consider the optimization problem

$$P \in \arg\min_{W\in\mathbb{R}^{d\times r}} \left\|X - WW^\top X\right\|_F^2. \tag{8}$$

Note: Now do the exercise of proving Lemma 4.2.

The solution to (8) is given by the leading eigenvectors of A. This is sometimes also known as the Principle Component Analysis transform.

Note: Revise Singular Value Decomposition and Eckart-Young-Mirks Theorem before moving on.

Theorem 2.4. Let $d \geq n \geq r$. Let $X = U\Sigma V^\top$ be a Singular Value Decomposition (SVD) decomposition of X where $U = [u_1, \ldots, u_d] \in \mathbb{R}^{d\times d}$ is an orthogonal matrix, $D = \mathbf{diag}\,(\sigma_1, \ldots, \sigma_d) \in \mathbb{R}^{d\times d}$ and $V \in \mathbb{R}^{n\times d}$ with $VV^\top = I$. Let $U_r = [u_1, \ldots, u_r] \in \mathbb{R}^{d\times r}$. The solution to (8) is the Principal Component Analysis (PCA) transform given by $P = U_r$

Proof. **Proof relying on SVD properties.**

It is well known that

$$U_r\Sigma_{r\times r}V_r^\top \in \arg\min_{\mathbf{Rank}(Z)=r} \|X - Z\|_F^2\,, \tag{9}$$

where $U_r\Sigma_{r\times r}V_r^\top \in \mathbb{R}^{d\times n}$, $\Sigma = \mathbf{diag}\,(\sigma_1, \ldots, \sigma_r) \in \mathbb{R}^{r\times r}$ and $V_r = [v_1, \ldots, v_r] \in \mathbb{R}^{n\times r}$. This result (9) is known as the Eckart-Young-Mirks Theorem.

Since $WW^\top X$ has rank r, we have that

$$\left\|X - WW^\top X\right\|_F^2 \geq \left\|X - U_r\Sigma_{r\times r}V_r^\top\right\|_F^2. \tag{10}$$

Furthermore by choosing $W = U_r$ we have that

$$WW^\top X = U_rU_r^\top U\Sigma V^\top = U_r[\Sigma_{r\times r}0]V^\top = U_r\Sigma_{r\times r}V_r^\top.$$

Thus the lower bound (10) is achieved and thus $W = U_r$ must be the minimum.
See Section 23.1 in [6] for a different proof that does not rely on the existence of an SVD decomposition. □

Exercise 2.5. Consider the setting of Theorem 2.4. Assume that X has rank $r \in \mathbb{N}$. Show that

$$\mathbf{Range}\left(U_rU_r^\top X\right) = \mathbf{Range}\left(U_rX\right) = \mathbf{Range}\left(X\right).$$

Show that as a consequence, by Theorem 2.2, after applying the PCA transform to the data $\hat{x} \to U_r^\top x \in \mathbb{R}^r$, one can still recover the exact solution to (1) using $w^* \leftarrow U_r\hat{w}^*$.

Proof. Since X has rank r, then the SVD of X reduces to $X = U_r\Sigma_{r\times r}V_r^\top$. By Lemma 4.2 we have that

$$\mathbf{Range}\left(U_r^\top X\right) = \mathbf{Range}\left(U_rU_r^\top X\right) = \mathbf{Range}\left(U_r\Sigma_{r\times r}V_r^\top\right) = \mathbf{Range}\left(X\right),$$

and thus by Theorem 2.2 we have that that $w^* \leftarrow U_r\hat{w}^*$ is a solution to (1). □

In practice it may be hard to determine the rank of the matrix X. Furthermore, it seems reasonable now to suspect that PCA may work well even if we choose r leading eigenvectors associated to the largest r eigenvalues.

