STATISTICAL LEARNING

BRAD BAXTER

[**Version: 202404171320**]. *You can access more material from my server, including the PDF version of these notes:* `http://econ109.econ.bbk.ac.uk/brad/teaching/sl/slnotes.pdf`. *My slides are based on these notes.*

Abstract. **These notes contain all examinable theoretical material for the course**.

Contents

1. Introduction

The term "Statistical Learning" is poorly defined, but most problems fit into the following framework in one way or another: we are given observation vectors $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n$ in $\mathbb{R}^d$ and want to discover some useful pattern, or trend, in the data. Typically n and d are large, but this is not an essential feature. Thus "Statistical Learning" is really just a new name for statistical analysis combined with numerical analysis, but there are some important reasons which justify the jargon's existence. In particular, we shall devote several lectures to the use of *radial basis functions (RBFs)* for data fitting. RBFs are becoming vital tools in several fields closely related to statistics, such as learning theory, neural networks and Support

Vector Machines. We shall begin by studying other methods, including clustering algorithms and the famous *PageRank* algorithm, which forms the heart of Google. Further, an important theme will be numerical linear algebra. In particular, we shall introduce the Singular Value Decomposition to study linear and nonlinear least squares problems, including several matrix nearness problems. Successful use of these techniques requires a solid theoretical foundation, so this course is *not* a recipe book. We shall also be using MATLAB although examination questions will not test MATLAB knowledge.

Terminology is also fluid in all modern applications of applied mathematics and statistics to anything now called learning. In the MSc Statistics Handbook, **regression** is the general name used for all of the linear and nonlinear least squares methods considered in this course, while **clustering** refers to any algorithm such as *PageRank* or k-means. **Kernel methods** (including **support vector machines**) are here studied via Radial Basis Functions. The newer names are more fashionable, but I prefer the accuracy of the older names. Thus an algorithm designed to estimate parameters by minimizing a sum of squares on some sample set is often described as a learning method trained on the sample set.

This is **NOT** a course intended to provide numerical recipes to be used without true understanding of the algorithms. In my view, modern applications of mathematics and computing are plagued by the misconception that the algorithms can be used without any real comprehension of why they work and when they don't: this is the same phenomenon that led to the well-known saying "There are three kinds of lies: lies, damned lies and statistics!". We shall be delving into the mathematical details that underpin these and many other algorithms.

I founded this course in the early 2000s, initially named "Data Mining". I'm now lecturing the course for the first time in approximately a decade and have decided to add new material. You can also complete the assignment using these notes. Further, the presentation in these notes is much more concise than in lectures. I shall sometimes provide examples in MATLAB and R, but your examination will not test your knowledge of these software packages.

I have made my book "Foundations of Scientific Computing" available on the server (`nabook.pdf`), which will help you with some of the background numerical analysis.

Notation: I shall use $\mathbb{R}$ to denote the real numbers, $\mathbb{R}^n$ to denote real n-dimensional space, and $\mathbb{R}^{m\times n}$ to denote the set of all $m \times n$ real matrices.

2. PageRank

How do we construct a search engine? The PageRank algorithm was devised by Page and Brin in 1994, the founders of Google, and provides an excellent example of linear algebra applied to statistical learning.

We can summarize the links between websites by a single matrix containing 0s and 1s. Specifically, if there are N websites, then we let $W_{ij} = 1$ if site i links to site j and $i \neq j$, but otherwise set $W_{ij} = 0$. At present Google uses $N \approx 10^9$, so almost all the elements of W are zero (why?); we say that W is a *sparse matrix*.

Page and Brin decided to rank these N websites by simulating user behaviour with a Markov model based on the connectivity matrix W. [All students have indicated some familiarity with the basics of Markov chains, but any standard textbook should contain the fundamental properties used here.] Specifically, we imagine vast numbers of users surfing the web in discrete time. At the kth step, the vector $\boldsymbol{\pi}^{(k)}$ denotes the probability distribution for our users, that is, $\pi_i^{(k)}$ is the probability that a user is surfing site i at time k. We then let our users surf to new sites according to the transition matrix $P \in \mathbb{R}^{N\times N}$, where

$$P_{ij} = \frac{W_{ij}}{\sum_{k=1}^{N} W_{ik}}, \qquad 1 \leq i, j \leq N. \tag{2.1}$$

Further, we shall assume that $\sum_{k=1}^{n} W_{ik} \neq 0$, for all i, to avoid a zero denominator in the definition of P (we are assuming that there are no *dangling pages*, to use Google's jargon). Thus every row of P contains non-negative numbers summing to one.

Therefore the new probability vector is given by

$$\boldsymbol{\pi}^{(k+1)} = P^T \boldsymbol{\pi}^{(k)} \tag{2.2}$$

and, over time, we hope to obtain an *invariant measure* (or stationary probability vector) $\boldsymbol{\pi}$. Unfortunately this Markov chain turns out to be inadequate, because most sites tend to fall into isolated clusters and it inherits this stagnation. One way to avoid this is a *teleporting random walk*: we choose a parameter $c \in (0, 1)$ and *either* use P with probability c, *or* move to one of the N websites with equal probability. Thus our new transition matrix is

$$M = cP + (1 - c)\frac{\mathbf{e}\mathbf{e}^T}{N}, \tag{2.3}$$

where

$$\mathbf{e} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}. \tag{2.4}$$

The new invariant measure vector $\boldsymbol{\pi}$ now satisfies $M^T\boldsymbol{\pi} = \boldsymbol{\pi}$.

Exercise 2.1. *Show that $\boldsymbol{\pi}$ satisfies*

$$cP^T\boldsymbol{\pi} + (1 - c)\frac{\mathbf{e}}{N} = \boldsymbol{\pi}. \tag{2.5}$$

Page and Brin decided to define the *rank vector* $\mathbf{r} = N\boldsymbol{\pi}$. Thus the last equation becomes

$$\left(I - cP^T\right)\mathbf{r} = (1 - c)\mathbf{e}. \tag{2.6}$$

This linear system contains N linear equations in N unknowns, but $N \approx 10^9$. Unfortunately, direct elimination requires $T(N) = CN^3$ seconds, where $T(10^3) \approx 1$ on basic modern computer.

Exercise 2.2. *Calculate $T(10^9)$ assuming that $T(10^3) = 1$. (One year contains approximately 3×10^7 seconds.)*

Thus elimination is completely unsuitable for solving (2.6). Fortunately, a simple iterative algorithm called *Jacobi's method* is available. Specifically, given any $n \times n$ matrix A, Jacobi's method attempts to solve $A\mathbf{x} = \mathbf{y}$ as follows. We first choose any initial vector $\mathbf{x}^{(0)}$. Then, given $\mathbf{x}^{(k-1)}$, we define $\mathbf{x}^{(k)}$ by the equation

$$x_i^{(k)} = \frac{y_i}{A_{ii}} - \sum_{j=1, j\neq i}^{n} \left(\frac{A_{ij}}{A_{ii}}\right) x_j^{(k-1)}, \qquad 1 \leq i \leq n. \tag{2.7}$$

It is important to understand that Jacobi's method is **not** a good method for solving a general linear system $A\mathbf{x} = \mathbf{y}$, but can be efficient if we are lucky in our choice of A, as turns out to be the case for PageRank.

Exercise 2.3. *Show that Jacobi* (2.7) *applied to* (2.6) *becomes*

$$\mathbf{r}^{(k)} = cP^T\mathbf{r}^{(k-1)} + (1-c)\mathbf{e}. \tag{2.8}$$

The analysis of convergence in Jacobi's method is much more suited to the L_1 *norm*, which is defined by

$$\|\mathbf{v}\|_1 = \sum_{1\leq k\leq n} |v_k|, \qquad \mathbf{v} \in \mathbb{R}^n. \tag{2.9}$$

Theorem 2.1. *Let $\mathbf{r}^{(0)}$ be any initial vector and generate $\mathbf{r}^{(k)}$ using Jacobi's method* (2.8) *applied to* (2.6)*. Then the kth stage error $\mathbf{e}^{(k)} = \mathbf{r}^{(k)} - \mathbf{r}$ satisfies*

$$\|\mathbf{e}^{(k)}\|_1 \leq c\|\mathbf{e}^{(k-1)}\|_1, \qquad k \geq 1. \tag{2.10}$$

Hence $\|\mathbf{e}^{(k)}\|_1 \leq c^k\|\mathbf{e}^{(0)}\|_1$ and, since $0 \leq c < 1$, we deduce that $\|\mathbf{e}^{(k)}\|_1 \to 0$, as $k \to \infty$, for any initial vector $\mathbf{r}^{(0)}$.

