Signal Processing Course

William D. Penny

April 28, 2000

Contents

Chapter 1

Statistics

1.1 Introduction

This lecture is a quick review of basic statistical concepts; probabilities, mean, variance, covariance, correlation, linear regression, probability density functions and significance testing.

1.2 Probabilities

1.2.1 Discrete Variables

The table below shows the probability of occurrence $p(x = x_i)$ of selected letters x_i in the English alphabet. Table 2 shows the probability of occurence of selected

x_i	$p(x_i)$
a	0.06
e	0.09
j	0.00
q	0.01
t	0.07
z	0.00

Table 1.1: *Probability of letters*

pairs of letters x_i and y_j where x_i is followed by y_j. This is called the *joint probability* $p(x = x_i, y = y_i)$. If we fix x to, say x_i then the probability of y taking on a particular value, say y_j, is given by the *conditional probability*

$$p(y = y_j | x = x_i) = \frac{p(x = x_i, y = y_j)}{p(x = x_i)} \tag{1.1}$$

x_i	y_j	$p(x_i, y_j)$
t	h	0.037
t	s	0.000
t	r	0.012

Table 1.2: *Probability of pairs of letters*

For example, if $x_i = t$ and $y_j = h$ then the joint probability $p(x = x_i, y = y_j)$ is just the probability of occurence of the pair (which table 2 tells us is 0.037). The conditional probability $p(y = y_j|x = x_i)$, however, says that, *given* we've seen the letter t, what's the probability that the next letter will be h (which is, from tables 1 and 2, $0.037/0.07 = 0.53$). Re-arranging the above relationship gives

$$p(x = x_i, y = y_j) = p(y = y_j|x = x_i)p(x = x_i) \tag{1.2}$$

Now if y does *not* depend on x then $p(y = y_j|x = x_i) = p(y = y_j)$. Hence, for independent variables, we have

$$p(x = x_i, y = y_j) = p(y = y_j)p(x = x_i) \tag{1.3}$$

The *marginal probability* is given by

$$p(x = x_i) = \sum_{\{y_j\}} p(y = y_j, x = x_i) \tag{1.4}$$

This is the same probability that we started with.

1.2.2 Continuous Variables

The probability of a continuous variable, x, assuming a particular value or range of values is defined by a Probability Density Funcion (PDF), $p(x)$. *Probability is measured by the area under the PDF*; the total area under a PDF is therefore unity

$$\int p(x)dx = 1 \tag{1.5}$$

The probability of x assuming a value between a and b is given by

$$p(a \leq x \leq b) = \int_a^b p(x)dx \tag{1.6}$$

which is the area under the PDF between a and b. *The probability of x taking on a single value is therefore zero.* This makes sense because we are dealing with continuous values; as your value becomes more precise the probability for it decreases. It only makes sense, therefore to talk about the probability of a value being within a certain precision or being above or below a certain value.

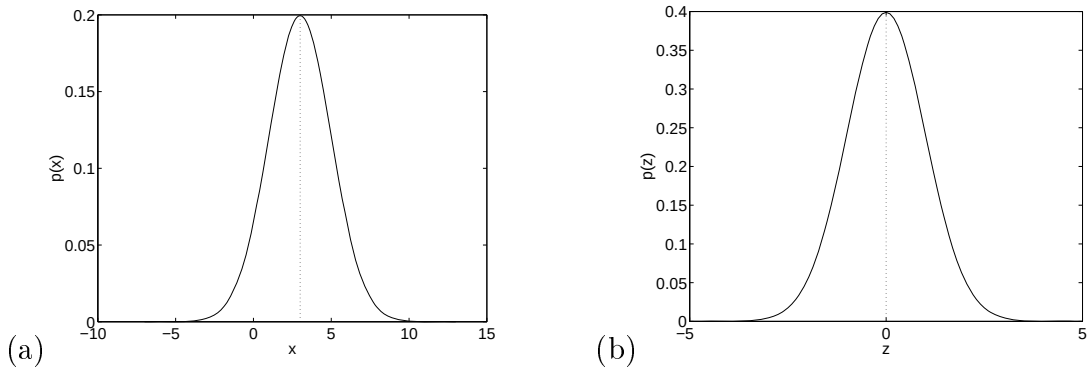


Figure 1.1: *(a) The Gaussian Probability Density Function with mean $\mu = 3$ and standard deviation $\sigma = 2$, (b) The standard Gaussian density, $p(z)$. This has zero mean and unit variance.*

To calculate such probabilities we need to calculate integrals like the one above. This process is simplified by the use of Cumulative Density Functions (CDF) which are defined as

$$CDF(a) = p(x \leq a) = \int_{-\infty}^{a} p(x)dx \tag{1.7}$$

Hence

$$p(a \leq x \leq b) = CDF(b) - CDF(a) \tag{1.8}$$

1.2.3 The Gaussian Density

The *Normal* or *Gaussian* probability density function, for the case of a single variable, is

$$p(x) \equiv N(x; \mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{1.9}$$

where μ and σ^2 are known as the *mean* and *variance*, and σ (the square root of the variance) is called the *standard deviation*. The quantity in front of the exponential ensures that $\int p(x)dx = 1$. The above formula is often abbreviated to the shorthand $p(x) = N(x; \mu, \sigma)$. The terms Normal and Gaussian are used interchangeably.

If we subtract the mean from a Gaussian variable and then divide by that variables *standard deviation* the resulting variable, $z = (x - \mu)/\sigma$, will be distributed according the *standard* normal distribution, $p(z) = N(z; 0, 1)$ which can be written

$$p(z) = \frac{1}{(2\pi)^{1/2}} \exp\left(-\frac{z^2}{2}\right) \tag{1.10}$$

The probability of z being above 0.5 is given by the area to the right of 0.5. We can calculate it as

$$\begin{aligned} p(z) \geq 0.5 &= \int_{0.5}^{\infty} p(z)dz \\ &= 1 - CDF_{Gauss}(0.5) \end{aligned} \tag{1.11}$$

where CDF_{Gauss} is the cumulative density function for a Gaussian.

1.2.4 Probability relations

The same probability relations hold for continuous variables as for discrete variables ie. the conditional probability is

$$p(y|x) = \frac{p(x,y)}{p(x)} \tag{1.12}$$

Re-arranging gives the joint probability

$$p(x,y) = p(y|x)p(x) \tag{1.13}$$

which, if y does not depend on x (ie. x and y are independent) means that

$$p(x,y) = p(y)p(x) \tag{1.14}$$

1.3 Expectation and Moments

The *expected value* of a function $f(x)$ is defined as

$$E[f(x)] \equiv < f(x) >= \int p(x)f(x)dx \tag{1.15}$$

and $E[]$ is referred to as the *expectation* operator, which is also sometimes written using the angled brackets $<>$. The kth *moment* of a distribution is given by

$$E[x^k] = \int p(x)x^k dx \tag{1.16}$$

The mean is therefore the first moment of a distribution.

$$E[x] = \int p(x)xdx = \mu \tag{1.17}$$

The kth *central moment* of a distribution is given by

$$E[(x-\mu)^k] = \int p(x)(x-\mu)^k dx \tag{1.18}$$

The variance is therefore the second central moment

$$E[(x-\mu)^2] = \int p(x)(x-\mu)^2 dx = \sigma^2 \tag{1.19}$$

Sometimes we will use the notation

$$Var(x) = E[(x-\mu)^2] \tag{1.20}$$

The third central moment is *skewness* and the fourth central moment is *kurtosis* (see later). In the appendix we give examples of various distributions and of skewness and kurtosis.

1.4 Maximum Likelihood Estimation

We can learn the mean and variance of a Gaussian distribution using the Maximum Likelihood (ML) framework as follows. A Gaussian variable x_n has the PDF

$$p(x_n) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{1.21}$$

which is also called the likelihood of the data point. Given N Independent and Identically Distributed (IID) (it is often assumed that the data points, or errors, are independent and come from the same distribution) samples $y = [y_1, y_2, .., y_N]$ we have

$$p(y) = \prod_{n=1}^{N} p(y_n) \tag{1.22}$$

which is the likelihood of the data set. We now wish to set μ and σ^2 so as to maximise this likelihood. For numerical reasons (taking logs gives us bigger numbers) this is more conveniently achieved by maximising the log-likelihood (note: the maximum is given by the same values of μ and σ)

$$L \equiv \log p(y) = -\frac{N}{2}\log 2\pi - \frac{N}{2}\log\sigma^2 - \sum_{n=}^{N} \frac{(y_n-\mu)^2}{2\sigma^2} \tag{1.23}$$

The optimal values of μ and σ are found by setting the derivatives $\frac{dL}{d\mu}$ and $\frac{dL}{d\sigma}$ to zero. This gives

$$\mu = \frac{1}{N}\sum_{n=1}^{N} y_n \tag{1.24}$$

and

$$\sigma^2 = \frac{1}{N}\sum_{n=1}^{N} (y_n-\mu)^2 \tag{1.25}$$

We note that the last formula is different to the usual formula for estimating variance

$$\sigma^2 = \frac{1}{N-1}\sum_{n=1}^{N} (x_n-\mu)^2 \tag{1.26}$$

because of the difference in normalisation. The last estimator of variance is preferred as it is an *unbiased* estimator (see later section on bias and variance).

If we had an input-dependent mean, $\mu_n = wx_n$, then the optimal value for w can be found by maximising L. As only the last term in equation 1.23 depends on w this therefore corresponds to minimisation of the squared errors between μ_n and y_n. This provides the connection between ML estimation and Least Squares (LS) estimation; ML reduces to LS for the case of Gaussian noise.