Remark 2.6 (Practical implementation of PCA)**.** *Theorem 2.4 states that the PCA transform is given by the leading eigenvectors of $A = XX^\top \in \mathbb{R}^{d\times d}$. Computing the complete spectral decomposition of A costs $O(d^3)$ which may be prohibitive if d is very large. Fortunately when $n < d$ we can compute the PCA transform using $O(n^3)$. The trick is to realize that the matrix $B = X^\top X$ has the same non-zero eigenvalues of A and furthermore, if v is an eigenvector of B then Xv is an eigenvector of A. As an exercise you should prove this and also write the pseudo-code for an efficient PCA algorithm that takes X, n and d as inputs and makes use of this trick (assume you can call an SVD subroutine).*

When the dimensions of data are trully large, computing eigenvectors may be prohibitively expensive.

3 Random Projections and the Johnson-Lindenstrauss Lemma

Calculating the PCA transformation requires computing the leading r eigenvectors of a possibly large dimensional matrix. This can be prohibitively expensive for certain applications. Fortunately it turns out that *very* high dimensional data has a special property in that it can be efficiently mapped to a low dimension space while preserving the distance between points.

Furthermore for certain applications, it is more important to preserve pairwise distances. Such as in distance based clustering or kernel based methods.

Our original objective was to minimize the sum of the error for each data point

$$\left\|X - WW^\top X\right\|_F^2 = \sum_{i=1}^n \left\|x_i - WW^\top x_i\right\|_2^2.$$

In the exercise list we also show that, when X has been centred, this is equivalent to maximizing the scattering of projected data points

$$\max \sum_{i,j=1}^n \left\|W^\top x_i - W^\top x_j\right\|_2^2.$$

Now we take a slightly different approach, and investigate if the *distance* between datapoints can be preserved after begin embedded in a low dimensional space. That is, can we find $M \in \mathbb{R}^{d\times r}$ such that

$$\left\|W^\top x_i - W^\top x_j\right\|_2 \approx \left\|x_i - x_j\right\|_2.$$

It turns out that this can be done, upto a certain precision, by using random maps.

Lemma 3.1 (Johnson-Lindenstrauss Lemma [3])**.** Let $X = [x_1, \ldots, x_n] \in \mathbb{R}^{d\times n}$. Let $\delta \in (0,\, 1)$, $\epsilon < 3$ and $r \in \mathbb{N}$ such

$$r \geq 12\frac{\log(n/\delta)}{\epsilon^2}. \tag{11}$$

Let $W \in \mathbb{R}^{d\times r}$ be such that each element is normally distributed with zero mean and 1 variance. We have that

$$(1-\epsilon)\left\|x_i - x_j\right\|_2^2 \;\leq\; \left\|\left(1/\sqrt{r}\right)W^\top x_i - \left(1/\sqrt{r}\right)W^\top x_j\right\|_2^2 \;\leq\; (1+\epsilon)\left\|x_i - x_j\right\|_2^2, \quad \forall i,j = 1,\ldots,n, \tag{12}$$

with probability $1-\delta$ over the choice of W.

This lemma holds for many distributions over W other than simply Gaussian as we will see later. The JL Lemma is remarkable in that it does not depend on the dimension of the data d and has only a very weak log() dependency on the number of data points. Though it has a strong dependency on the accuracy ϵ. As an example, say we wanted to preserve the pairwise distances to with an error of $\epsilon = 10^{-2} = 1\%$ with high probability $1-\delta = 99\%$. Ignoring the log terms, the JL Lemma states that the embedding dimension needs to be $r = O(10^4)$. While this is useless if $d \leq 10^4$, what if d is much larger, such as $d = 10^9$?