Proof. Subtracting (2.6) from (2.8) provides

$$\mathbf{e}^{(k)} = cP^T\mathbf{e}^{(k-1)}. \tag{2.11}$$

Now

$$\begin{aligned}
\|P^T\mathbf{e}^{(k-1)}\|_1 &= \sum_{i=1}^{N} \left|\left(P^T\mathbf{e}^{(k-1)}\right)_i\right| \\
&= \sum_{i=1}^{N} \left|\sum_{j=1}^{N} (P^T)_{ij} e_j^{(k-1)}\right| \\
&\leq \sum_{i=1}^{N}\sum_{j=1}^{N} |(P^T)_{ij}||e_j^{(k-1)}| = \sum_{j=1}^{N} |e_j^{(k-1)}| \sum_{i=1}^{N} P_{ji} = \|\mathbf{e}^{(k-1)}\|_1,
\end{aligned}$$

using the fact that, by construction, every row of P contains non-negative numbers summing to one. Hence $\|\mathbf{e}^{(k)}\|_1 \leq c\|\mathbf{e}^{(k-1)}\|_1$, which implies $\|\mathbf{e}^{(k)}\|_1 \leq c^k\|\mathbf{e}^{(0)}\|_1$. Since $0 \leq c < 1$, we deduce $\lim_{k\to\infty} \|\mathbf{e}^{(k)}\|_1 = 0$. □

3. Applied Linear Algebra

3.1. **Complex Numbers.** In this subsection, I will assume that you already know that the 2×2 matrix

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

rotates vectors anticlockwise through an angle θ. It is geometrically obvious that $R(\theta_1 + \theta_2) = R(\theta_1)R(\theta_2)$.

Exercise 3.1. *Use the relation $R(\theta_1+\theta_2) = R(\theta_1)R(\theta_2)$ to prove the trigonometric addition formulae*

$$\cos(\theta_1 + \theta_2) = \cos\theta_1 \cos\theta_2 - \sin\theta_1 \sin\theta_2$$

and

$$\sin(\theta_1 + \theta_2) = \sin\theta_1 \cos\theta_2 + \cos\theta_1 \sin\theta_2.$$

Exercise 3.2. *Write down $R(0)$ and $R(\pi/2)$. Interpret them geometrically.*

We shall now consider the set of *rotation–enlargements*. Specifically, a rotation–enlargement is simply a matrix of the form $sR(\theta)$, for some $s, \theta \in \mathbb{R}$. In other words, it's just a multiple of a rotation matrix. If we take any two rotation–enlargements $s_1R(\theta_1)$ and $s_2R(\theta_2)$, then

$$[s_1R(\theta_1)]\,[s_2R(\theta_2)] = s_1s_2R(\theta_1 + \theta_2).$$

Further

$$sR(\theta) = s\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} = \begin{pmatrix} a & -b \\ b & a \end{pmatrix},$$

where $a = s\cos\theta$ and $b = s\sin\theta$. Thus every rotation enlargement can be written as a linear combination of the two special matrices

$$\mathbf{1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad \mathbf{I} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}.$$

Exercise 3.3. *Show that $\mathbf{I}^2 = -\mathbf{1}$ and*

$$(a\mathbf{1} + b\mathbf{I})\,(c\mathbf{1} + d\mathbf{I}) = (ac - bd)\,\mathbf{1} + (ad + bc)\,\mathbf{I}.$$

Historically, we usually write $a + ib$ instead of $a\mathbf{1} + b\mathbf{I}$, so that $i^2 = -1$ and the previous displayed equation becomes

$$(a + ib)\,(c + id) = ac - bd + i\,(ad + bc)\,.$$

Thus real arithmetic corresponds to adding and multiplying enlargement matrices, whilst complex arithmetic augments this to adding and multiplying rotation–enlargement matrices. Multiplication by i is simply rotation (clockwise) through 90 degrees, so $i^2 = -1$ is geometrically obvious. Instead of speaking of rotation-enlargements $a\mathbf{1} + b\mathbf{I}$, we shall now revert to the time-honoured notation $a + ib$ and let $\mathbb{C}$ denote the *complex plane*, the set of all complex numbers.

The *conjugate* z^* of $z = a + ib$ is defined by

$$z^* = a - ib.$$

Exercise 3.4. *Prove that $z = z^*$ if and only if z is real (i.e. $b = 0$). Further, show that $z^*z = |z|^2$, where $|z| = \sqrt{a^2 + b^2}$ for $z = a + ib$.*

Exercise 3.5. *Let $z = a + ib$, $w = c + id$. Show that $z^* w^* = (zw)^*$. Further, let $c_1, \ldots, c_n$ and $\theta_1, \ldots, \theta_n$ be real numbers. Prove that*

$$\left(c_1 e^{i\theta_1} + \cdots + c_n e^{i\theta_n}\right)^* = c_1 e^{-i\theta_1} + \cdots + c_n e^{-i\theta_n}.$$

One of the great triumphs of Eighteenth and Nineteenth century mathematics was a fuller understanding of complex function theory. It turns out that the exponential function can be defined for any complex number by

$$\exp(z) = 1 + z + z^2/2! + z^3/3! + z^4/4! + \cdots,$$

and that this series is convergent for all $z \in \mathbb{C}$. Further, the Taylor series

$$\cos z = 1 - z^2/2! + z^4/4! - z^6/6! + \cdots \quad \text{and} \quad \sin z = z - z^3/3! + z^5/5! - z^7/7! + \cdots$$

for all $z \in \mathbb{C}$. The brilliant Swiss mathematician Euler realized that these series implied the remarkable formula

$$\exp(iz) = \cos z + i \sin z, \qquad z \in \mathbb{C}.$$

If $z = \theta \in \mathbb{R}$, then Euler's formula takes its traditional form

$$e^{i\theta} = \cos\theta + i \sin\theta, \qquad \theta \in \mathbb{R}.$$

Exercise 3.6. *Show that, if $z = e^{i\theta}$ and $\theta \in \mathbb{R}$, then $z^* = e^{-i\theta}$ and $|z| = 1$.*

Exercise 3.7. *Show that $e^{i\pi} = -1$.*

3.2. **Covariance Matrices.** Let $X_1, \ldots, X_n$ be random variables and set $\mu_j = \mathbb{E}X_j$¡ for $1 \leq j \leq n$. Now the corresponding covariance matrix $M \in \mathbb{R}^{n \times n}$, where

$$M_{jk} = \mathbb{E}\left[(X_j - \mu_j)(X_k - \mu_k)\right], \qquad 1 \leq j, k \leq n,$$

should always be non-negative definite. The proof is easy:

$$\begin{aligned} \mathbf{v}^T M \mathbf{v} &= \sum_{j=1}^{n} \sum_{k=1}^{n} v_j v_k M_{jk} \\ &= \mathbb{E}\left[\sum_{j=1}^{n} \sum_{k=1}^{n} v_j v_k (X_j - \mu_j)(X_k - \mu_k) \right] \\ &= \mathbb{E}\left[\left(\sum_{j=1}^{n} v_j (X_j - \mu_j) \right)^2 \right]. \end{aligned} \tag{3.1}$$

Exercise 3.8. *Let*

$$\mathbf{X} = \begin{pmatrix} X_1 \\ \vdots \\ X_n \end{pmatrix}, \boldsymbol{\mu} = \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_n \end{pmatrix}$$

and let $\mathbf{Y} = \mathbf{X} - \boldsymbol{\mu}$. Prove that $M = \mathbb{E}(\mathbf{Y}\mathbf{Y}^T)$, so that

$$\mathbf{v}^T M \mathbf{v} = \mathbf{v}^T \mathbb{E}\left(\mathbf{Y}\mathbf{Y}^T\right) \mathbf{v} = \mathbb{E}\left(\mathbf{v}^T \mathbf{Y}\mathbf{Y}^T \mathbf{v}\right) = \mathbb{E}\left[\left(\mathbf{v}^T \mathbf{Y}\right)^2\right].$$

Thus every real, symmetric, non-negative definite matrix is a covariance matrix. In fact, the converse is also true, so real symmetric, non-negative definite matrices are exactly the same as covariance matrices.

3.3. **Orthogonal Matrices.** Modern numerical linear algebra began with the computer during the Second World War, its progress accelerating enormously as computers became faster and more convenient in the 1960s. One of the most vital conclusions of this research field is the enormous practical importance of matrices which leave Euclidean length invariant. More formally:

Definition 3.1. *We shall say that $Q \in \mathbb{R}^{n\times n}$ is* distance-preserving *if $\|Q\mathbf{x}\| = \|\mathbf{x}\|$, for all $\mathbf{x} \in \mathbb{R}^n$, where the* **Euclidean norm** $\|\mathbf{x}\|$ *is defined by*

$$\|\mathbf{x}\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}, \qquad \text{for } \mathbf{x} \in \mathbb{R}^n.$$

The following simple result is very useful.