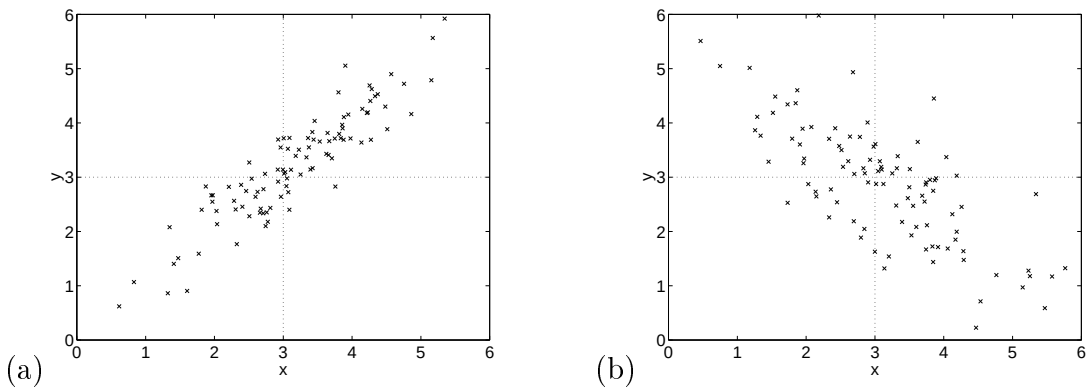


Figure 1.2: *(a) Positive correlation,* $r = 0.9$ *and (b) Negative correlation,* $r = -0.7$*. The dotted horizontal and vertical lines mark* μ_x *and* μ_y*.*

1.5 Correlation and Regression

1.5.1 Correlation

The *covariance* between two variables x and y is measured as

$$\sigma_{xy} = \frac{1}{N-1} \sum_{n=1}^{N} (x_i - \mu_x)(y_i - \mu_y) \tag{1.27}$$

where μ_x and μ_y are the means of each variable. Note that $\sigma_{yx} = \sigma_{xy}$. Sometimes we will use the notation

$$Var(x, y) = \sigma_{xy} \tag{1.28}$$

If x tends to be above its mean when y is above its mean then σ_{xy} will be positive. If they tend to be on opposite sides of their means σ_{xy} will be negative. The *correlation* or *Pearson's correlation coefficient* is a normalised covariance

$$r = \frac{\sigma_{xy}}{\sigma_x \sigma_y} \tag{1.29}$$

such that $-1 \leq r \leq 1$, a value of -1 indicating perfect negative correlation and a value of $+1$ indicating perfect positive correlation; see Figure 1.2. A value of 0 indicates no correlation. The strength of a correlation is best measured by r^2 which takes on values between 0 and 1, a value near to 1 indicating strong correlation (regardless of the sign) and a value near to zero indicating a very weak correlation.

1.5.2 Linear regression

We now look at modelling the relationship between two variables x and y as a linear function; given a collection of N data points $\{x_i, y_i\}$, we aim to estimate y_i from x_i using a linear model

$$\hat{y}_i = ax_i + b \tag{1.30}$$

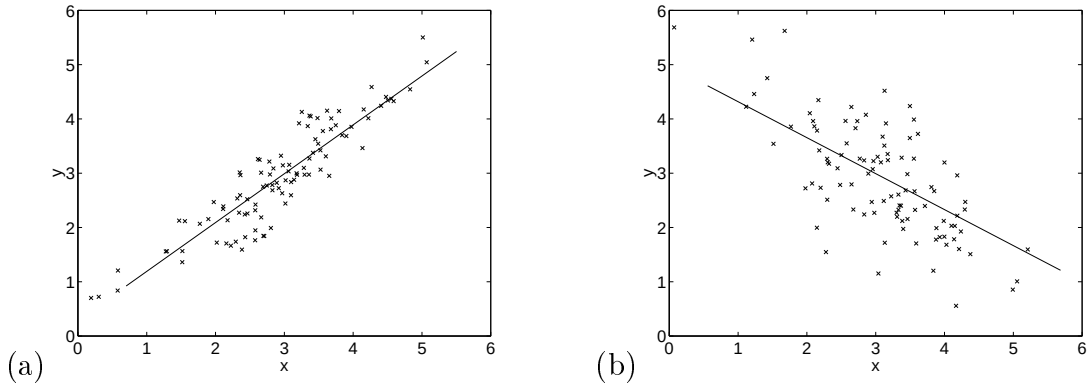


Figure 1.3: *The linear regression line is fitted by minimising the vertical distance between itself and each data point. The estimated lines are (a)* $\hat{y} = 0.9003x + 0.2901$ *and (b)* $\hat{y} = -0.6629x + 4.9804$.

where we have written $\hat{y}$ to denote our estimated value. Regression with one input variable is often called *univariate* linear regression to distinguish it from *multivariate* linear regression where we have lots of inputs. The goodness of fit of the model to the data may be measured by the least squares cost function

$$E = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{1.31}$$

The values of a and b that minimize the above cost function can be calculated by setting the first derivatives of the cost function to zero and solving the resulting simultaneous equations (derivatives are used to find maxima and minima of functions). The result is derived in the Appendix. The solutions are

$$a = \frac{\sigma_{xy}}{\sigma_x^2} \tag{1.32}$$

and

$$b = \mu_y - a\mu_x \tag{1.33}$$

where μ_x and μ_y are the mean observed values of the data and σ_x^2 and σ_{xy} are the input variance and input-output covariance. This enables least squares fitting of a regression line to a data set as shown in Figure 1.3.

The model will fit some data points better than others; those that it fits well constitute the *signal* and those that it does'nt fit well constitute the *noise*. The strength of the noise is measured by the noise variance

$$\sigma_e^2 = \frac{1}{N-1} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{1.34}$$

and the strenth of the signal is given by $\sigma_y^2 - \sigma_e^2$. The *signal-to-noise ratio* is therefore $(\sigma_y^2 - \sigma_e^2)/\sigma_e^2$.

Splitting data up into signal and noise components in this manner (ie. breaking down the variance into what the model *explains* and what it does not) is at the heart of statistical procedures such as analysis of variance (ANOVA) [32].

Relation to correlation

The correlation measure r is intimately related to the linear regression model. Indeed (by substituting σ_{xy} from equation 1.27 into equation 1.32) r may be expressed as

$$r = \frac{\sigma_x}{\sigma_y} a \tag{1.35}$$

where a is the slope of the linear regression model. Thus, for example, the sign of the slope of the regression line defines the sign of the correlation. The correlation is, however, also a function of the standard deviation of the x and y variables; for example, if σ_x is very large, it is possible to have a strong correlation even though the slope may be very small.

The relation between r and linear regression emphasises the fact that r is only a measure of *linear* correlation. It is quite possible that two variables have a strong nonlinear relationship (ie. are nonlinearly correlated) but that $r = 0$. Measures of nonlinear correlation will be discussed in a later lecture.

The strenth of correlation can also be expressed in terms of quantites from the linear regresssion model

$$r^2 = \frac{\sigma_y^2 - \sigma_e^2}{\sigma_y^2} \tag{1.36}$$

where σ_e^2 is the noise variance and σ_y^2 is the variance of the variable we are trying to predict. Thus r^2 is seen to measure the proportion of variance explained by a linear model, a value of 1 indicating that a linear model perfectly describes the relationship between x and y.

1.6 Bias and Variance

Given any estimation process, if we repeat it many times we can look at the expected (or average) errors (the difference between true and estimated values). This is comprised of a systematic error (the 'bias') and an error due to the variability of the fitting process (the 'variance'). We can show this as follows.

Let w be the true value of a parameter and $\hat{w}$ be an estimate from a given sample. The expected squared error of the estimate can be decomposed as follows

$$\begin{aligned} E &= E[(\hat{w} - w)^2] \\ &= E[(\hat{w} - E[\hat{w}] + E[\hat{w}] - w)^2] \end{aligned} \tag{1.37}$$

where the expectation is wrt. the distribution over $\hat{w}$ and we have introduced $E[\hat{w}]$, the mean value of the estimate. Expanding the square gives

$$\begin{aligned} E &= E[(\hat{w} - E[\hat{w}])^2 + (E[\hat{w}] - w)^2 + 2(\hat{w} - E[\hat{w}])(E[\hat{w}] - w)] \\ &= E[(\hat{w} - E[\hat{w}])]^2 + (E[\hat{w}] - w)^2 \\ &= V + B^2 \end{aligned} \tag{1.38}$$

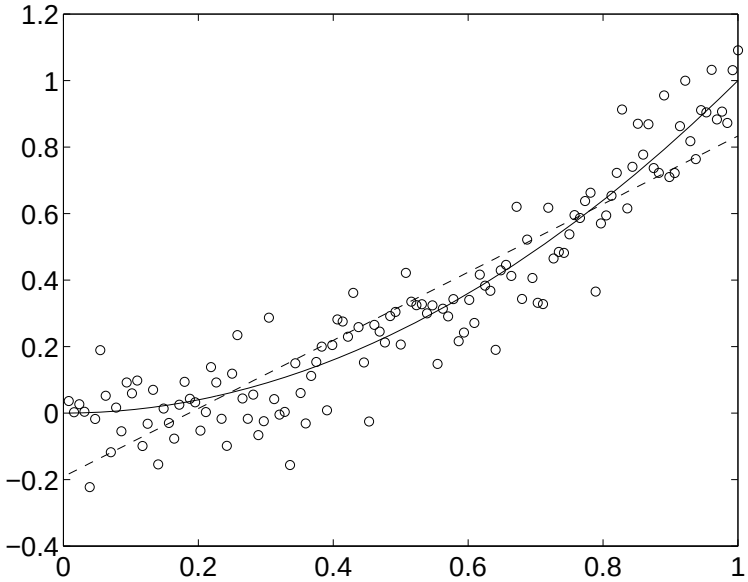


Figure 1.4: *Fitting a linear regression model (dotted line) to data points (circles) which are generated from a quadratic function (solid line) with additive noise (of variance 0.01).*

where the third term has dropped out because $E[\hat{w}] - E[\hat{w}] = 0$. The error thus consists of two terms (i) a variance term V and (ii) a bias term; the square of the bias, B^2.

Estimates of parameters are often chosen to be *unbiased* ie. to have zero bias. This is why we see the $1/(N-1)$ term in an estimate of variance, for example.

Simple models (eg. linear models) have a high bias but low variance whereas more complex models (eg. polynomial models) have a low bias but a high variance. To select the optimal model complexity, or model order, we must solve this *bias-variance dilemma* [20].

1.7 Minimum variance estimation

There is a lower bound to the variance of any unbiased estimate which is given by

$$Var(\hat{\theta}) \geq \frac{1}{E[\partial L(D,\theta)/\partial\theta]^2} \tag{1.39}$$

where $L(D;\theta) \equiv \log p(D;\theta)$ is the log-likelihood of the data and the expectation is taken wrt. $p(D;\theta)$. This is known as the *Cramer-Rao bound*. Any estimator that attains this variance is called the Minimum Variance Unbiased Estimator (MVUE).