Proof of JL Lemma 3.1. For a given $x \in \mathbb{R}^d$ we have that the ith row of $W^\top x$ is $W_{:i}^\top x$ which is linear combination of Gaussian random variables, and is thus Gaussian with mean 0 and variance $\left\|x\right\|_2^2$. Thus $(W_{:i}^\top x)/\left\|x\right\|_2$ is a standard normal random variable with mean 0 and variance 1. Thus $\left\|Wx\right\|_2^2/\left\|x\right\|_2^2 = \sum_{i=1}^r (W_{:i}^\top x)^2/\left\|x\right\|_2^2$ has a χ_r^2 distribution. A well known concentration of measure for χ_r^2 distributions states that, for $\epsilon < 3$ we have that

$$\mathbb{P}\left[\left|\frac{1}{r}\frac{\left\|W^\top x\right\|_2^2}{\left\|x\right\|_2^2} - 1\right| \geq \epsilon\right] \quad \leq \quad 2e^{-\epsilon^2 r/6}. \tag{13}$$

which is proven in the Appendix in Lemma 4.3 and in Lemma B.12 in [6].

The proof now follows by applying a union bound over the vectors $(x_i - x_j)$ for all pairs of indices, that is for $1 \le i < j \le n$, we have that

$$\mathbb{P}\left[\bigcup_{1\le i\le j\le n}\left\{\left|\frac{\left\|(1/\sqrt{r})\,W^\top(x_i-x_j)\right\|_2^2}{\|x_i-x_j\|_2^2}-1\right|\ge\epsilon\right\}\right] \quad\le\quad \sum_{1\le i\le j\le n}\mathbb{P}\left[\left|\frac{\left\|(1/\sqrt{r})\,W^\top(x_i-x_j)\right\|_2^2}{\|x_i-x_j\|_2^2}-1\right|\ge\epsilon\right]$$
$$\overset{(13)}{\le}\quad \binom{n}{2}2e^{-\epsilon^2 r/6}\le n^2e^{-\epsilon^2 r/6}. \tag{14}$$

Now choosing r such that $n^2e^{-\epsilon^2 r/6}\le\delta$ gives $r\ge 12\frac{\log(n/\sqrt{\delta})}{\epsilon^2}$ which we can enforce using

$$r\ge 12\frac{\log(n/\delta)}{\epsilon^2}\ge 12\frac{\log(n/\sqrt{\delta})}{\epsilon^2},$$

which gives the bound (11). Taking the complement of the event (14) we have that

$$\mathbb{P}\left[\bigcap_{1\le i\le j\le n}\left\{\left|\frac{\left\|(1/\sqrt{r})\,W^\top(x_i-x_j)\right\|_2^2}{\|x_i-x_j\|_2^2}-1\right|\le\epsilon\right\}\right]\ge 1-\delta, \tag{15}$$

which gives (12). □

Calculating SVD decompositions of such large dimensional data would cost of the order of $O(nd^2)$ and thus would not be possible on most personal computers. Multiplying with a relatively small $r\times d$ Gaussian matrix costs $O(rdn)$ operations. Even this cost can be significantly reduced.

Very recently in 2017 it was shown by Larsen and Nelson [4] that there exists of a particular data matrix $X\in\mathbb{R}^{d\times n}$ such that any transform satisfying (12) must have at $r=O(\epsilon^{-2}\log(n))$ dimension, thus showing that the JL Lemma is tight. Thus there is no hope in improving the lemma in terms of the embedding dimension. What can be improved is the bottleneck cost of computing the matrix-matrix product $W^\top X$. This has been improved by making W sparse [1] or defining W as implicit transform such that there exist fast recurrsive algorithms for computing $W^\top X$, such as subsampled Fourier transforms [2].

For example, the Achlioptas transform $(1/\sqrt{r})\,W\in\mathbb{R}^{d\times r}$ where each element is sampled independently via

$$W_{ij}=\sqrt{3}\begin{cases}+1 & \text{with probability } 1/6\\ 0 & \text{with probability } 2/3\\ -1 & \text{with probability } 1/6,\end{cases} \tag{16}$$

also satisfies the JL Lemma.