Lemma 3.1. *Let $M \in \mathbb{R}^{n\times n}$ be any symmetric matrix for which $\mathbf{x}^T M\mathbf{x} = 0$, for every $\mathbf{x} \in \mathbb{R}^n$. Then M is the zero matrix.*

Proof. Let $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n \in \mathbb{R}^n$ be the usual coordinate vectors. Then

$$M_{jk} = \mathbf{e}_j^T M\mathbf{e}_k = \frac{1}{2}\left(\mathbf{e}_j + \mathbf{e}_k\right)^T M\left(\mathbf{e}_j + \mathbf{e}_k\right) = 0, \qquad 1 \le j, k \le n.$$

□

Theorem 3.2. *The matrix $Q \in \mathbb{R}^n$ is distance-preserving if and only if $Q^TQ = I$.*

Proof. If $Q^TQ = I$, then

$$\|Q\mathbf{x}\|^2 = (Q\mathbf{x})^T(Q\mathbf{x}) = \mathbf{x}^TQ^TQ\mathbf{x} = \mathbf{x}^T\mathbf{x} = \|\mathbf{x}\|^2,$$

and Q is distance-preserving. Conversely, if $\|Q\mathbf{x}\|^2 = \|\mathbf{x}\|^2$, for all $\mathbf{x} \in \mathbb{R}^n$, then

$$\mathbf{x}^T\left(Q^TQ - I\right)\mathbf{x} = 0, \qquad \mathbf{x} \in \mathbb{R}^n.$$

Since $Q^TQ - I$ is a symmetric matrix, Lemma 3.1 implies $Q^TQ - I = 0$, i.e. $Q^TQ = I$. □

The condition $Q^TQ = I$ simply states that the columns of Q are orthonormal vectors, that is, if the columns of Q are $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n$, then $\|\mathbf{q}_1\| = \cdots = \|\mathbf{q}_n\| = 1$ and $\mathbf{q}_j^T\mathbf{q}_k = 0$ when $j \neq k$. For this reason, Q is also called an *orthogonal matrix*. We shall let $O(n)$ denote the set of all (real) $n \times n$ orthogonal matrices.

Exercise 3.9. *Let $Q \in O(n)$. Prove that $Q^{-1} = Q^T$. Further, prove that $O(n)$ is closed under matrix multiplication, that is, $Q_1Q_2 \in O(n)$ when $Q_1, Q_2 \in O(n)$. (In other words, $O(n)$ forms a group under matrix multiplication. This observation is important, and $O(n)$ is often called the* Orthogonal Group*.)*

3.4. **The Singular Value Decomposition.** If $A \in \mathbb{R}^{n\times n}$ is a symmetric matrix, then its spectral structure, that is, its eigenvalues and eigenvectors, are vital for understanding its behaviour in many problems. To compute the spectrum in MATLAB , we use `[Q, D] = eig(A)`. Thus $Q \in \mathbb{R}^{n\times n}$ is the matrix whose columns are eigenvalues of A and D is the diagonal matrix whose elements are the eigenvalues of A. When A is symmetric, the matrix Q is *orthogonal*, that is $Q^TQ = I$, and $A = QDQ^T$.

Exercise 3.10. *Generate a random symmetric matrix A using*

```
M = randn(n);
A = M + M';
```

and use MATLAB *to calculate its spectral decomposition. Test Q for orthogonality.*

If A is *not* symmetric, or if A is rectangular (as is typical in least squares problems), then the *Singular Value Decomposition* (SVD) provides the guidance given by the spectral decomposition for real symmetric matrices.

Theorem 3.3. *Let $A \in \mathbb{R}^{m\times n}$, $m \geq n$. Then there exist orthogonal matrices $U \in O(m)$, $V \in O(n)$, and a diagonal matrix $S \in \mathbb{R}^{m\times n}$, such that $A = USV^T$, and this is called the* **Singular Value Decomposition (SVD)**. *Further, the elements of S satisfy $s_1 \geq s_2 \geq \cdots \geq s_n \geq 0$, and they are called the* **singular values** *of A.*

Proof. **This proof is included for general interest, but is not examinable.** The unit sphere

$$S = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\| = 1\}$$

in $\mathbb{R}^n$ is a closed and bounded set. Since the function $f(\mathbf{x}) = \|A\mathbf{x}\|$, $\mathbf{x} \in \mathbb{R}^n$, is continuous, there exists a unit vector $\mathbf{v}_1 \in \mathbb{R}^n$ for which f attains its maximum value, that is,

$$\|A\mathbf{x}\| \leq \|A\mathbf{v}_1\|, \qquad \text{for } \mathbf{x} \in S.$$

Let $\sigma_1 = \|A\mathbf{v}_1\|$ and $\mathbf{u}_1 = A\mathbf{v}_1/\sigma_1$, so that $\mathbf{u}_1 \in \mathbb{R}^m$ is a unit vector for which $A\mathbf{v}_1 = \sigma_1\mathbf{u}_1$. Using Lemma 3.4, whose proof is based on the next guided exercise, we construct orthogonal matrices $U_1 \in O(m)$, $V_1 \in O(n)$, whose first columns are $\mathbf{u}_1$ and $\mathbf{v}_1$, respectively. Then

$$A_1 := U_1^T A V_1 = \begin{pmatrix} \sigma_1 & \mathbf{w}^T \\ 0 & B \end{pmatrix}.$$

Now let

$$\mathbf{y} = \begin{pmatrix} \sigma_1 \\ \mathbf{w} \end{pmatrix}.$$

Then

$$\|\mathbf{y}\|^2 = \sigma_1^2 + \|\mathbf{w}\|^2$$

and

$$(A_1\mathbf{y})_1 = \sigma_1^2 + \|\mathbf{w}\|^2.$$

Thus

$$\|A_1\mathbf{y}\|^2 \geq (A_1\mathbf{y})_1^2 = \left(\sigma_1^2 + \|\mathbf{w}\|^2\right)^2$$

and

$$\frac{\|A_1\mathbf{y}\|}{\|\mathbf{y}\|} \geq \sqrt{\sigma_1^2 + \|\mathbf{w}\|^2}.$$

Thus $\mathbf{z} := \mathbf{y}/\|\mathbf{y}\|$ is a unit vector in $\mathbb{R}^n$ and

$$\sigma_1 \geq \|A_1\mathbf{z}\| \geq \left(\sigma_1^2 + \|\mathbf{w}\|^2\right)^{1/2},$$

which implies $\mathbf{w} = 0$. Thus

$$A_1 := U_1^T A V_1 = \begin{pmatrix} \sigma_1 & 0 \\ 0 & B \end{pmatrix}.$$

The proof now proceeds inductively, working on the $(n-1)\times(n-1)$ matrix B. □

Exercise 3.11. *Let* $\mathbf{a}$ *and* $\mathbf{b}$ *be different unit vectors in $\mathbb{R}^n$, i.e.* $\mathbf{a} \neq \mathbf{b}$ *and* $\|\mathbf{a}\| = \|\mathbf{b}\| = 1$. *Let*

$$\mathbf{u} = \frac{\mathbf{a} - \mathbf{b}}{\|\mathbf{a} - \mathbf{b}\|}$$

and define $Q = I - 2\mathbf{u}\mathbf{u}^T$. Show that $Q = Q^T$, $Q^TQ = I$ and $Q\mathbf{a} = \mathbf{b}$. [Thus we can always construct an orthogonal matrix mapping one vector onto another. This is important in several algorithms. Geometrically, Q is reflection in the subspace of vectors orthogonal to u.]

Lemma 3.4. *Let $\mathbf{u}_1 \in \mathbb{R}^n$ be any unit vector. Then there is an orthogonal matrix $Q \in O(n)$ whose first column is $\mathbf{u}_1$.*

Proof. We substitute $\mathbf{a} = \mathbf{e}_1$ and $\mathbf{b} = \mathbf{u}_1$ in Exercise 3.11. □

3.5. **Linear Least Squares.** One use of the SVD is to calculate the least squares solution of linear systems. If $A \in \mathbb{R}^{m\times n}$ has the SVD $A = USV^T$, then

$$\|A\mathbf{x} - \mathbf{y}\|^2 = \|USV^T\mathbf{x} - \mathbf{y}\|^2 = \|S\mathbf{a} - \mathbf{b}\|^2,$$

where $\mathbf{a} = V^T\mathbf{x}$ and $\mathbf{b} = U^T\mathbf{y}$. But

$$S\mathbf{a} = \begin{pmatrix} s_1a_1 \\ \vdots \\ s_na_n \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Thus

$$\|A\mathbf{x} - \mathbf{y}\|^2 = \sum_{k=1}^{n} (s_ka_k - b_k)^2 + \sum_{\ell=n+1}^{m} b_\ell^2,$$

and this is minimized by choosing

$$a_k = b_k/s_k, \qquad 1 \le k \le n,$$

the minimum value being $b_{n+1}^2 + \cdots + b_m^2$.

All of this is done automatically by MATLAB : simply type `x = A\ y;`.

Example 3.1. *We want to calculate coefficients $c_1, \ldots, c_n \in \mathbb{R}$ such that the function*

$$s(\mathbf{x}) = \sum_{k=1}^{n} c_k f_k(\mathbf{x}), \qquad \mathbf{x} \in \mathbb{R}^n,$$

minimizes the sum of squares

$$\sum_{\ell=1}^{m} (g_\ell - s(\mathbf{x}_\ell))^2 .$$

In matrix form, we let

$$\mathbf{g} = \begin{pmatrix} g_1 \\ g_2 \\ \vdots \\ g_m \end{pmatrix} \in \mathbb{R}^m$$

and define $A \in \mathbb{R}^{m\times n}$ by

$$A_{jk} = f_k(\mathbf{x}_j), \qquad 1 \le j \le m, \quad 1 \le k \le n.$$

Thus we must minimize $\|\mathbf{g} - A\mathbf{c}\|^2$, whose solution is given by the SVD analysis above.