The denominator, being an inverse variance, therefore measures the maximum precision with which we can estimate θ. It is known as the *Fisher Information*

$$I(\theta) = E[\partial L(D;\theta)/\partial\theta]^2 \tag{1.40}$$

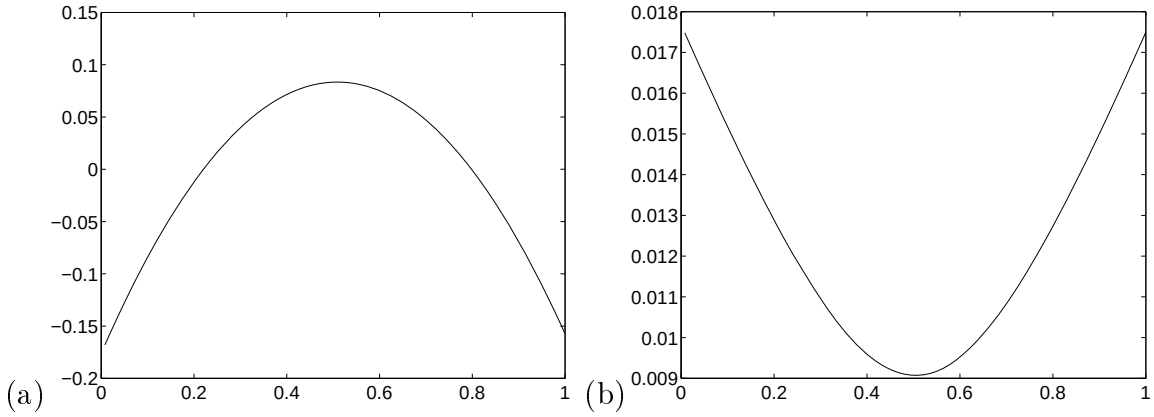


Figure 1.5: *(a) Bias component B and (b) Variance component V. The bias represents a systematic error in our modelling procedure (ie. fitting a quadratic function with a linear function); the linear model systematically underpredicts at the edges and overpredicts in the middle. The variance represents the variability of the model fitting process; linear models lock on to the middle of a data set and then set their slope as necessary. The variance is therefore less in the middle than at the edges; in the middle this variance is simply the variance of the additive noise (0.01). The expected prediction error at any point is the sum of the variance plus the bias squared.*

For unbiased estimates [53] it can also be expressed as

$$I(\theta) = -E[\partial^2 L(D;\theta)/\partial\theta^2] \tag{1.41}$$

1.8 Statistical Inference

When we estimate the mean and variance from small samples of data our estimates may not be very accurate. But as the number of samples increases our estimates get more and more accurate and as this number approaches infinity the sample mean approaches the true mean or *population* mean. In what follows we refer to the sample means and variances as m and s and the population means and standard deviations as μ and σ.

Hypothesis Testing: Say we have a hypothesis $\boldsymbol{H}$ which is *The mean value of my signal is 32.* This is often referred to as the *null hypothesis* or H_0. We then get some data and test $\boldsymbol{H}$ which is then either *accepted* or *rejected* with a certain probability or *significance level*, p. Very often we choose $p = 0.05$ (a value used throughout science).

We can do a *one-sided* or a *two-sided* statistical test depending on exactly what the null hypothesis is. In a one-sided test our hypothesis may be (i) our parameter is less than x or (ii) our parameter is greater than x. For two-sided tests our hypothesis is of the form (iii) our parameter is x. This last hypothesis can be rejected if the sample statistic is either much smaller or much greater than it should be if the parameter truly equals x.

1.8.1 Means

To find out if your mean is significantly different from a hypothesized value μ there are basically two methods. The first assumes you know the population/true variance and the second allows you to use the sample variance.

Known variance

If we estimate the mean from a sample of data, then this estimate itself has a mean and a standard deviation. The standard deviation of the sample mean is (see appendix)

$$\sigma_m = \sigma/\sqrt{N} \tag{1.42}$$

where σ is the known true standard deviation. The probability of getting a particular sample mean from N samples is given by $p(z)$ where

$$z = \frac{m-\mu}{\sigma/\sqrt{N}} \tag{1.43}$$

For example, suppose we are given 50 data points from a normal population with hypothesized mean $\mu = 32$ and standard deviation $\sigma = 2$ and we get a sample mean of 32.3923, as shown in Figure 1.6. The probability of getting a sample mean *at least* this big is

$$p(m > 32.3923) = 1 - CDF_{Gauss}(z) \tag{1.44}$$

where $z = (32.3923 - 32)/(2/\sqrt{50}) = 1.3869$ which is (from tables or computer evaluation) 0.0827 ie. reasonably likely; we would accept the hypothesis at the $p = 0.05$ level (because we are doing a two-sided test we would accept H_0 unless the probability was less than $p = 0.025$).

Unknown variance

If we don't know the true variance we can use the sample variance instead. We can then calculate the statistic

$$t = \frac{m-\mu}{s/\sqrt{N}} \tag{1.45}$$

which is distributed according the t-distribution (see appendix). Now, the t-distribution has a parameter v, called the degrees of freedom (DF). It is plotted in Figure 1.7 with $v = 3$ and $v = 49$ degrees of freedom; smaller v gives a wider distribution.

Now, from our $N = 50$ data points we calculated the sample variance ie. given, originally, 50 DF we have used up one DF leaving $N - 1 = 49$ DF. Hence, our t-statistic has $v = 49$ degrees of freedom.

Assume we observed $s = 2$ and $m = 32.3923$ (as before) and our hypothesized mean is 32. We can calculate the associated probability from

$$p(m > 32.3923) = 1 - CDF_t(t) \tag{1.46}$$

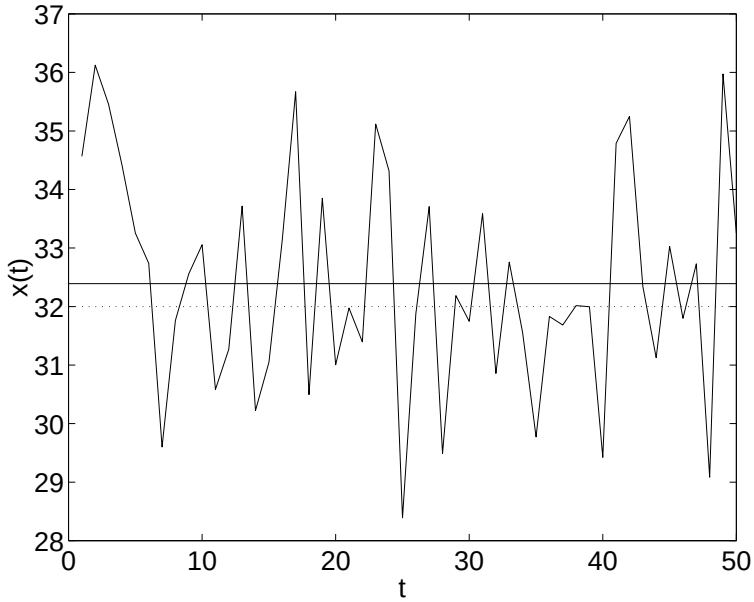


Figure 1.6: *N=50 data points. The hypthosized mean value of 32 is shown as a dotted line and the sample mean as a solid line.*

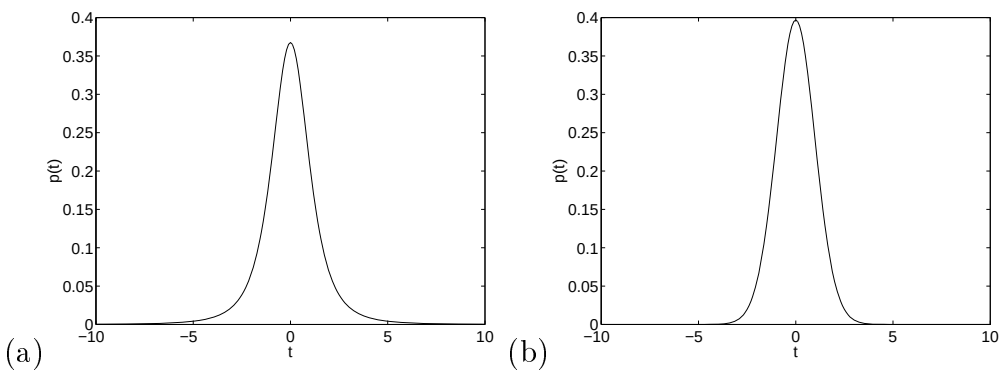


Figure 1.7: *The t-distribution with (a) $v = 3$ and (b) $v = 49$ degrees of freedom.*

where $t = (32.3923 - 32/(2/\sqrt{50}) = 1.3869$. From tables this gives 0.0859 ie. reasonably likely (again, because we are doing a two-sided test, we would accept H_0 unless the probability was less than $p = 0.025$). Notice, however, that the probability is higher than when we *knew* the standard deviation to be 2. This shows that a t-distribution has heavier tails than a Normal distribution ie. extreme events are more likely.

1.8.2 Regression

In a linear regression model we are often interested in whether or not the gradient is significantly different from zero or other value of interest.

To answer the question we first estimate the variance of the slope and then perform

a t-test. In the appendix we show that the variance of the slope is given by [1]

$$\sigma_a^2 = \frac{\sigma_e^2}{(N-1)\sigma_x^2} \tag{1.47}$$

We then calculate the t-statistic

$$t = \frac{a - a_h}{\sigma_a} \tag{1.48}$$

where a_h is our hypothesized slope value (eg. a_h may be zero) and look up $p(t)$ with $N-2$ DF (we have used up 1DF to estimate the input variance and 1DF to estimate the noise variance). In the data plotted in Figure 1.3(b) the estimated slope is $a = -0.6629$. From the data we also calculate that $\sigma_a = 0.077$. Hence, to find out if the slope is significantly non-zero we compute $CDF_t(t)$ where $t = -0.6629/0.077 = -8.6$. This has a p-value of 10^{-13} ie. a very significant value. To find out if the slope is significantly different from -0.7 we calculate $CDF_t(t)$ for $t = (-0.6629+0.7)/0.077 = 0.4747$ which gives a p-value of 0.3553 ie. not significantly different (again, we must bear in mind that we need to do a two-sided test; see earlier).

1.8.3 Correlation

Because of the relationship between correlation and linear regression we can find out if correlations are significantly non-zero by using exactly the same method as in the previous section; if the slope is significantly non-zero then the corresponding correlation is also significantly non-zero.