Example 3.2. Show that each element of the Achlioptas transform [1] satisfies

$$\mathbb{E}[W_{:j}^\top x]=0\quad\text{and}\quad\mathbb{E}[(W_{:j}^\top x)^2]=\|x\|_2^2\,. \tag{17}$$

Conclude that $\mathbb{E}[\left\|\frac{1}{\sqrt{r}}W^\top x\right\|^2\Big/\|x\|_2^2]=1$ and thus the JL lemma would follow with this W by using an appropriate concentration bound of the random variable $\frac{\left\|\frac{1}{\sqrt{r}}W^\top x\right\|^2}{\|x\|_2^2}$ around its mean.

Proof. Clearly $\mathbb{E}[W_{ij}]=\sqrt{3}(1/6-1/6)=0$. Thus

$$\mathbb{E}[W_{:j}^\top x]=\mathbb{E}[W_{:j}]^\top x=0.$$

For the second moment we have that

$$\mathbb{E}[(W_{:j}^\top x)^2]\quad=\quad x^\top\mathbb{E}[W_{:j}W_{:j}^\top]x=x^\top x=\|x\|_2^2\,, \tag{18}$$

where we used that $\mathbb{E}[W_{ij}W_{kj}] = 0$ if $i \neq k$ since the elements are sampled independently and

$$\mathbb{E}[W_{ij}^2] = 3(1/6 + 1/6) = 1.$$

Thus finally

$$\frac{\mathbb{E}[\left\| \frac{1}{\sqrt{r}} W^\top x \right\|^2]}{\|x\|_2^2} = \frac{1}{r\|x\|_2^2} \sum_{j=1}^{r} \mathbb{E}[(W_{:j}^\top x)^2] \overset{(17)}{=} \frac{1}{r\|x\|_2^2} r\|x\|_2^2 = 1.$$

Now we just need apply Lemma (3.3) taken from [1] together with a union bound as used in (15) to get our result.

Lemma 3.3.

$$\mathbb{P}\left[\left|\frac{1}{r}\frac{\|W^\top x\|_2^2}{\|x\|_2^2} - 1\right| \geq \epsilon\right] \quad \leq \quad 2e^{-r(\epsilon^2/2 - \epsilon^3/3)} \quad \leq \quad 2e^{-r\epsilon^2/2}. \tag{19}$$

□

3.1 Sparse Matrix Formats

To use the Achlioptas transform (16) efficiently, it needs to be implemented using sparse linear algebra. Assuming X is sparse, as it often is in most machine learning problems, the matrix $W \in \mathbb{R}^{d\times r}$ needs to be generated in a sparse matrix format in such a way that the product $W^\top X$ (or equivalently $X^\top W$) can be efficiently computed.

The Compressed Sparse Column (CSC) Format is a sparse matrix representation such that left matrix multiplications can be computed efficiently, for example $X^\top W$. Let **nnz** be the number of non-zero elements in W. The CSC format represents a matrix W using three arrays:

$$\texttt{data} \in \mathbb{R}^{\mathbf{nnz}}, \quad \texttt{rowindex} \in \mathbb{R}^{\mathbf{nnz}}, \quad \texttt{colptr} \in \mathbb{R}^{r+1}.$$

The `data` array stores all the nonzero elements of W ordered by top-to-bottom left-to-right, otherwise known as the *column-major* order, see Figure 2. That is,

$$\texttt{data} = [w_1, \ldots, w_{\mathbf{nnz}}],$$

where w_i are the nonzero elements of W. The array `rowindex` contains the row indexes of each nonzero element. That is $\texttt{rowindex}_i$ contains the column index of w_i. Finally the jth element of $\texttt{colptr}_j$ is the number of nonzeros in the first i columns. Specifically, `colptr` is defined recursively as

$$\begin{aligned} \texttt{colptr}_0 &= 0 \\ \texttt{colptr}_{j+1} &= \texttt{colptr}_j + \mathbf{nnz}(W_{:j}), \quad \text{for } j = 1, \ldots, r. \end{aligned}$$

Here we use $\mathbf{nnz}(W_{i:})$ to denote the number of nonzero elements in the ith row of W. Consequently $\texttt{rowptr}_{n+1} = \mathbf{nnz}$.