3.6. **Matrix Nearness Problems.** Certain important matrix nearness problems give rise to several special nonlinear least squares problems. All of these problems provide excellent illustrations of the power of the singular value decomposition and the spectral decomposition.

(1) (**Nearest Covariance Matrix**) Let $\mathbb{P}_n$ denote the set of $n \times n$ non-negative definite symmetric real matrices. Given a symmetric $A \in \mathbb{R}^{n \times n}$, we want to calculate the nearest element of $\mathbb{P}_n$ in the Frobenius norm. For example, a corrupted covariance matrix A might have some small negative eigenvalues, and these can be disastrous if the corrupted matrix is used "as is". Thus we need to calculate M_A minimizing the sum of squares

$$\|A - M_A\|_F^2 = \min_{M \in \mathbb{P}_n} \|A - M\|_F^2.$$

The solution is simple given the spectral decomposition $A = QDQ^T$. Here Q is the orthogonal matrix formed by the eigenvectors of A and the diagonal elements of the diagonal matrix D are the corresponding eigenvalues. We then let $\hat{D} = \max\{D, 0\}$ and set

$$M_A = Q\hat{D}Q^T.$$

In Matlab, we use

```
[Q, D] = eig(A); Dhat = max(D,0)); MA = Q*Dhat*Q';
```

(2) (**Square Procrustes**) Let $O(n)$ denote the set of $n \times n$ real orthogonal matrices. Given $A \in \mathbb{R}^{n \times n}$, we want to compute the nearest orthogonal matrix $Q_A \in O(n)$. For example, the matrix A might contain the principal axes of a computer-controlled flying vehicle, corrupted by measurement error. Thus we need to calculate $Q_A \in O(n)$ minimizing the sum of squares

$$\|A - Q_A\|_F^2 = \min_{Q \in O(n)} \|A - Q\|_F^2.$$

The solution is simple given the singular value decomposition $A = USV^T$, where $U, V \in O(n)$ and S is a diagonal matrix whose diagonal elements (the singular values of A) satisfy

$$s_1 \geq s_2 \geq \cdots \geq s_n \geq 0.$$

As shown in lectures, $Q_A = UV^T$. In Matlab, we use

```
[U, S, V] = svd(A); QA = U*V';
```

(3) (**Rectangular Procrustes**) Let $m \geq n$ and let $A, B \in \mathbb{R}^{m \times n}$ be given matrices. We want to compute the orthogonal matrix $\hat{Q}$ minimizing $\|A - BQ\|_F$. For example, the rows of A and B might be coordinates of points with respect to two orthogonal coordinate systems, such as the computer systems controlling a ship and one of its missiles after launch.

More formally, we need to calculate $Q \in O(n)$ minimizing the sum of squares

$$\|A - BQ\|_F^2 = \min_{U \in O(n)} \|A - BU\|_F^2.$$

As shown in lectures, we first calculate the SVD $B^T A = USV^T$, then simply set $Q = UV^T$. In Matlab,

```
[U, S, V] = svd(B'*A);
Q = U*V';
```

(4) (**Closest matrix of given rank**) Given any matrix $A \in \mathbb{R}^{m\times n}$, for $m \geq n$, the *rank* of A is simply the number of nonzero singular values. It's also equal to the number of linearly independent columns (and also equal to the number of linearly independent rows). However, in practice, we are often faced with a matrix A with *no* singular values exactly equal to zero. Instead, we find matrices whose first r singular values, say, are much larger than the remaining singular values. In this case, it is often useful to regard the tiny singular values as corresponding to noise. Thus we want to find the nearest matrix of rank r in the Frobenius norm; more formally, A_r must satisfy

$$\|A - A_r\|_F^2 = \min_{\text{rank } B=r} \|A - B\|_F^2.$$

The solution is, once again, very simple given the SVD $A = USV^T$. We simply set

$$S_r = \text{diag}\,\{s_1, \ldots, s_r, 0, \ldots, 0\}.$$

In other words, we just set all singular values to zero except for the r largest singular values. As shown in lectures, $A_r = US_rV^T$. You have seen this technique in Principal Component Analysis, for which A is symmetric (and non-negative definite), in which case the SVD is precisely the spectral decomposition.

Exercise 3.12. *Write* MATLAB *code to generate the closest matrix of rank r.*

Our study of the SVD will make use of the following so-called Frobenius inner product on matrices. Just think of an $m \times n$ matrix as a long vector in $\mathbb{R}^{mn}$ and use the usual inner product there.

Definition 3.2. *Given any two matrices $A, B \in \mathbb{R}^{m\times n}$, their* Frobenius inner product *is defined by the equation*

$$\langle A, B\rangle_F = \sum_{k=1}^{m}\sum_{\ell=1}^{n} A_{k\ell}B_{k\ell}.$$

The Frobenius norm *of A is defined by*

$$\|A\|_F = \sqrt{\langle A, A\rangle_F} = \sqrt{\sum_{k=1}^{m}\sum_{\ell=1}^{n} A_{k\ell}^2}.$$

Lemma 3.5. *Let $A \in \mathbb{R}^{n\times n}$ and let $Q \in O(n)$. Then*

$$\|A\|_F = \|QA\|_F = \|AQ\|_F.$$

Proof. Let $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$ be the columns of A. Then QA is the matrix with columns $Q\mathbf{a}_1, Q\mathbf{a}_2, \ldots, Q\mathbf{a}_n$. Since Q is an orthogonal matrix, $\|Q\mathbf{a}_j\|^2 = \|\mathbf{a}_j\|^2$, for $1 \leq j \leq n$, and

$$\|QA\|_F^2 = \|Q\mathbf{a}_1\|^2 + \cdots + \|Q\mathbf{a}_n\|^2 = \|\mathbf{a}_1\|^2 + \cdots + \|\mathbf{a}_n\|^2 = \|A\|_F^2.$$

□

Lemma 3.6. *Let $A, B \in \mathbb{R}^{m\times n}$ and let $U \in O(m)$, $V \in O(n)$. Then $\langle A, B\rangle_F = \langle UA, UB\rangle_F = \langle AV, BV\rangle_F$.*

Proof. This is almost identical to the proof of the previous lemma, and is left as an exercise. □

Lemma 3.7. *Let $A, B \in \mathbb{R}^{m\times n}$ and let $C \in \mathbb{R}^{n\times n}$. Then*

$$\langle A, BC\rangle_F = \langle B^T A, C\rangle_F.$$

Proof. Let $\mathbf{a}_1, \ldots, \mathbf{a}_n$ be the columns of A and let $\mathbf{b}_1, \ldots, \mathbf{b}_n$ be the columns of B. Then $B\mathbf{c}_1, \ldots, B\mathbf{c}_n$ form the columns of BC and we obtain

$$\begin{aligned}\langle A, BC\rangle_F &= \mathbf{a}_1^T(B\mathbf{c}_1) + \cdots \mathbf{a}_n^T(B\mathbf{c}_n)\\ &= (B^T\mathbf{a}_1)^T\mathbf{c}_1 + \cdots + (B^T\mathbf{a}_n)^T\mathbf{c}_n\\ &= \langle B^T A, C\rangle_F,\end{aligned}$$

because $B^T\mathbf{a}_1, \ldots, B^T\mathbf{a}_n$ form the columns of $B^T A$. □

Lemma 3.7 can also be proved using a highly useful alternative formula for the Frobenius inner product of two matrices. We recall that the **trace** of a matrix is the sum of its diagonal elements, that is,

$$\operatorname{trace} A = \sum_{j=1}^{n} A_{jj},$$

for $A \in \mathbb{R}^{n\times n}$.

Example 3.2. *Of course, $AB \neq BA$ in general, but* $\operatorname{trace} AB = \operatorname{trace} BA$*, as the following direct calculation show. We have*

$$\operatorname{trace} AB = \sum_{j=1}^{n}\sum_{k=1}^{n} A_{jk}B_{kj}$$

and

$$\operatorname{trace} BA = \sum_{\ell=1}^{n}\sum_{m=1}^{n} B_{\ell m}A_{m\ell}$$

and these double sums are identical.

Here is the alternative expression.

Theorem 3.8. *Let $A, B \in \mathbb{R}^{m\times n}$. Then*

$$\langle A, B\rangle_F = \operatorname{trace}\left(AB^T\right)$$

Proof. We have

$$\begin{aligned}\operatorname{trace} AB^T &= \sum_{j=1}^{n}\left(AB^T\right)_{jj}\\ &= \sum_{j=1}^{n}\sum_{k=1}^{n} A_{jk}\left(B^T\right)_{kj}\\ &= \sum_{j=1}^{n}\sum_{k=1}^{n} A_{jk}B_{jk}\\ &= \langle A, B\rangle_F.\end{aligned}$$

□

Example 3.3. *We can use Theorem 3.8 to give a slick proof of Lemma 3.7. We have*

$$\langle A, BC\rangle_F = \operatorname{trace} A\,(BC)^T = \operatorname{trace} AC^TB^T = \operatorname{trace} B^TAC^T = \langle B^TA, C\rangle_F.$$

Our final theorem justifies the procedures given above for both the rectangular and square Procrustes' problems.