By substituting $a = (\sigma_y/\sigma_x)r$ (this follows from equation 1.32 and equation 1.29) and $\sigma_e^2 = (1-r^2)\sigma_y^2$ (from equation 1.36) into equation 1.47 and then σ_a into equation 1.48 we get the test statistic [2]

$$t = \frac{r\sqrt{N-2}}{\sqrt{1-r^2}} \tag{1.49}$$

which has $N-2$ DF.

For example, the two signals in Figure 1.8(a) have, over the $N = 50$ given samples, a correlation of $r = 0.8031$ which gives $t = 9.3383$ and a p-value of 10^{-12}. We therefore reject the hypothesis that the signals are not correlated; they clearly are. The signals in Figure 1.8(b) have a correlation of $r = 0.1418$ over the $N = 50$ given samples which gives $t = 0.9921$ and a p-value of $p = 0.1631$. We therefore accept the null hypothesis that the signals are not correlated.

[1] When estimating σ_x^2 we should divide by $N-1$ and when estimating σ_e^2 we should divide by $N-2$.

[2] Strictly, we should use $\sigma_e^2 = \frac{N-1}{N-2}(1-r^2)\sigma_y^2$ to allow for using $N-2$ in the denominator of σ_e^2.

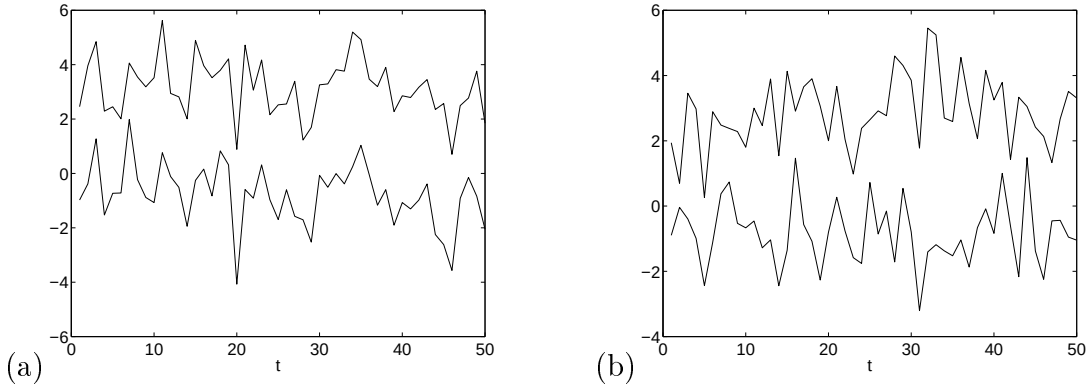


Figure 1.8: *Two signals (a) sample correlation* $r = 0.8031$ *and (b) sample correlation,* r=0.1418. *Strong correlation; by shifting and scaling one of the time series (ie. taking a linear function) we can make it look like the other time series.*

1.9 Discussion

For a more comprehensive introduction to basic statistics, linear regression and significance testing see Grimmett and Welsh [26] or Kleinbaum et al. [32]. Also, *Numerical Recipes* [49] has very good sections on *Are two means different ?* and *Are two variances different ?*. See Priestley for a more comprehensive introduction to statistical estimation in time series models ([50], chapter 5).

Chapter 2

Linear Algebra

2.1 Introduction

We discuss vectors, matrices, transposes, covariance, correlation, diagonal and inverse matrices, orthogonality, subspaces and eigenanalysis. An alterntive source for much of this material is the excellent book by Strang [58].

2.2 Transposes and Inner Products

A collection of variables may be treated as a single entity by writing them as a *vector*. For example, the three variables x_1, x_2 and x_3 may be written as the vector

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{2.1}$$

Bold face type is often used to denote vectors (scalars - single variables - are written with normal type). Vectors can be written as *column vectors* where the variables go down the page or as *row vectors* where the variables go across the page (it needs to be made clear when using vectors whether $\boldsymbol{x}$ means a row vector or a column vector - most often it will mean a column vector and in our text it will *always* mean a column vector, unless we say otherwise). To turn a column vector into a row vector we use the *transpose* operator

$$\boldsymbol{x}^T = [x_1, x_2, x_3] \tag{2.2}$$

The transpose operator also turns row vectors into column vectors. We now define the *inner product* of two vectors

$$\begin{aligned} \boldsymbol{x}^T\boldsymbol{y} &= [x_1, x_2, x_3] \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \\ &= x_1y_1 + x_2y_2 + x_3y_3 \end{aligned} \tag{2.3}$$

$$= \sum_{i=1}^{3} x_i y_i$$

which is seen to be a scalar. The *outer product* of two vectors produces a matrix

$$\begin{aligned} \boldsymbol{x}\boldsymbol{y}^T &= \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} [y_1, y_2, y_3] \\ &= \begin{bmatrix} x_1 y_1 & x_1 y_2 & x_1 y_3 \\ x_2 y_1 & x_2 y_2 & x_2 y_3 \\ x_3 y_1 & x_3 y_2 & x_3 y_3 \end{bmatrix} \end{aligned} \tag{2.4}$$

An $N \times M$ matrix has N rows and M columns. The ijth entry of a matrix is the entry on the jth column of the ith row. Given a matrix $\boldsymbol{A}$ (matrices are also often written in bold type) the ijth entry is written as $\boldsymbol{A}_{ij}$. When applying the transpose operator to a matrix the ith row becomes the ith column. That is, if

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{2.5}$$

then

$$\boldsymbol{A}^T = \begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \\ a_{13} & a_{23} & a_{33} \end{bmatrix} \tag{2.6}$$

A matrix is *symmetric* if $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji}$. Another way to say this is that, for symmetric matrices, $\boldsymbol{A} = \boldsymbol{A}^T$.

Two matrices can be multiplied if the number of columns in the first matrix equals the number of rows in the second. Multiplying $\boldsymbol{A}$, an $N \times M$ matrix, by $\boldsymbol{B}$, an $M \times K$ matrix, results in $\boldsymbol{C}$, an $N \times K$ matrix. The ijth entry in $\boldsymbol{C}$ is the inner product between the ith row in $\boldsymbol{A}$ and the jth column in $\boldsymbol{B}$. As an example

$$\begin{bmatrix} 2 & 3 & 4 \\ 5 & 6 & 7 \end{bmatrix} \begin{bmatrix} 1 & 3 & 7 & 2 \\ 4 & 3 & 4 & 1 \\ 5 & 6 & 4 & 2 \end{bmatrix} = \begin{bmatrix} 34 & 39 & 42 & 15 \\ 64 & 75 & 87 & 30 \end{bmatrix} \tag{2.7}$$

Given two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ we note that

$$(\boldsymbol{A}\boldsymbol{B})^T = \boldsymbol{B}^T \boldsymbol{A}^T \tag{2.8}$$

2.2.1 Properties of matrix multiplication

Matrix multiplication is associative

$$(\boldsymbol{A}\boldsymbol{B})\boldsymbol{C} = \boldsymbol{A}(\boldsymbol{B}\boldsymbol{C}) \tag{2.9}$$

distributive

$$\boldsymbol{A}(\boldsymbol{B} + \boldsymbol{C}) = \boldsymbol{A}\boldsymbol{B} + \boldsymbol{A}\boldsymbol{C} \tag{2.10}$$

but not commutative

$$\boldsymbol{A}\boldsymbol{B} \neq \boldsymbol{B}\boldsymbol{A} \tag{2.11}$$

2.3 Types of matrices

2.3.1 Covariance matrices

In the previous chapter the covariance, σ_{xy}, between two variables x and y was defined. Given p variables there are $p \times p$ covariances to take account of. If we write the covariances between variables x_i and x_j as σ_{ij} then all the covariances can be summarised in a *covariance matrix* which we write below for $p = 3$

$$\boldsymbol{C} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_2^2 & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 \end{bmatrix} \tag{2.12}$$

The ith diagonal element is the covariance between the ith variable and itself which is simply the variance of that variable; we therefore write σ_i^2 instead of σ_{ii}. Also, note that because $\sigma_{ij} = \sigma_{ji}$ covariance matrices are symmetric.

We now look at computing a covariance matrix from a given data set. Suppose we have p variables and that a single observation $\boldsymbol{x}_i$ (a row vector) consists of measuring these variables and suppose there are N such observations. We now make a matrix $\boldsymbol{X}$ by putting each $\boldsymbol{x}_i$ into the ith row. The matrix $\boldsymbol{X}$ is therefore an $N \times p$ matrix whose rows are made up of different observation vectors. If all the variables have zero mean then the covariance matrix can then be evaluated as

$$\boldsymbol{C} = \frac{1}{N-1}\boldsymbol{X}^T\boldsymbol{X} \tag{2.13}$$

This is a multiplication of a $p \times N$ matrix, $\boldsymbol{X}^T$, by a $N \times p$ matrix, $\boldsymbol{X}$, which results in a $p \times p$ matrix. To illustrate the use of covariance matrices for time series, figure 2.1 shows 3 time series which have the following covariance relation

$$\boldsymbol{C}_1 = \begin{bmatrix} 1 & 0.1 & 1.6 \\ 0.1 & 1 & 0.2 \\ 1.6 & 0.2 & 2.0 \end{bmatrix} \tag{2.14}$$

and mean vector

$$\boldsymbol{m}_1 = [13, 17, 23]^T \tag{2.15}$$

2.3.2 Diagonal matrices

A *diagonal matrix* is a square matrix ($M = N$) where all the entries are zero except along the diagonal. For example

$$\boldsymbol{D} = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 6 \end{bmatrix} \tag{2.16}$$

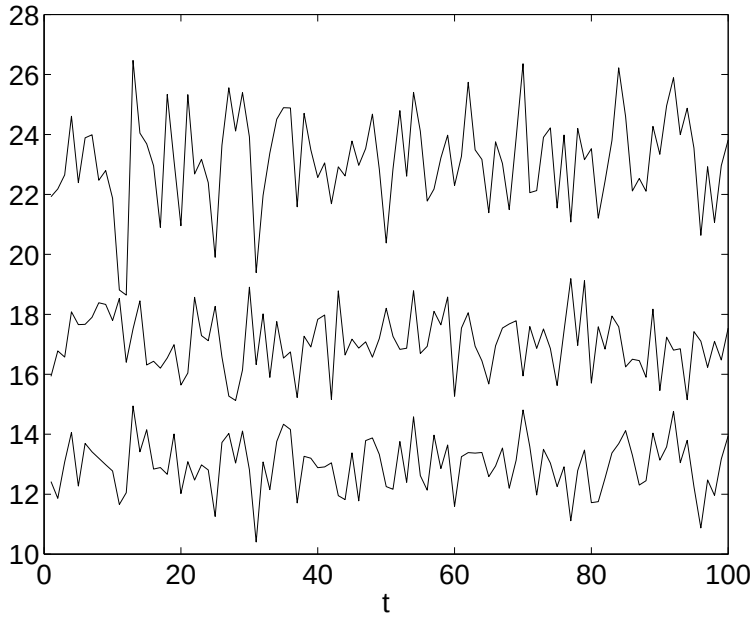


Figure 2.1: *Three time series having the covariance matrix* $\boldsymbol{C}_1$ *and mean vector* $\boldsymbol{m}_1$ *shown in the text. The top and bottom series have high covariance but none of the other pairings do.*

There is also a more compact notation for the same matrix

$$\boldsymbol{D} = diag([4, 1, 6]) \tag{2.17}$$

If a covariance matrix is diagonal it means that the covariances between variables are zero, that is, the variables are all uncorrelated. Non-diagonal covariance matrices are known as *full* covariance matrices. If $\boldsymbol{V}$ is a vector of variances $\boldsymbol{V} = [\sigma_1^2, \sigma_2^2, \sigma_3^2]^T$ then the corresponding diagonal covariance matrix is $\boldsymbol{V}_d = diag(\boldsymbol{V})$.