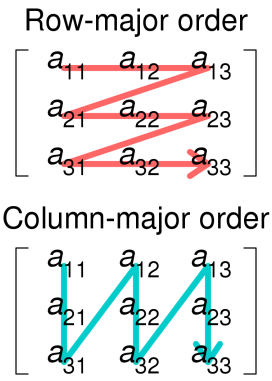

Figure 1: Row-major and Colum-major order for transversing the elements of a matrix (image taken from Wikipedia).

The Compressed Sparse Row (CSR) Format is sparse matrix representation such that right matrix multiplications can be done efficiently. Let **nnz** be the number of non-zero elements in W. The CSR format represents a matrix W using three arrays:

$$\texttt{data} \in \mathbb{R}^{\mathbf{nnz}}, \quad \texttt{colindex} \in \mathbb{R}^{\mathbf{nnz}}, \quad \texttt{rowptr} \in \mathbb{R}^{d+1}.$$

The `data` array stores all the nonzero elements of W ordered left-to-right top-to-bottom, otherwise known as the *row-major* order, see Figure 2. That is,

$$\texttt{data} = [w_1, \ldots, w_{\mathbf{nnz}}],$$

where w_i are the nonzero elements of W. The array `colindex` contains the column indexes of each nonzero element. That is $\texttt{colindex}_i$ contains the column index of w_i. Finally `rowptr` is defined recursively as

$$\begin{aligned} \texttt{rowptr}_0 &= 0 \\ \texttt{rowptr}_{i+1} &= \texttt{rowptr}_i + \mathbf{nnz}(W_{i:}), \quad \text{for } i = 1, \ldots, d. \end{aligned}$$

Here we use $\mathbf{nnz}(W_{i:})$ to denote the number of nonzero elements in the ith row of W. Consequently $\texttt{rowptr}_{n+1} = \mathbf{nnz}$. As an example, consider the matrix

$$W = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 5 & 8 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 6 & 0 & 0 \end{bmatrix}.$$

The CSC format of this matrix is given by the three arrays

$$\begin{aligned} \texttt{data} &= \begin{bmatrix} 5 & 8 & 6 & 3 & 1 \end{bmatrix} \\ \texttt{rowindex} &= \begin{bmatrix} 2 & 2 & 4 & 3 & 1 \end{bmatrix} \\ \texttt{colptr} &= \begin{bmatrix} 0 & 1 & 3 & 4 & 5 \end{bmatrix} \end{aligned}$$

The CSR format of this matrix is given by the three arrays

$$\begin{aligned} \texttt{data} &= \begin{bmatrix} 1 & 5 & 8 & 3 & 6 \end{bmatrix} \\ \texttt{colindex} &= \begin{bmatrix} 4 & 1 & 2 & 3 & 2 \end{bmatrix} \\ \texttt{rowptr} &= \begin{bmatrix} 0 & 1 & 3 & 4 & 5 \end{bmatrix} \end{aligned}$$

4 Appendix

Lemma 4.1. For any matrix W and symmetric positive definite matrix G,

$$\mathbf{Null}\left(W\right) = \mathbf{Null}\left(W^\top G W\right) \tag{20}$$

and

$$\mathbf{Range}\left(W^\top\right) = \mathbf{Range}\left(W^\top G W\right). \tag{21}$$