Theorem 3.9. *Let $A, B \in \mathbb{R}^{m\times n}$, where $m \geq n$, and let $B^TA = USV^T$ be the SVD of B^TA. Then the orthogonal matrix solving the rectangular Procrustes problem*

$$\min_{Q\in O(n)} \|A - BQ\|_F$$

is given by $\widehat{Q} = UV^T$.

Proof. Lemma 3.5 and Lemma 3.7 imply that

$$\begin{aligned}\|A - BQ\|_F^2 &= \langle A - BQ, A - BQ\rangle_F \\ &= \|A\|_F^2 + \|BQ\|_F^2 - 2\langle A, BQ\rangle_F \\ &= \|A\|_F^2 + \|B\|_F^2 - 2\langle B^TA, Q\rangle_F.\end{aligned}$$

Thus minimizing $\|A-BQ\|_F^2$ is equivalent to maximizing $\langle B^TA, Q\rangle_F$, for $Q \in O(n)$. To this end, substituting the SVD of B^TA and applying Lemma 3.6, we obtain

$$\begin{aligned}\max_{Q\in O(n)} \langle B^TA, Q\rangle_F &= \max_{Q\in O(n)} \langle USV^T, Q\rangle_F \\ &= \max_{Q\in O(n)} \langle S, U^TQV\rangle_F \\ &= \max_{W\in O(n)} \langle S, W\rangle_F \\ &= \max_{W\in O(n)} \sum_{k=1}^n s_kW_{kk},\end{aligned}$$

where $s_1 \geq s_2 \geq \cdots \geq s_n \geq 0$ are the singular values of B^TA. Now W is an orthogonal matrix, so its columns all have Euclidean norm one. Hence $|W_{kk}| \leq 1$, for all k, with equality if and only if $W = I$. Thus

$$\sum_{k=1}^n s_kW_{kk} \leq \left|\sum_{k=1}^n s_kW_{kk}\right| \leq \sum_{k=1}^n s_k|W_{kk}| \leq \sum_{k=1}^n s_k,$$

with equality if and only if $W = I$, that is, $Q = UV^T$. Hence $\|A - BQ\|_F^2$ is minimized when $\widehat{Q} = UV^T$. □

The mathematics of Procrustes' problem leads to the so-called *Polar Factorization*. The name "polar factorization" refers to the analogy with complex numbers: any $z \in \mathbb{C}$ can be written as $z = re^{i\theta}$, and this is said to be its polar factorization.

Exercise 3.13. *Let $A \in \mathbb{R}^{n\times n}$. Then there exists an orthogonal matrix $Q \in O(n)$, and a symmetric non-negative definite matrix P such that $A = PQ$. [Hint: Rewrite the SVD $A = USV^T$ as $A = (USU^T)(UV^T)$.]*

4. Alternative Procrustes

This was one of the questions set on assigned work last year. The singular value decomposition is not always the quickest way to solve the rectangular Procrustes problems, particular when the ambient dimension n is small and A is already close to an orthogonal matrix. Here's an alternative.

Several applications, such as robotics and aircraft control systems, present the following problem: the system attempts to maintain an orthogonal matrix $Q(t)$ that describes its orientation at each time t (the columns of the matrix are orthonormal vectors fixed in the system). Unfortunately, measurement errors occur which cause the measured $Q(t)$ to lose orthogonality, i.e. we no longer have $Q(t)^T Q(t) = I$. Therefore there is a need to calculate an orthogonal matrix $\hat{Q}(t)$ that is closest to the observed matrix $Q(t)$ in some sense. If we decide to choose $\hat{Q}(t)$ to minimize the Frobenius norm $\|Q(t) - \hat{Q}(t)\|_F$, then there is a clever algorithm for calculating $\hat{Q}(t)$: we choose $W_0 = Q(t)$ and then set

$$W_{\ell+1} = \frac{1}{2}\Big(W_\ell + (W_\ell^{-1})^T\Big), \qquad \ell \geq 0.$$

It can be shown that $\|W_\ell - \hat{Q}(t)\|_F \to 0$ as $\ell \to \infty$, and you will see that convergence is fast (in many cases 5 steps will be enough).

Exercise 4.1. *Write a short* MATLAB *script to generate the matrices of this iteration and investigate the speed of convergence (for 3×3 matrices) by plotting $\log \|\hat{Q}(t) - W_\ell\|_F$. You will need to generate random orthogonal matrices, for which the following* MATLAB *code is suitable.*

```
A = randn(3); [Q, R] = qr(A);
```

You can assume that Q is a suitable random orthogonal matrix. You can then slightly perturb Q by setting

```
W = Q + delta*randn(3);
```

Of course, a large `delta` *is a large perturbation. I suggest starting with* `delta = 0.1`*, but try larger values also. Does the algorithm ever fail? Generate some histograms displaying the average behaviour for fixed delta and many random initial perturbed orthogonal matrices. [You can enhance your Christmas by discovering Procrustes' sadistic practices via Google.]*

In fact, the SVD is the key to proving that this algorithm converges.

Exercise 4.2. *Given the SVD $W_k = U_k S_k V_k^T$, show that*

$$\left(W_k^T\right)^{-1} = U_k S_k^{-1} V_k^T.$$

Thus

$$W_{k+1} = \frac{1}{2}\left(W_k + \left(W_k^{-T}\right)\right) = U_k S_{k+1} V_k^T,$$

where

$$S_{k+1} = \frac{1}{2}\left(S_k + S_k^{-1}\right).$$

Thus the orthogonal factors of the SVD $W_{k+1} = U_{k+1} S_{k+1} V_{k+1}$ are the same as the orthogonal factors of W_k – only the singular values change. This is true at every iteration. Thus $W_0 = U_0 S_0 V_0^T$ and

$$W_k = U_0 S_k V_0^T,$$

where

$$S_{k+1} = \frac{1}{2}\left(S_k + S_k^{-1}\right).$$

Exercise 4.3. *Let $s_1^{(k)} \geq \cdots \geq s_n^{(k)} \geq 0$ be the diagonal elements of S_k. Prove that*

$$s_i^{(k+1)} = \frac{1}{2}\left(s_i^{(k)} + \frac{1}{s_i^{(k)}}\right), \qquad 1 \leq i \leq n.$$

Thus everything reduces to the study of the iteration

$$x_{k+1} = \frac{1}{2}\left(x_k + \frac{1}{x_k}\right), \qquad k \geq 0,$$

when x_0 can be any positive number. It is not hard to prove that $x_k \to 1$, for any $x_0 > 0$. Thus the singular value matrix S_k of W_k converges to the identity matrix, which implies $W_k \to U_0 V_0^T$. Further it is not hard to prove that the algorithm converges *quadratically*: given any initial point $x_0 > 0$, there is a constant C and an integer N for which

$$|1 - x_{k+1}| \leq C\,(1 - x_k)^2, \qquad \text{for } k \geq N.$$

Thus we ultimately obtain a sequence of errors of the typical form

$$10^{-1}, 10^{-2}, 10^{-4}, 10^{-8}, 10^{-16},$$

at which point computer arithmetic reaches its limit.

5. Generating Random Matrices

As you know, random number generation is a vital part of modern statistics, being required for all simulation work. Our purpose here is to construct certain *random matrices*. We shall apply these random matrices to test the matrix nearness problems given in the previous section.

5.1. **Gaussian Random Vectors and Matrices.** Let $\{Z_{jk} : 1 \leq j, k \leq n\}$ be n^2 independent $N(0,1)$ Gaussian random variables. Then the matrix Z formed by these components is called a (normalized) Gaussian random matrix. In MATLAB , we simply type `Z = randn(n)`.

It is sometimes important to generate symmetric Gaussian random matrices. Given a general Gaussian random matrix Z, we simply set `W = (Z + Z')/2`.

A (normalized) Gaussian random vector $\mathbf{X}$ simply a vector whose components are independent $N(0,1)$ random variables. In other words, it's just a Gaussian random matrix that is $n \times 1$. In MATLAB , we type `X = randn(n,1);`

5.2. **Uniformly distributed points on the Sphere.** The following problem is very common in *geostatistics*: generate unit vectors $u_1, \ldots, u_N \in \mathbb{R}^3$ that are uniformly distributed on the sphere. The solution is very simple and works in n-dimensions: we generate a normalized Gaussian random vector and divide by its Euclidean norm. In MATLAB :

```
X = randn(n,1);
Y = X/norm(X);
```

Let us now state this more formally.