2.3.3 The correlation matrix

The correlation matrix, $\boldsymbol{R}$, can be derived from the covariance matrix by the equation

$$\boldsymbol{R} = \boldsymbol{BCB} \tag{2.18}$$

where $\boldsymbol{B}$ is a diagonal matrix of inverse standard deviations

$$\boldsymbol{B} = diag([1/\sigma_1, 1/\sigma_2, 1/\sigma_3]) \tag{2.19}$$

2.3.4 The identity matrix

The identity matrix is a diagonal matrix with ones along the diagonal. Multiplication of any matrix, $\boldsymbol{X}$ by the identity matrix results in $\boldsymbol{X}$. That is

$$\boldsymbol{IX} = \boldsymbol{X} \tag{2.20}$$

The identity matrix is the matrix equivalent of multiplying by 1 for scalars.

2.4 The Matrix Inverse

Given a matrix $\boldsymbol{X}$ its inverse $\boldsymbol{X}^{-1}$ is defined by the properties

$$\begin{aligned} \boldsymbol{X}^{-1}\boldsymbol{X} &= \boldsymbol{I} \\ \boldsymbol{X}\boldsymbol{X}^{-1} &= \boldsymbol{I} \end{aligned} \tag{2.21}$$

where $\boldsymbol{I}$ is the identity matrix. The inverse of a diagonal matrix with entries d_{ii} is another diagonal matrix with entries $1/d_{ii}$. This satisfies the definition of an inverse, eg.

$$\begin{bmatrix} 4 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 6 \end{bmatrix} \begin{bmatrix} 1/4 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1/6 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.22}$$

More generally, the calculation of inverses involves a lot more computation. Before looking at the general case we first consider the problem of solving simultaneous equations. These constitute relations between a set of *input or independent* variables $\boldsymbol{x}_i$ and a set of *output or dependent* variables y_i. Each input-output pair constitutes an observation. In the following example we consider just $N = 3$ observations and $p = 3$ dimensions per observation

$$\begin{aligned} 2w_1 \quad & +w_2 + w_3 &= 5 \\ 4w_1 \quad & -6w_2 &= -2 \\ -2w_1 \quad & +7w_2 + 2w_3 &= 9 \end{aligned}$$

which can be written in matrix form

$$\begin{bmatrix} 2 & 1 & 1 \\ 4 & -6 & 0 \\ -2 & 7 & 2 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 5 \\ -2 \\ 9 \end{bmatrix} \tag{2.23}$$

or in matrix form

$$\boldsymbol{X}\boldsymbol{w} = \boldsymbol{y} \tag{2.24}$$

This system of equations can be solved in a systematic way by subtracting multiples of the first equation from the second and third equations and then subtracting multiples of the second equation from the third. For example, subtracting twice the first equation from the second and -1 times the first from the third gives

$$\begin{bmatrix} 2 & 1 & 1 \\ 0 & -8 & -2 \\ 0 & 8 & 3 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 5 \\ -12 \\ 4 \end{bmatrix} \tag{2.25}$$

Then, subtracting -1 times the second from the third gives

$$\begin{bmatrix} 2 & 1 & 1 \\ 0 & -8 & -2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 5 \\ -12 \\ 2 \end{bmatrix} \tag{2.26}$$

This process is known as *forward elimination*. We can then substitute the value for w_3 from the third equation into the second etc. This process is *back-substitution*. The

two processes are together known as *Gaussian elimination*. Following this through for our example we get $\boldsymbol{w} = [1, 1, 2]^T$.

When we come to invert a matrix (as opposed to solve a system of equations as in the previous example) we start with the equation

$$\boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{I} \tag{2.27}$$

and just write down all the entries in the $\boldsymbol{A}$ and $\boldsymbol{I}$ matrices in one big matrix

$$\begin{bmatrix} 2 & 1 & 1 & 1 & 0 & 0 \\ 4 & -6 & 0 & 0 & 1 & 0 \\ -2 & 7 & 2 & 0 & 0 & 1 \end{bmatrix} \tag{2.28}$$

We then perform forward elimination [1] until the part of the matrix corresponding to $\boldsymbol{A}$ equals the identity matrix; the matrix on the right is then $\boldsymbol{A}^{-1}$ (this is because in equation 2.27 if $\boldsymbol{A}$ becomes $\boldsymbol{I}$ then the left hand side is $\boldsymbol{A}^{-1}$ and the right side must equal the left side). We get

$$\begin{bmatrix} 1 & 0 & 0 & \frac{12}{16} & \frac{-5}{16} & \frac{-6}{16} \\ 0 & 1 & 0 & \frac{4}{8} & \frac{-3}{8} & \frac{-2}{8} \\ 0 & 0 & 1 & -1 & 1 & 1 \end{bmatrix} \tag{2.29}$$

This process is known as the *Gauss-Jordan* method. For more details see Strang's excellent book on Linear Algebra [58] where this example was taken from.

Inverses can be used to solve equations of the form $\boldsymbol{X}\boldsymbol{w} = \boldsymbol{y}$. This is achieved by multiplying both sides by $\boldsymbol{X}^{-1}$ giving

$$\boldsymbol{w} = \boldsymbol{X}^{-1}\boldsymbol{y} \tag{2.30}$$

Hence,

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} \frac{12}{16} & \frac{-5}{16} & \frac{-6}{16} \\ \frac{4}{8} & \frac{-3}{8} & \frac{-2}{8} \\ -1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 5 \\ -2 \\ 9 \end{bmatrix} \tag{2.31}$$

which also gives $\boldsymbol{w} = [1, 1, 2]^T$.

The inverse of a product of matrices is given by

$$(\boldsymbol{A}\boldsymbol{B})^{-1} = \boldsymbol{B}^{-1}\boldsymbol{A}^{-1} \tag{2.32}$$

Only square matrices are invertible because, for $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$, if $\boldsymbol{y}$ and $\boldsymbol{x}$ are of different dimension then we will not necessarily have a one-to-one mapping between them.

[1] We do not perform back-substitution but instead continue with forward elimination until we get a diagonal matrix.

2.5 Orthogonality

The length of a d-element vector $\boldsymbol{x}$ is written as $||\boldsymbol{x}||$ where

$$\begin{aligned} ||\boldsymbol{x}||^2 &= \sum_{i=1}^{d} x_i^2 \\ &= \boldsymbol{x}^T \boldsymbol{x} \end{aligned} \tag{2.33}$$

Two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ are *orthogonal* if

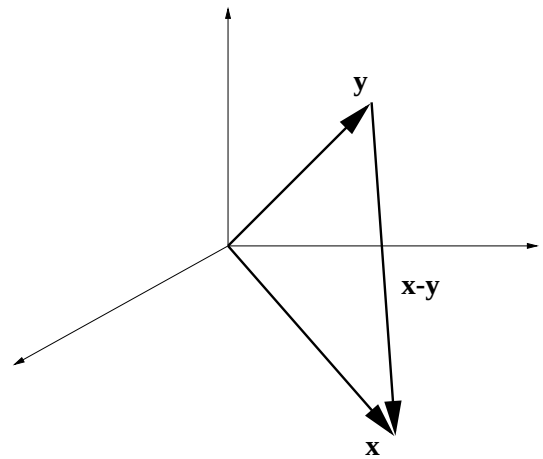


Figure 2.2: *Two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. These vectors will be orthogonal if they obey Pythagoras' relation ie. that the sum of the squares of the sides equals the square of the hypoteneuse.*

$$||\boldsymbol{x}||^2 + ||\boldsymbol{y}||^2 = ||\boldsymbol{x} - \boldsymbol{y}||^2 \tag{2.34}$$

That is, if

$$x_1^2 + ... + x_d^2 + y_1^2 + ... + y_d^2 = (x_1 - y_1)^2 + ... + (x_d - y_d)^2 \tag{2.35}$$

Expanding the terms on the right and re-arranging leaves only the cross-terms

$$\begin{aligned} x_1 y_1 + ... + x_d y_d &= 0 \\ \boldsymbol{x}^T \boldsymbol{y} &= 0 \end{aligned} \tag{2.36}$$

That is, two vectors are orthogonal if their inner product is zero.