Proof. In order to establish (20), it suffices to show the inclusion $\mathbf{Null}\left(W\right) \supseteq \mathbf{Null}\left(W^\top G W\right)$ since the reverse inclusion trivially holds. Letting $s \in \mathbf{Null}\left(W^\top G W\right)$, we see that $\|G^{1/2} W s\|^2 = 0$, which implies $G^{1/2} W s = 0$. Therefore, $s \in \mathbf{Null}\left(W\right)$. Finally, (21) follows from (20) by taking orthogonal complements. Indeed, $\mathbf{Range}\left(W^\top\right)$ is the orthogonal complement of $\mathbf{Null}\left(W\right)$ and $\mathbf{Range}\left(W^\top G W\right)$ is the orthogonal complement of $\mathbf{Null}\left(W^\top G W\right)$. □

Lemma 4.2. For any matrix $M \in \mathbb{R}^{d \times k}$ and $X \in \mathbb{R}^{d \times n}$ we have that

$$\mathbf{Null}\left(W^\top X\right) = \mathbf{Null}\left(W W^\top X\right) \tag{22}$$

and

$$\mathbf{Range}\left(X^\top W W^\top\right) = \mathbf{Range}\left(X^\top W\right). \tag{23}$$

Proof. In order to establish (22), it suffices to show the inclusion $\mathbf{Null}\left(W^\top X\right) \supseteq \mathbf{Null}\left(WW^\top X\right)$ since the reverse inclusion trivially holds. Letting $s \in \mathbf{Null}\left(WW^\top X\right)$, thus $WW^\top Xs = 0$. Let multiplying by $s^\top X^\top$ we have that $\|W^\top Xs\|^2 = 0$, which implies $W^\top Xs = 0$. Therefore, $s \in \mathbf{Null}\left(W\right)$. Finally, (23) follows from (22) by taking orthogonal complements. □

Lemma 4.3 (χ^2_r concentration)**.** Let $Z \sim \chi^2_r$. Then for all $0 \leq \epsilon < 3$ we have that Z is concentrated around its mean r according to

$$\mathbb{P}\left[\,(1-\epsilon)r \leq Z \leq (1+\epsilon)r\,\right] \quad \geq \quad 1 - 2e^{-\epsilon^2 r/6}. \tag{24}$$

In other words, taking the complement of this event, and dividing within the probability by r we have that

$$\mathbb{P}\left[\,|\,(1/r)\,Z - 1| \geq \epsilon\,\right] \quad \leq \quad 2e^{-\epsilon^2 r/6}. \tag{25}$$

Proof. Let $Z = \sum_{i=1}^r X_i^2$ where each $X_i \sim \mathcal{N}(0,1)$ is a standard normal random variable. We make use of the closed form expression for the moment generating function of χ^2_r variables

$$\mathbb{E}[e^{\lambda Z}] = (1-2\lambda)^{-r/2}, \quad \forall \lambda < \frac{1}{2}. \tag{26}$$

The proof follows by applying Markov's inequality together with (26) and can be found in Lemma 1 in [5]. □

References

[1] D. Achlioptas. "Database-friendly Random Projections: Johnson-Lindenstrauss with Binary Coins". *J. Comput. Syst. Sci.* 66.4 (June 2003), pp. 671–687.

[2] N. Ailon and B. Chazelle. "The Fast Johnson-Lindenstrauss Transform and Approximate Nearest Neighbors". *SIAM J. Comput.* 39.1 (May 2009), pp. 302–322.

[3] W. Johnson and J. Lindenstrauss. "Extensions of Lipschitz mappings into a Hilbert space". In: *Conference in modern analysis and probability (New Haven, Conn., 1982)*. Vol. 26. Contemporary Mathematics. American Mathematical Society, 1984, pp. 189–206.

[4] K. G. Larsen and J. Nelson. "Optimality of the Johnson-Lindenstrauss Lemma". In: *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*. 2017, pp. 633–638.

[5] B. Laurent and P. Massart. "Adaptive estimation of a quadratic functional by model selection". *The Annals of Statistics* 28.5 (2000), pp. 1302–1338.

[6] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. New York, NY, USA: Cambridge University Press, 2014.