Theorem 5.1. *Let $\mathbf{X} \in \mathbb{R}^n$ be a normalized Gaussian random vector. The $\mathbf{Y} := \mathbf{X}/\|\mathbf{X}\|$ is uniformly distributed on the unit sphere S in n-dimensions.*

Proof. Let U be any subset[1] of the unit sphere S. Then $\mathbf{Y} \in U$ if and only if $X \in K(U)$, where

$$K(U) = \{r\mathbf{u} : r \geq 0 \text{ and } \mathbf{u} \in U\}.$$

Hence, if

$$p(\mathbf{z}) = (2\pi)^{-n/2} e^{-\|\mathbf{z}\|^2/2}, \qquad \mathbf{z} \in \mathbb{R}^n,$$

denotes the probability density function for the Gaussian, then

$$\begin{aligned}
\mathbb{P}(\mathbf{Y} \in U) &= \mathbb{P}(\mathbf{X} \in K(U)) \\
&= \int_{K(U)} p(\mathbf{z})\, d\mathbf{z} \\
&= (2\pi)^{-n/2} \left(\int_0^\infty e^{-r^2/2} r^{n-1}\, dr \right) \mathrm{vol}_{n-1}(U) \qquad (5.1) \\
&= I_{n-1}\, \mathrm{vol}_{n-1}(U). \qquad (5.2)
\end{aligned}$$

If $U = S$, then the previous equation becomes $1 = I_{n-1}\, \mathrm{vol}_{n-1}(S)$, so that

$$\mathbb{P}(\mathbf{Y} \in U) = \frac{\mathrm{vol}_{n-1}(U)}{\mathrm{vol}_{n-1}(S)},$$

and $\mathbf{Y}$ is uniformly distributed on the sphere. □

[1] If you've taken a course in measure theory, then this should be a Lebesgue measurable subset of S.

Exercise 5.1. *Write out the previous proof in the special case when $n = 2$, to ensure you understand it.*

Exercise 5.2. *Use* MATLAB *to generate uniformly distributed points on the unit circle in two dimensions using the construction above. How do we check that they are uniformly distributed?*

Example 5.1. *The last exercise, and the* MATLAB *session, point towards a χ^2 goodness-of-fit test. In fact, the χ^2 distribution is only a slight modification of the proof given above, and we can now derive its probability density distribution, which you have probably used for years. Specifically, if $\mathbf{X} \in \mathbb{R}^n$ is a Gaussian random vector, the we say that the random variable $\|\mathbf{X}\|$ has the χ^2 distribution with n degrees of freedom. Now, if we let*

$$\mathrm{ann}(c, d) = \{\mathbf{x} \in \mathbb{R}^n : c < \|\mathbf{x}\| < d\},$$

that is, the annulus with inner radius c and outer radius d, then

$$\begin{aligned}
\mathbb{P}\left(a < \|\mathbf{X}\|^2 < b\right) &= \mathbb{P}\left(\mathbf{X} \in \mathrm{ann}(a^{1/2}, b^{1/2})\right) \\
&= \int_{\mathrm{ann}(a^{1/2}, b^{1/2})} e^{-\|\mathbf{z}\|^2/2} (2\pi)^{-n/2}\, d\mathbf{z} \\
&= \omega_{n-1}(2\pi)^{-n/2} \int_{a^{1/2}}^{b^{1/2}} e^{-r^2/2} r^{n-1}\, dr \\
&= \frac{1}{2}\omega_{n-1}(2\pi)^{-n/2} \int_a^b e^{-s/2} s^{n/2-1}\, ds,
\end{aligned}$$

where ω_{n-1} is the $(n-1)$-dimensional volume of the unit sphere and I substituted $s = r^2$ to obtain the last integral. Thus the probability density function of the χ^2 distribution, with n degrees of freedom, is given by

$$p_n(s) = \frac{1}{2}\omega_{n-1}(2\pi)^{-n/2} e^{-s/2} s^{n/2-1}, \qquad s \geq 0.$$

If we set $a = 0$ and $b = \infty$, then

$$\begin{aligned}
1 &= \frac{1}{2}\omega_{n-1}(2\pi)^{-n/2} \int_0^\infty e^{-s/2} s^{n/2-1}\, ds \\
&= \frac{1}{2}\omega_{n-1}(2\pi)^{-n/2} 2^{n/2} \Gamma(n/2),
\end{aligned}$$

using the fact that the Gamma function is defined by the integral

$$\Gamma(z) = \int_0^\infty e^{-t} t^{z-1}\, dt,$$

for any complex number z with non-negative real part. Thus

$$p_n(s) = \frac{1}{2^{n/2}\Gamma(n/2)} e^{-s/2} s^{n/2-1}, \qquad s \geq 0.$$

As a byproduct of this analysis, we obtain $\omega_{n-1} = 2\pi^{n/2}/\Gamma(n/2)$.

Exercise 5.3. *Use the technique of the previous example to find the probability density function for $\|\mathbf{X}\|$ when $\mathbf{X} \in \mathbb{R}^2$ is a two-dimensional Gaussian random vector.*

5.3. **Random Orthogonal Matrices.** The mathematical details of what precisely is meant by a random orthogonal matrix are beyond the scope of this course. To generate them, we shall use the QR-factorizationg: given any matrix $A \in \mathbb{R}^{n \times n}$, there exists an orthogonal matrix $Q \in O(n)$, and an upper triangular matrix $R \in \mathbb{R}^{n \times n}$, such that $A = QR$. The MATLAB command for calculating this important factorization is simply `[Q, R] = qr(A)`.

To generate a random orthogonal matrix Q, we first generate a (normalized) Gaussian random matrix Z, and then let Q be the orthogonal factor in its QR-factorization, that is

```
Z = randn(n);
[Q, R] = qr(Z);
```

The QR-factorization is in fact the Gram–Schmidt algorithm in disguise, which you might well have seen as undergraduates..

6. Clustering Algorithms

Given points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$, we often want to classify them into *clusters*. For example, suppose the point are the first d characters of n email messages, containing spam and *bona fide* emails. Many spam filter programs are based on the hope that these emails will fall into two well-defined clusters, spam and non-spam. Some filters look for k clusters, reflecting the fact that spam often falls into fairly well-defined categories. But what do we mean by clusters in d-dimensions?

6.1. **The k-means Clustering Algorithm.** Let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ be points in $\mathbb{R}^d$. We shall study a simple algorithm for iteratively updating a set of k *cluster centres* $\mathbf{m}_1, \ldots, \mathbf{m}_k$. At the start of the algorithm, these points can be any vectors; in my MATLAB code below, I've chosen them randomly.

Now the k cluster centres partition $\mathbb{R}^d$ into k clusters: we let the ith cluster C_i be those points in $\mathbb{R}^d$ for which $\mathbf{m}_i$ is the closest cluster centre, that is

$$C_i = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x} - \mathbf{m}_i\| = \min_{1 \le \ell \le k} \|\mathbf{x} - \mathbf{m}_\ell\|\}, \qquad 1 \le i \le n.$$

We then replace each cluster centre $\mathbf{m}_i$ by the centroid of the subset of points in $x_1, \ldots, x_n$ which are contained in the ith-cluster (the centroid of a finite set of points $\mathbf{v}_1, \ldots, \mathbf{v}_j$ is simply the sample average $(\mathbf{v}_1 + \cdots + \mathbf{v}_j)/j$). The new cluster centres then define corresponding new centres, and we then repeat the procedure until the cluster centres converge.

Exercise 6.1. *Let $k = 2$ and give a sketch illustrating C_1 and C_2.*

In fact, the above algorithm is minimizing yet another nonlinear least squares problem. We want to choose centres $\hat{\mathbf{m}}_1, \ldots, \hat{\mathbf{m}}_k \in \mathbb{R}^d$ minimizing

$$F(\mathbf{m}_1, \ldots, \mathbf{m}_k) = \sum_{j=1}^{n} \min_{1 \le \ell \le k} \|\mathbf{x}_j - \mathbf{m}_\ell\|^2.$$

The algorithm below performs the iteration described above.

```
%
% One iteration of the k-means algorithm
%
% INPUT:
%   d = dimension
%   n = number of points
%   k = number of clusters
%   x = d x n array of points
%   m = d x k array of trial centroids
%
% OUTPUT:
%   m = new d x k array of trial centroids
%
C = zeros(d,k); csize = zeros(k);

for i=1:n
    %
    % Find the nearest cluster centre to the current point
    %
```

```
    mindist = norm(x(:,i) - m(:,1));
    cluster_index = 1;
    for j=2:k
        dist = norm(x(:,i) - m(:,j));
        if dist < mindist
            mindist = dist;
            cluster_index = j;
        end
    end
    %
    % cluster_index is the index of the closest trial centroid
    %
    C(:,cluster_index) = C(:,cluster_index) + x(:, i);
    csize(cluster_index) = csize(cluster_index) + 1;
end

for j=1:k
    m(:,j) = C(:,j)/csize(j);
end
```

How can we test this? One way to is to first choose $k = d = 2$ and generate data that lie in two fairly well-defined clusters. For example:

```
N=100; x = randn(2,2*N); x(:,N+1:2*N)=[4; 5]*ones(1,N);
```

7. Radial Basis Functions

You will have seen in the MATLAB classes that polynomials are really quite unsuitable for practical approximation. The modern alternative is to use radial basis functions (RBFs), which we describe via an example.