2.5.1 Angles between vectors

Given a vector $\boldsymbol{b} = [b_1, b_2]^T$ and a vector $\boldsymbol{a} = [a_1, a_2]^T$ we can work out that

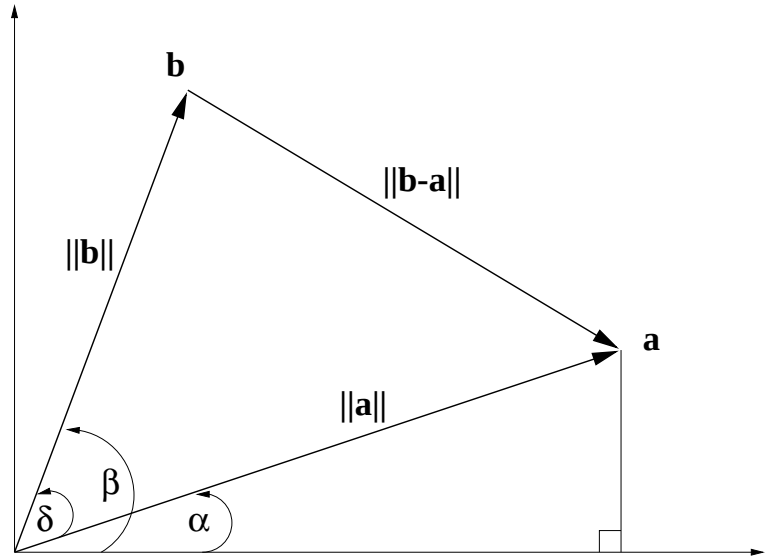


Figure 2.3: *Working out the angle between two vectors.*

$$\cos\alpha = \frac{a_1}{||\boldsymbol{a}||} \tag{2.37}$$

$$\sin\alpha = \frac{a_2}{||\boldsymbol{a}||}$$

$$\cos\beta = \frac{b_1}{||\boldsymbol{b}||}$$

$$\sin\beta = \frac{b_2}{||\boldsymbol{b}||} \tag{2.38}$$

Now, $cos\delta = cos(\beta - \alpha)$ which we can expand using the trig identity

$$\cos(\beta - \alpha) = \cos\beta\cos\alpha + \sin\beta\sin\alpha \tag{2.39}$$

Hence

$$\cos(\delta) = \frac{a_1 b_1 + a_2 b_2}{||\boldsymbol{a}||||\boldsymbol{b}||} \tag{2.40}$$

More generally, we have

$$\cos(\delta) = \frac{\boldsymbol{a}^T\boldsymbol{b}}{||\boldsymbol{a}||||\boldsymbol{b}||} \tag{2.41}$$

Because, $\cos\pi/2 = 0$, this again shows that vectors are orthogonal for $\boldsymbol{a}^T\boldsymbol{b} = 0$. Also, because $|\cos\delta| \leq 1$ where $|x|$ denotes the absolute value of x we have

$$|\boldsymbol{a}^T\boldsymbol{b}| \leq ||\boldsymbol{a}||||\boldsymbol{b}|| \tag{2.42}$$

which is known as the *Schwarz Inequality.*

2.5.2 Projections

The projection of a vector $\boldsymbol{b}$ onto a vector $\boldsymbol{a}$ results in a projection vector $\boldsymbol{p}$ which is the point on the line $\boldsymbol{a}$ which is closest to the point $\boldsymbol{b}$. Because $\boldsymbol{p}$ is a point on $\boldsymbol{a}$ it

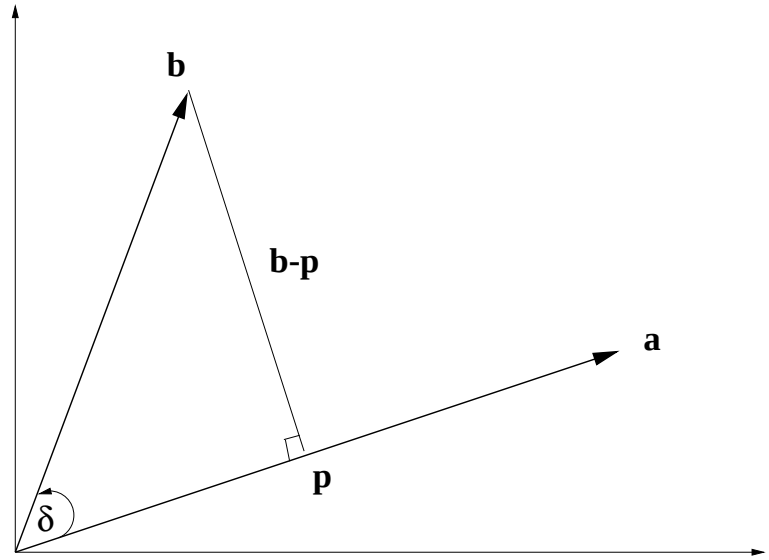


Figure 2.4: *The projection of* $\boldsymbol{b}$ *onto* $\boldsymbol{a}$ *is the point on* $\boldsymbol{a}$ *which is closest to* $\boldsymbol{b}$.

must be some scalar multiple of it. That is

$$\boldsymbol{p} = w\boldsymbol{a} \tag{2.43}$$

where w is some coefficient. Because $\boldsymbol{p}$ is the point on $\boldsymbol{a}$ *closest* to $\boldsymbol{b}$ this means that the vector $\boldsymbol{b} - \boldsymbol{p}$ is orthogonal to $\boldsymbol{a}$. Therefore

$$\begin{aligned} \boldsymbol{a}^T(\boldsymbol{b} - \boldsymbol{p}) &= 0 \\ \boldsymbol{a}^T(\boldsymbol{b} - w\boldsymbol{a}) &= 0 \end{aligned} \tag{2.44}$$

Re-arranging gives

$$w = \frac{\boldsymbol{a}^T\boldsymbol{b}}{\boldsymbol{a}^T\boldsymbol{a}} \tag{2.45}$$

and

$$\boldsymbol{p} = \frac{\boldsymbol{a}^T\boldsymbol{b}}{\boldsymbol{a}^T\boldsymbol{a}}\boldsymbol{a} \tag{2.46}$$

We refer to $\boldsymbol{p}$ as the *projection vector* and to w as the *projection*.

2.5.3 Orthogonal Matrices

The set of vectors $\boldsymbol{q}_1..\boldsymbol{q}_k$ are *orthogonal* if

$$\boldsymbol{q}_j^T\boldsymbol{q}_k = \begin{matrix} 0 & j \neq k \\ d_{jk} & j = k \end{matrix} \tag{2.47}$$

If these vectors are placed in columns of the matrix $\boldsymbol{Q}$ then

$$\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{D} \tag{2.48}$$

2.5.4 Orthonormal Matrices

The set of vectors $\boldsymbol{q}_1..\boldsymbol{q}_k$ are *orthonormal* if

$$\boldsymbol{q}_j^T \boldsymbol{q}_k = \begin{matrix} 0 & j \neq k \\ 1 & j = k \end{matrix} \tag{2.49}$$

If these vectors are placed in columns of the matrix $\boldsymbol{Q}$ then

$$\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{I} \tag{2.50}$$

Hence, the transpose equals the inverse

$$\boldsymbol{Q}^T = \boldsymbol{Q}^{-1} \tag{2.51}$$

The vectors $\boldsymbol{q}_1..\boldsymbol{q}_k$ are said to provide an *orthonormal basis*. This means that *any* vector can be written as a linear combination of the basis vectors. A trivial example is the two-dimensional cartesian coordinate system where $\boldsymbol{q}_1 = [1, 0]^T$ (the x-axis) and $\boldsymbol{q}_2 = [0, 1]^T$ (the y-axis). More generally, to represent the vector $\boldsymbol{x}$ we can write

$$\boldsymbol{x} = \tilde{x}_1\boldsymbol{q}_1 + \tilde{x}_2\boldsymbol{q}_2 + ... + \tilde{x}_d\boldsymbol{q}_d \tag{2.52}$$

To find the appropriate coefficients $\tilde{x}_k$(the co-ordinates in the new basis), multiply both sides by $\boldsymbol{q}_k^T$. Due to the orthonormality property all terms on the right disappear except one leaving

$$\tilde{x}_k = \boldsymbol{q}_k^T\boldsymbol{x} \tag{2.53}$$

The new coordinates are the projections of the data onto the basis functions (re. equation 2.45, there is no denominator since $\boldsymbol{q}_k^T\boldsymbol{q}_k = 1$). In matrix form, equation 2.52 can be written as $\boldsymbol{x} = \boldsymbol{Q}\tilde{\boldsymbol{x}}$ which therefore has the solution $\tilde{\boldsymbol{x}} = \boldsymbol{Q}^{-1}\boldsymbol{x}$. But given that $\boldsymbol{Q}^{-1} = \boldsymbol{Q}^T$ we have

$$\tilde{\boldsymbol{x}} = \boldsymbol{Q}^T\boldsymbol{x} \tag{2.54}$$

Transformation to an orthonormal basis preserves lengths. This is because [2]

$$\begin{aligned} ||\tilde{\boldsymbol{x}}|| &= ||\boldsymbol{Q}^T\boldsymbol{x}|| \\ &= (\boldsymbol{Q}^T\boldsymbol{x})^T\boldsymbol{Q}^T\boldsymbol{x} \\ &= \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{x} \\ &= \boldsymbol{x}^T\boldsymbol{x} \\ &= ||\boldsymbol{x}|| \end{aligned} \tag{2.55}$$

Similarly, inner products and therefore angles between vectors are preserved. That is

$$\begin{aligned} \tilde{\boldsymbol{x}}^T\tilde{\boldsymbol{y}} &= (\boldsymbol{Q}^T\boldsymbol{x})^T\boldsymbol{Q}^T\boldsymbol{y} \\ &= \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{y} \\ &= \boldsymbol{x}^T\boldsymbol{y} \end{aligned} \tag{2.56}$$

Therefore, transformation by an orthonormal matrix constitutes a *rotation* of the co-ordinate system.

[2] Throughout this chapter we will make extensive use of the matrix identities $(\boldsymbol{AB})^T = \boldsymbol{B}^T\boldsymbol{A}^T$ and $(\boldsymbol{AB})\boldsymbol{C} = \boldsymbol{A}(\boldsymbol{BC})$. We will also use $(\boldsymbol{AB})^{-1} = \boldsymbol{B}^{-1}\boldsymbol{A}^{-1}$.

2.6 Subspaces

A space is, for example, a set of real numbers. A subspace S is a set of points $\{x\}$ such that (i) if we take two vectors from S and add them we remain in S and (ii) if we take a vector from S and multiply by a scalar we also remain in S (S is said to be closed under addition and multiplication). An example is a 2-D plane in a 3-D space. A subspace can be defined by a basis.

2.7 Determinants

The determinant of a two-by-two matrix

$$\boldsymbol{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \tag{2.57}$$

is given by

$$\det(\boldsymbol{A}) = ad - bc \tag{2.58}$$

The determinant of a three-by-three matrix

$$\boldsymbol{A} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \tag{2.59}$$

is given by

$$\det(\boldsymbol{A}) = a \det\left(\begin{bmatrix} e & f \\ h & i \end{bmatrix}\right) - b \det\left(\begin{bmatrix} d & f \\ g & i \end{bmatrix}\right) + c \det\left(\begin{bmatrix} d & e \\ g & h \end{bmatrix}\right) \tag{2.60}$$

Determinants are important because of their properties. In particular, if two rows of a matrix are equal then the determinant is zero eg. if

$$\boldsymbol{A} = \begin{bmatrix} a & b \\ a & b \end{bmatrix} \tag{2.61}$$

then

$$\det(\boldsymbol{A}) = ab - ba = 0 \tag{2.62}$$

In this case the transformation from $\boldsymbol{x} = [x_1, x_2]^T$ to $\boldsymbol{y} = [y_1, y_2]^T$ given by

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y} \tag{2.63}$$

reduces two pieces of information (x_1 and x_2) to one piece of information

$$y = y_1 = y_2 = ax_1 + bx_2 \tag{2.64}$$

In this case it is not possible to reconstruct $\boldsymbol{x}$ from $\boldsymbol{y}$; the transformation is not invertible - the matrix $\boldsymbol{A}$ does not have an inverse and the value of the determinant provides a test for this: If $\det(\boldsymbol{A}) = 0$ the matrix $\boldsymbol{A}$ is not invertible; it is *singular*.

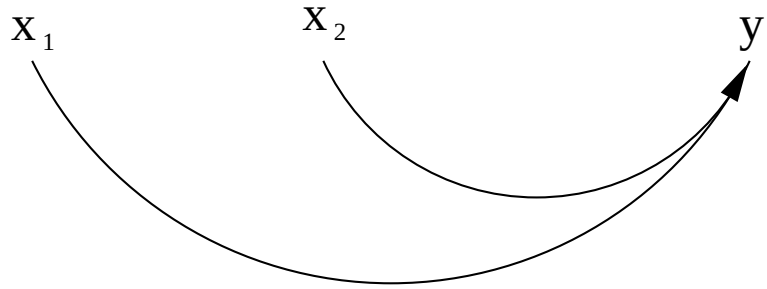


Figure 2.5: *A singular (non-invertible) transformation.*

Conversely, if $\det(\boldsymbol{A}) \neq 0$ then $\boldsymbol{A}$ *is* invertible. Other properties of the determinant are

$$\begin{aligned} \det(\boldsymbol{A}^T) &= \det(\boldsymbol{A}) \\ \det(\boldsymbol{AB}) &= \det(\boldsymbol{A})\det(\boldsymbol{B}) \\ \det(\boldsymbol{A}^{-1}) &= 1/\det(\boldsymbol{A}) \\ \det(\boldsymbol{A}) &= \prod_k a_{kk} \end{aligned} \tag{2.65}$$

Another important property of determinants is that they measure the 'volume' of a matrix. For a 3-by-3 matrix the three rows of the matrix form the edges of a cube. The determinant is the volume of this cube. For a d-by-d matrix the rows form the edges of a 'parallepiped'. Again, the determinant is the volume.

2.8 Eigenanalysis

The square matrix $\boldsymbol{A}$ has eigenvalues λ and eigenvectors $\boldsymbol{q}$ if

$$\boldsymbol{Aq} = \lambda\boldsymbol{q} \tag{2.66}$$

Therefore

$$(\boldsymbol{A} - \lambda\boldsymbol{I})\boldsymbol{q} = 0 \tag{2.67}$$

To satisfy this equation either $\boldsymbol{q} = 0$, which is uninteresting, or the matrix $\boldsymbol{A} - \lambda\boldsymbol{I}$ must reduce $\boldsymbol{q}$ to the null vector (a single point). For this to happen $\boldsymbol{A} - \lambda\boldsymbol{I}$ must be singular. Hence

$$\det(\boldsymbol{A} - \lambda\boldsymbol{I}) = 0 \tag{2.68}$$

Eigenanalysis therefore proceeds by (i) solving the above equation to find the eigenvalues λ_i and then (ii) substituting them into equation 2.66 to find the eigenvectors. For example, if

$$\boldsymbol{A} = \begin{bmatrix} 4 & -5 \\ 2 & -3 \end{bmatrix} \tag{2.69}$$

then

$$\det(\boldsymbol{A} - \lambda\boldsymbol{I}) = (4-\lambda)(-3-\lambda) - (-5)(2) = 0 \tag{2.70}$$

which can be rearranged as

$$\begin{aligned} \lambda^2 - \lambda - 2 &= 0 \\ (\lambda+1)(\lambda-2) &= 0 \end{aligned} \tag{2.71}$$

Hence the eigenvalues are $\lambda = -1$ and $\lambda = 2$. Substituting back into equation 2.66 gives an eigenvector $\boldsymbol{q}_1$ which is any multiple of $[1,1]^T$. Similarly, eigenvector $\boldsymbol{q}_2$ is any multiple of $[5,2]^T$.

We now note that the determinant of a matrix is also equal to the product of its eigenvalues

$$\det(\boldsymbol{A}) = \prod_k \lambda_k \tag{2.72}$$

We also define the *Trace* of a matrix as the sum of its diagonal elements

$$Tr(\boldsymbol{A}) = \sum_k a_{kk} \tag{2.73}$$

and note that it is also equal to the sum of the eigenvalues

$$Tr(\boldsymbol{A}) = \sum_k \lambda_k \tag{2.74}$$

Eigenanalysis applies only to *square* matrices.

2.9 Gram-Schmidt

A general class of procedures for finding eigenvectors are the *deflation methods* of which QR-decomposition and Gram-Schmidt orthogonalization are examples.

In Gram-Schmidt, we are given a set of vectors, say $\boldsymbol{a}$,$\boldsymbol{b}$ and $\boldsymbol{c}$ and we wish to find a set of corresponding orthonormal vectors which we'll call $\boldsymbol{q}_1$,$\boldsymbol{q}_2$ and $\boldsymbol{q}_3$. To start with we let

$$\boldsymbol{q}_1 = \frac{\boldsymbol{a}}{||\boldsymbol{a}||} \tag{2.75}$$

We then compute $\boldsymbol{b}'$ which is the original vector b minus the projection vector (see equation 2.46) of $\boldsymbol{b}$ onto q_1

$$\boldsymbol{b}' = \boldsymbol{b} - \boldsymbol{q}_1^T \boldsymbol{b} \boldsymbol{q}_1 \tag{2.76}$$

The second orthogonal vector is then a unit length version of b'

$$\boldsymbol{q}_2 = \frac{\boldsymbol{b}'}{||\boldsymbol{b}'||} \tag{2.77}$$

Finally, the third orthonormal vector is given by

$$\boldsymbol{q}_3 = \frac{\boldsymbol{c}'}{||\boldsymbol{c}'||} \tag{2.78}$$

where

$$\boldsymbol{c}' = \boldsymbol{c} - \boldsymbol{q}_1^T \boldsymbol{c} \boldsymbol{q}_1 - \boldsymbol{q}_2^T \boldsymbol{c} \boldsymbol{q}_2 \tag{2.79}$$

In QR-decomposition the Q terms are given by $\boldsymbol{q}_i$ and the R terms by $\boldsymbol{q}_i^T \boldsymbol{c}$.

2.9.1 Diagonalization

If we put the eigenvectors into the columns of a matrix

$$\boldsymbol{Q} = \begin{bmatrix} | & | & \cdot & | \\ | & | & \cdot & | \\ \boldsymbol{q}_1 & \boldsymbol{q}_2 & \cdot & \boldsymbol{q}_d \\ | & | & \cdot & | \\ | & | & \cdot & | \end{bmatrix} \tag{2.80}$$

then, because, $\boldsymbol{A}\boldsymbol{q}_k = \lambda_k \boldsymbol{q}_k$, we have

$$\boldsymbol{A}\boldsymbol{Q} = \begin{bmatrix} | & | & \cdot & | \\ | & | & \cdot & | \\ \lambda_1 \boldsymbol{q}_1 & \lambda_2 \boldsymbol{q}_2 & \cdot & \lambda_d \boldsymbol{q}_d \\ | & | & \cdot & | \\ | & | & \cdot & | \end{bmatrix} \tag{2.81}$$

If we put the eigenvalues into the matrix $\boldsymbol{\Lambda}$ then the above matrix can also be written as $\boldsymbol{Q}\boldsymbol{\Lambda}$. Therefore,

$$\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{\Lambda} \tag{2.82}$$

Pre-multiplying both sides by $\boldsymbol{Q}^{-1}$ gives

$$\boldsymbol{Q}^{-1}\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{\Lambda} \tag{2.83}$$

This shows that any square matrix can be converted into a diagonal form (provided it has distinct eigenvalues; see eg. [58] p. 255). Sometimes there won't be d distinct eigenvalues and sometimes they'll be complex.

2.9.2 Spectral Theorem

For any real *symmetric* matrix all the eigenvalues will be real and there will be d distinct eigenvalues and eigenvectors. The eigenvectors will be orthogonal (if the matrix is not symmetric the eigenvectors won't be orthogonal). They can be normalised and placed into the matrix $\boldsymbol{Q}$. Since $\boldsymbol{Q}$ is now orthonormal we have $\boldsymbol{Q}^{-1} = \boldsymbol{Q}^T$. Hence

$$\boldsymbol{Q}^T\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{\Lambda} \tag{2.84}$$

Pre-multiplying by $\boldsymbol{Q}$ and post-multiplying by $\boldsymbol{Q}^T$ gives

$$\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T \tag{2.85}$$

which is known as the *spectral theorem*. It says that any real, symmetric matrix can be represented as above where the columns of $\boldsymbol{Q}$ contain the eigenvectors and $\boldsymbol{\Lambda}$ is a diagonal matrix containing the eigenvalues, λ_i. Equivalently,

$$\boldsymbol{A} = \begin{bmatrix} | & | & \cdot & | \\ | & | & \cdot & | \\ \boldsymbol{q}_1 & \boldsymbol{q}_2 & \cdot & \boldsymbol{q}_d \\ | & | & \cdot & | \\ | & | & \cdot & | \end{bmatrix} \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & & \\ & & & \lambda_d \end{bmatrix} \begin{bmatrix} - & - & \boldsymbol{q}_1 & - & - \\ - & - & \boldsymbol{q}_2 & - & - \\ \cdot & \cdot & \cdot & & \cdot \\ - & - & \boldsymbol{q}_d & - & - \end{bmatrix} \tag{2.86}$$

This can also be written as a summation

$$\boldsymbol{A} = \sum_{k=1}^{d} \lambda_k \boldsymbol{q}_k \boldsymbol{q}_k^T \tag{2.87}$$

2.10 Complex Matrices

If

$$\boldsymbol{A} = \begin{bmatrix} 3+2i & 4 & 6+3i \\ -2+i & 3+2i & 7+4i \end{bmatrix} \tag{2.88}$$

then the complex transpose or Hermitian transpose is given by

$$\boldsymbol{A}^H = \begin{bmatrix} 3-2i & -2-i \\ 4 & 3-2i \\ 6-3i & 7-4i \end{bmatrix} \tag{2.89}$$

ie. each entry changes into its complex conjugate (see appendix) and we then transpose the result. Just as $\boldsymbol{A}^{-T}$ denotes the transpose of the inverse so $\boldsymbol{A}^{-H}$ denotes the Hermitian transpose of the inverse.