Example 7.1. *Suppose a bank wants to credit score a large set of n customers. Each customer is specified by a d-dimensional "credit vector" $\mathbf{x}_i$, which might contain their annual salary, the value of any property owned, their current debt, their age, their number of dependents, etc. Suppose there is a much smaller subset of m points, $\mathbf{y}_1, \ldots, \mathbf{y}_m$ say, to which the company has assigned the credit scores $f_1, \ldots, f_m$. These credit scores might be calculated on the basis of the customers' detailed financial history, which might only be available for this smaller "training set". We want a function which learns from these credit scores and enables us to calculate the score for any one of our much larger set of n customers, for most of whom we only know their credit vector. Thus we need a function $s : \mathbb{R}^d \to \mathbb{R}$ such that*

$$s(\mathbf{y}_i) = f(\mathbf{y}_i), \qquad \text{for } 1 \le i \le m.$$

We say that s interpolates the data *(or that s is an interpolant). If the interpolant is inexpensive to evaluate, then we can assign a credit score very easily to our entire database of n customers. We might even use s to allow credit scoring to take place online.*

A radial basis function has the following general form:

$$s(\mathbf{x}) = \sum_{k=1}^{n} a_k \phi(\|\mathbf{x} - \mathbf{b}_k\|), \qquad \mathbf{x} \in \mathbb{R}^d, \tag{7.1}$$

where $\|\mathbf{x}\|$ is simply the Euclidean norm in $\mathbb{R}^d$, that is,

$$\|\mathbf{x}\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_d^2}, \text{ for } \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} \in \mathbb{R}^d. \tag{7.2}$$

We shall be using three norms in these notes: the symbol $\|\mathbf{x}\|$ will always denote the Euclidean norm of a vector $\mathbf{x}$, as defined by (7.2); the infinity norm $\|\mathbf{x}\|_\infty$ is used in Section 7; and the Frobenius norm $\|A\|_F$ for a matrix A is described in Section 3.2.

Theorem 7.1. *The following choices of $\phi : [0, \infty) \to \mathbb{R}$ are suitable for RBF interpolation.*

- *(the Gaussian) $\phi(r) = e^{-cr^2}$;*
- *(the multiquadric) $\phi(r) = (r^2 + c^2)^{1/2}$;*
- *(the inverse multiquadric) $\phi(r) = (r^2 + c^2)^{-1/2}$;*
- *(the Euclidean norm) $\phi(r) = r$ (we need $n > 1$ for this RBF).*

In these examples c is a positive constant.

Proof. The suitability of the Gaussian is given in Theorems 7.5 and 7.6. The suitability of the inverse multiquadric comprises 7.5. The proofs for the multiquadric and the Euclidean norm are not examinable. □

The numbers $a_1, \ldots, a_n$ in (7.1) are usually called the *coefficients*. The points $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^d$ are called *centres* (and sometimes *nodes*). We shall assume that the centres are distinct.

To interpolate function values $f_1, \ldots, f_n \in \mathbb{R}$ at the centres $\mathbf{b}_1, \ldots, \mathbf{b}_n$, we must be able to solve the linear equations

$$s(\mathbf{b}_i) = f_i, \qquad 1 \leq i \leq n, \tag{7.3}$$

that is,

$$\sum_{j=1}^{n} a_j \phi(\|\mathbf{b}_i - \mathbf{b}_j\|) = f_i, \qquad 1 \leq i \leq n. \tag{7.4}$$

In matrix form, we must solve the linear system

$$A\mathbf{a} = \mathbf{f}, \tag{7.5}$$

where the *interpolation matrix* $A \in \mathbb{R}^{n \times n}$ is given by

$$A_{ij} = \phi(\|\mathbf{b}_i - \mathbf{b}_j\|), \qquad 1 \leq i, j \leq n, \tag{7.6}$$

and

$$\mathbf{a} = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix}, \quad \mathbf{f} = \begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix}. \tag{7.7}$$

It is not obvious that the interpolation matrix A is invertible, and the possibility that it might be singular impeded the adoption of RBF approximations until Charles Micchelli, building on much earlier work of Schoenberg, established the non-singularity of interpolation matrices for suitable ϕ in a brilliant paper published in 1985. Other potential users were equally sceptical that RBF approximations were suitable for data fitting, but this was dispelled by an extensive survey of multivariate approximation methods in the early 1980s, undertaken by Richard Franke at the US Bureau of Naval Research. This began an explosion of theoretical and empirical study of RBFs, which continues to this day. We shall be studying the proof of invertibility of the distance matrix in some detail, because the mathematics required is beautiful and very useful in some of the newer applications of RBFs, such as classification methods. Micchelli's proof relies on use of characteristic functions, which therefore form our next object of study.

Let us summarize the use of RBFs to emphasize their simplicity. Given any number of points n in $\mathbb{R}^d$, for any dimension d (with the proviso that $n > 1$ if we're using the Euclidean norm for ϕ), we first choose one of the functions ϕ given in Theorem 7.1. We then form the interpolation matrix (7.6) and solve (7.5). The resulting function $s(\mathbf{x})$, given by (7.1), is the required interpolant.

7.1. **Characteristic Functions.** Let X be any real-valued random variable. Its characteristic function is defined[2] by the expectation

$$\phi_X(z) = \mathbb{E}e^{izX}, \qquad z \in \mathbb{R}.$$

Here[3] $i = \sqrt{-1}$

[2]Strictly speaking, we should also ensure that this expectation is well-defined by stipulating that $\mathbb{E}|X|$ is finite.

[3]I have provided a brief review of required complex arithmetic in the section Linear Algebra.

Why would anyone create and study characteristic functions? They were created by the great French probabilist Paul Lévy in the 1930s in order to prove the Central Limit Theorem, but they turn out to be enormously useful. First observe that, if X and Y are independent random variable, then

$$\phi_{X+Y}(z) = \mathbb{E}e^{iz(X+Y)} = \mathbb{E}e^{izX}\mathbb{E}e^{izY} = \phi_X(z)\phi_Y(z).$$

In other words, the characteristic function of a sum of independent random variables is the product of their characteristic functions. Thus, given any sequence of independent random variables $X_1, X_2, \ldots, X_n$, we have

$$\phi_{X_1+\cdots+X_n}(z) = \phi_{X_1}(z)\phi_{X_2}(z)\cdots\phi_{X_n}(z).$$

What happens if we scale a random variable X to form cX? Then

$$\phi_{cX}(z) = \mathbb{E}e^{iczX} = \phi_X(cz).$$

Let's calculate some characteristic functions. If X has a continuous probability density function $p(t)$, then its characteristic function is given by

$$\phi_X(z) = \int_{-\infty}^{\infty} p(t)e^{izt}\,dt, \qquad z \in \mathbb{R}. \tag{7.8}$$

Lévy was not the first mathematician to consider integrals of this form. In fact (7.8) is an example of a *Fourier transform*, an enormously important discovery of the great French mathematician *Jean Babtiste Fourier*.

Exercise 7.1. *Let X be uniformly distributed in the interval $[-1/2, 1/2]$. Then*

$$\begin{aligned}\phi_X(z) &= \mathbb{E}e^{izX} \\ &= \int_{-1/2}^{1/2} e^{izt}\,dt \\ &= \int_{-1/2}^{1/2} \cos(zt)\,dt + i\int_{-1/2}^{1/2} \sin(zt)\,dt.\end{aligned}$$

Show that

$$\phi_X(z) = \frac{\sin(z/2)}{z/2}, \qquad z \in \mathbb{R}.$$

Characteristic functions are equally important when the random variable $\mathbf{X} \in \mathbb{R}^d$ is a random vector. The definition is essentially unchanged:

$$\phi(\mathbf{z}) = \mathbb{E}e^{i\mathbf{z}^T\mathbf{X}}, \qquad \mathbf{z} \in \mathbb{R}^d. \tag{7.9}$$

If the random vector $\mathbf{X}$ has a continuous probability density function $p(\mathbf{t})$, for $\mathbf{t} \in \mathbb{R}^d$, then

$$\phi(\mathbf{z}) = \int_{\mathbb{R}^d} p(\mathbf{t})e^{i\mathbf{z}^T\mathbf{t}}\,dt, \qquad \mathbf{z} \in \mathbb{R}^d. \tag{7.10}$$

In fact, if the characteristic function $\phi(\mathbf{z})$, $\mathbf{z} \in \mathbb{R}^d$, of a continuous probability density function $p(\mathbf{t})$, $\mathbf{t} \in \mathbb{R}^d$, is *absolutely integrable*, which simply means that the integral

$$\int_{\mathbb{R}^d} |\phi(\mathbf{z})|\,d\mathbf{z}$$

is finite, then we can recover $p(\mathbf{t})$ using the integral

$$p(\mathbf{t}) = (2\pi)^{-d} \int_{\mathbb{R}^d} \phi(\mathbf{z}) e^{-i\mathbf{t}^T\mathbf{z}}\, d\mathbf{z}, \qquad \mathbf{t} \in \mathbb{R}^d.$$

The Gaussian is extremely privileged: the characteristic function of a Gaussian is another Gaussian. Let's state this formally, although its proof is not examinable.

Theorem 7.2. *The probability density function for the multivariate Gaussian $N(0, \sigma^2)$, that is,*

$$p(\mathbf{t}) = (2\pi/\sigma^2)^{-d/2} e^{-\|\mathbf{t}\|^2/2\sigma^2}, \qquad \mathbf{t} \in \mathbb{R}^d,$$

has characteristic function

$$\phi(z) = e^{-\sigma^2\|\mathbf{z}\|^2/2}, \qquad z \in \mathbb{R}^d.$$

Proof. This is beyond the scope of this course. □

The normalization factor required to make the Gaussian a probability density function makes the previous theorem unnecessarily complicated. It's much easier to use the standard notation of the Fourier transform, summarized in the following theorem.

Theorem 7.3. *Let $f(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^d$, be a continuous, absolutely integrable function. Its Fourier transform $\widehat{f}(\mathbf{z})$, $\mathbf{z} \in \mathbb{R}^d$, is given by*

$$\widehat{f}(\mathbf{z}) = \int_{\mathbb{R}^d} f(\mathbf{x}) e^{i\mathbf{z}^T\mathbf{x}}\, d\mathbf{x}, \qquad \mathbf{z} \in \mathbb{R}^d.$$

Further, if $\widehat{f}(\mathbf{z})$ is an absolutely integrable function, then $f(\mathbf{x})$ is given by the inverse Fourier transform *integral*

$$f(\mathbf{x}) = (2\pi)^{-d} \int_{\mathbb{R}^d} \widehat{f}(\mathbf{z}) e^{-i\mathbf{z}^T\mathbf{x}}\, d\mathbf{z}, \qquad \mathbf{x} \in \mathbb{R}^d.$$

Proof. This is beyond the scope of the course. □

Theorem 7.4. *If λ is a positive constant and*

$$G_\lambda(\mathbf{x}) = e^{-\lambda\|\mathbf{x}\|^2}, \qquad \mathbf{x} \in \mathbb{R}^d,$$

then its Fourier transform is given by

$$\widehat{G_\lambda}(\mathbf{z}) = (\pi/\lambda)^{d/2} \exp\left(\frac{-\|\mathbf{z}\|^2}{4\lambda}\right), \qquad \mathbf{z} \in \mathbb{R}^d.$$

Further,

$$G(\mathbf{x}) = (2\pi)^{-d} \int_{\mathbb{R}^d} \widehat{G_\lambda}(\mathbf{z}) e^{i\mathbf{z}^T\mathbf{x}}\, d\mathbf{z}, \qquad \mathbf{x} \in \mathbb{R}^d.$$

Proof. This is beyond the scope of the course. □

Exercise 7.2. *Let $d = 1$ and use* MATLAB *to draw $p(t)$ and $\phi(z)$ for small and large values of σ. What do you see?*

Remark 7.1. *As mentioned above, Lévy used characteristic functions to prove the Central Limit Theorem. I shall sketch his proof here for general interest, since it is one of the great theorems of Probability Theory.* **This is not examinable.**

Now suppose that $X_1, \ldots, X_n$ are independent, identically distributed random variables with mean zero and unit variance, and let $\phi(z)$ be their common characteristic function. Then the characteristic function of the scaled sum

$$A_n = \frac{X_1 + \cdots + X_n}{\sqrt{n}}$$

is given by

$$\phi_n(z) = \phi(z/\sqrt{n})^n, \qquad z \in \mathbb{R}.$$

Now

$$\phi(w) = \phi(0) + \phi'(0)w + \phi^{(2)}(0)w^2/2! + \cdots = 1 - w^2/2 + \cdots .$$

Thus, for any fixed $z \in \mathbb{R}$, we obtain

$$\phi_n(z) = \left(1 - \frac{z^2}{2n} + \cdots\right)^n \to e^{-z^2/2},$$

as $n \to \infty$. It can be shown that this implies that the scaled sum A_n converges (in probability) to the Gaussian distribution as $n \to \infty$.

We shall now prove that the interpolation matrix is positive definite, and therefore invertible, if the centres $\mathbf{b}_1, \ldots, \mathbf{b}_n$ are all different. We first demonstrate the weaker result that $A(\lambda)$ is always non-negative definite.

Theorem 7.5. *The Gaussian interpolation matrix $A(\lambda) \in \mathbb{R}^{n \times n}$, defined by*

$$A(\lambda)_{jk} = G_\lambda(\mathbf{b}_j - \mathbf{b}_k), \qquad 1 \le j, k \le n,$$

is non-negative definite, that is, $\mathbf{v}^T A(\lambda)\mathbf{v} \ge 0$, for every $\mathbf{v} \in \mathbb{R}^n$.

Proof. The following beautiful equation is vital:

$$\begin{aligned}
\mathbf{v}^T A(\lambda)\mathbf{v} &= \sum_{j=1}^n \sum_{k=1}^n v_j v_k G_\lambda(\mathbf{b}_j - \mathbf{b}_k) \\
&= \sum_{j=1}^n \sum_{k=1}^n v_j v_k \, (2\pi)^{-d} \int_{\mathbb{R}^d} \widehat{G_\lambda}(\mathbf{z}) e^{i\mathbf{z}^T(\mathbf{b}_j - \mathbf{b}_k)} \, d\mathbf{z} \\
&= (2\pi)^{-d} \int_{\mathbb{R}^d} \widehat{G_\lambda}(\mathbf{z}) \sum_{j=1}^n \sum_{k=1}^n v_j v_k e^{i\mathbf{z}^T(\mathbf{b}_j - \mathbf{b}_k)} \, d\mathbf{z} \\
&= (2\pi)^{-d} \int_{\mathbb{R}^d} \widehat{G_\lambda}(\mathbf{z}) \Big| \sum_{k=1}^n v_k e^{i\mathbf{b}_k^T \mathbf{z}} \Big|^2 \, d\mathbf{z}. \qquad (7.11)
\end{aligned}$$

Since the integrand is a non-negative function, we deduce $\mathbf{v}^T A(\lambda)\mathbf{v} \ge 0$, for every $\mathbf{v} \in \mathbb{R}^n$. Thus $A(\lambda)$ is indeed non-negative definite; its symmetry is obvious. □

If the points $\mathbf{b}_1, \ldots, \mathbf{b}_n$ are all different, then $\|\mathbf{b}_j - \mathbf{b}_k\| > 0$, for $j \ne k$. This simple observation implies that

$$\lim_{\lambda \to \infty} G_\lambda(\mathbf{b}_j - \mathbf{b}_k) = 0, \qquad \text{if } j \ne k.$$

Hence

$$\lim_{\lambda \to \infty} A(\lambda) = I$$

if the points $\mathbf{b}_1, \ldots, \mathbf{b}_n$ are all different.

Theorem 7.6. *Suppose the points* $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^d$ *are all different. Then every matrix* $A(\lambda)$ *is positive definite.*

Proof. If $\mathbf{v}^T A(\lambda)\mathbf{v} = 0$, then (7.11) implies that

$$\sum_{k=1}^{n} v_k e^{i\mathbf{b}_k^T \mathbf{z}} \equiv 0.$$

But then (7.11) also implies that $\mathbf{v}^T A(\mu)\mathbf{v} = 0$, for all $\mu > 0$. Letting $\mu \to \infty$, and recalling that $\lim_{\lambda\to\infty} A(\lambda) = I$, because the points $\mathbf{b}_1, \ldots, \mathbf{b}_n$ are all different, we deduce that $\mathbf{v}^T\mathbf{v} = 0$, that is, $\mathbf{v} = 0$. Thus $\mathbf{v}^T A\mathbf{v} \geq 0$, with equality if and only if $\mathbf{v}$ is the zero vector; but this is precisely the statement that $A(\lambda)$ is positive definite. □

Exercise 7.3. *Can* $A(\lambda)$ *be nonsingular when the points* $\mathbf{b}_1, \ldots, \mathbf{b}_n$ *are* not *all different?*

Exercise 7.4. *Let* $w : [0, \infty) \to \mathbb{R}$ *be any bounded, positive continuous function and define*

$$\phi(r) = \int_0^\infty e^{-sr^2} w(s)\, ds, \qquad r \geq 0.$$

Define $A \in \mathbb{R}^{n\times n}$ *by*

$$A_{jk} = \phi(\|\mathbf{x}_j - \mathbf{x}_k\|), \qquad 1 \leq j, k \leq n.$$

Prove that A *is non-negative definite by modifying the proof of Theorem 7.5. Is* A *positive definite when the points* $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ *are distinct?*

Exercise 7.5. *The theory of the Gamma function provides the equation*

$$\pi^{1/2} = \Gamma(1/2) = \int_0^\infty e^{-t} t^{-1/2}\, dt.$$

Use the change of variable $t = a^2 s$, *where* a *is a positive constant, to show that*

$$\frac{1}{a} = \int_0^\infty e^{-a^2 s} \frac{1}{\sqrt{\pi s}}\, ds, \qquad a > 0.$$

Use this integral to derive

$$(r^2 + c^2)^{-1/2} = \int_0^\infty e^{-r^2 s} w(s)\, ds, \qquad r \geq 0,$$

where

$$w(s) = e^{-c^2 s}(\pi s)^{-1/2}, \qquad s \geq 0.$$

Here c *is a positive constant. [This implies that the inverse multiquadric generates positive definite interpolation matrices, so can be used as a radial basis function to interpolate scattered data in any dimension* d*.]*

School of Computing and Mathematical Sciences, Birkbeck College, University of London, Malet Street, London WC1E 7HX, England

Email address: b.baxter@bbk.ac.uk