If $\boldsymbol{A}^H\boldsymbol{A}$ is a diagonal matrix then $\boldsymbol{A}$ is said to be a *unitary matrix*. It is the complex equivalent of an orthogonal matrix.

2.11 Quadratic Forms

The quadratic function

$$f(\boldsymbol{x}) = a_{11}x_1^2 + a_{12}x_1x_2 + a_{21}x_2x_1 + ... + a_{dd}x_d^2 \tag{2.90}$$

can be written in matrix form as

$$f(\boldsymbol{x}) = [x_1, x_2, ..., x_d] \begin{bmatrix} a_{11} & a_{12} & a_{1d} \\ a_{21} & a_{22} & a_{2d} \\ & & \\ a_{d1} & a_{d2} & a_{dd} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ . \\ x_d \end{bmatrix} \tag{2.91}$$

which is written compactly as

$$f(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \tag{2.92}$$

If $f(\boldsymbol{x}) > 0$ for any non-zero $\boldsymbol{x}$ then $\boldsymbol{A}$ is said to be positive-definite. Similarly, if $f(\boldsymbol{x}) \geq 0$ then $\boldsymbol{A}$ is positive-semi-definite.

If we substitute $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$ and $\boldsymbol{x} = \boldsymbol{Q}\boldsymbol{y}$ where $\boldsymbol{y}$ are the projections onto the eigenvectors, then we can write

$$\begin{aligned} f(\boldsymbol{x}) &= \boldsymbol{y}^T\boldsymbol{\Lambda}\boldsymbol{y} \\ &= \sum_i y_i^2\lambda_i \end{aligned} \tag{2.93}$$

Hence, for positive-definiteness we must therefore have $\lambda_i > 0$ for all i (ie. positive eigenvalues).

2.11.1 Ellipses

For 2-by-2 matrices if $\boldsymbol{A} = \boldsymbol{I}$ then we have

$$f = x_1^2 + x_2^2 \tag{2.94}$$

which is the equation of a circle with radius $\sqrt{f}$. If $\boldsymbol{A} = k\boldsymbol{I}$ we have

$$\frac{f}{k} = x_1^2 + x_2^2 \tag{2.95}$$

The radius is now $\sqrt{f/k}$. If $\boldsymbol{A} = diag([k_1, k_2])$ we have

$$f = k_1 x_1^2 + k_2 x_2^2 \tag{2.96}$$

which is the equation of an ellipse. For $k_1 > k_2$ the major axis has length $\sqrt{f/k_2}$ and the minor axis has length $\sqrt{f/k_1}$.

For a non-diagonal $\boldsymbol{A}$ we can diagonalise it using $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$. This gives

$$f = \lambda_1 \tilde{x}_1^2 + \lambda_2 \tilde{x}_2^2 \tag{2.97}$$

where the ellipse now lives in a new co-ordinate system given by the rotation $\tilde{\boldsymbol{x}} = \boldsymbol{x}^T\boldsymbol{Q}$. The major and minor axes have lengths $\sqrt{f/\lambda_2}$ and $\sqrt{f/\lambda_1}$.

Chapter 3

Multivariate Statistics

3.1 Introduction

We discuss covariance matrices, multivariate linear regression, feature selection, principal component analysis and singular value decomposition. See Chatfield's book on multivariate analysis for more details [10]. Also, a good practical introduction to the material on regression is presented by Kleinbaum et al. [32]. More details of matrix manipulations are available in Weisberg [64] and Strang has a great in-depth intro to linear algebra [58]. See also relevant material in *Numerical Recipes* [49].

3.2 Multivariate Linear Regression

For a multivariate linear data set, the dependent variable y_i is modelled as a linear combination of the input variables $\boldsymbol{x}_i$ and an error term [1]

$$y_i = \boldsymbol{x}_i \boldsymbol{w} + e_i \tag{3.1}$$

where $\boldsymbol{x}_i$ is a row vector, $\boldsymbol{w}$ is a column vector and e_i is an error. The overall goodness of fit can be assessed by the least squares cost function

$$E = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{3.2}$$

where $\hat{\boldsymbol{y}} = \boldsymbol{x}_i \boldsymbol{w}$.

[1] The error term is introduced because, very often, given a particular data set it will not be possible to find an exact linear relationship between $\boldsymbol{x}_i$ and y_i for every i. We therefore cannot directly estimate the weights as $\boldsymbol{X}^{-1}\boldsymbol{y}$.

3.2.1 Estimating the weights

The least squares cost function can be written in matrix notation as

$$E = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w})^T(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}) \tag{3.3}$$

where $\boldsymbol{X}$ is an N-by-p matrix whose rows are made up of different input vectors and $\boldsymbol{y}$ is a vector of targets. The weight vector that minimises this cost function can be calculated by setting the first derivative of the cost function to zero and solving the resulting equation.

By expanding the brackets and collecting terms (using the matrix identity $(\boldsymbol{AB})^T = \boldsymbol{B}^T\boldsymbol{A}^T$ we get

$$E = \boldsymbol{y}^T\boldsymbol{y} - 2\boldsymbol{w}\boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{w}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} \tag{3.4}$$

The derivative with respect to $\boldsymbol{w}$ is [2]

$$\frac{\partial E}{\partial \boldsymbol{w}} = -2\boldsymbol{X}^T\boldsymbol{y} - 2\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} \tag{3.5}$$

Equating this derivative to zero gives

$$(\boldsymbol{X}^T\boldsymbol{X})\boldsymbol{w} = \boldsymbol{X}^T\boldsymbol{y} \tag{3.6}$$

which, in regression analysis, is known as the 'normal equation'. Hence,

$$\hat{\boldsymbol{w}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{3.7}$$

This is the general solution for multivariate linear regression [3]. It is a unique minimum of the least squares error function (ie. this is the only solution).

Once the weights have been estimated we can then estimate the error or noise variance from

$$\sigma_e^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{3.8}$$

3.2.2 Understanding the solution

If the inputs are zero mean then the input covariance matrix multiplied by N-1 is

$$\boldsymbol{C}_x = \boldsymbol{X}^T\boldsymbol{X} \tag{3.9}$$

The weights can therefore be written as

$$\hat{\boldsymbol{w}} = \boldsymbol{C}_x^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{3.10}$$

ie. the inverse covariance matrix times the inner products of the inputs with the output (the ith weight will involve the inner product of the ith input with the output).

[2] From matrix calculus [37] we know that the derivative of $\boldsymbol{c}^T\boldsymbol{B}\boldsymbol{c}$ with respect to $\boldsymbol{c}$ is $(\boldsymbol{B}^T + \boldsymbol{B})\boldsymbol{c}$. Also we note that $\boldsymbol{X}^T\boldsymbol{X}$ is symmetric.

[3] In practice we can use the equivalent expression $\hat{\boldsymbol{w}} = \boldsymbol{X}^{+1}\boldsymbol{y}$ where $\boldsymbol{X}^{+1}$ is the pseudo-inverse [58]. This method is related to Singular Value Decomposition and is discussed later.

Single input

For a single input $\boldsymbol{C}_x^{-1} = 1/(N-1)\sigma_{x_1}^2$ and $\boldsymbol{X}^T\boldsymbol{y} = (N-1)\sigma_{x_1 y}$. Hence

$$\hat{w}_1 = \frac{\sigma_{x_1 y}}{\sigma_{x_1}^2} \tag{3.11}$$

This is *exactly* the same as the estimate for the slope in linear regression (first lecture). This is re-assuring.

Uncorrelated inputs

For two uncorrelated inputs

$$\boldsymbol{C}_x^{-1} = \begin{bmatrix} \frac{1}{(N-1)\sigma_{x_1}^2} & 0 \\ 0 & \frac{1}{(N-1)\sigma_{x_2}^2} \end{bmatrix} \tag{3.12}$$

We also have

$$\boldsymbol{X}^T\boldsymbol{y} = \begin{bmatrix} (N-1)\sigma_{x_1,y} \\ (N-1)\sigma_{x_2,y} \end{bmatrix} \tag{3.13}$$

The two weights are therefore

$$\begin{aligned} \hat{w}_1 &= \frac{\sigma_{x_1 y}}{\sigma_{x_1}^2} \\ \hat{w}_2 &= \frac{\sigma_{x_2 y}}{\sigma_{x_2}^2} \end{aligned} \tag{3.14}$$

Again, these solutions are the same as for the univariate linear regression case.

General case

If the inputs are correlated then a coupling is introduced in the estimates of the weights; weight 1 becomes a function of $\sigma_{x_2 y}$ as well as $\sigma_{x_1 y}$

$$\hat{\boldsymbol{w}} = \begin{bmatrix} \sigma_{x_1}^2 & \sigma_{x_1 x_2} \\ \sigma_{x_1 x_2} & \sigma_{x_2}^2 \end{bmatrix}^{-1} \begin{bmatrix} \sigma_{x_1,y} \\ \sigma_{x_2,y} \end{bmatrix} \tag{3.15}$$

3.2.3 Feature selection

Some of the inputs in a linear regression model may be very useful in predicting the output. Others, not so. So how do we find which inputs or *features* are useful ? This problem is known as feature selection.

The problem is tackled by looking at the coefficients of each input (ie. the weights) and seeing if they are significantly non-zero. The procedure is identical to that described for univariate linear regression.

The only added difficulty is that we have more inputs and more weights, but the procedure is basically the same. Firstly, we have to estimate the variance on each weight. This is done in the next section. We then compare each weight to zero using a t-test.

The weight covariance matrix

Different instantiations of target noise will generate different estimated weight vectors according to equation 3.7. For the case of Gaussian noise we do not actually have to compute the weights on many instantiations of the target noise and then compute the sample covariance [4]; the corresponding weight covariance matrix is given by the equation

$$\boldsymbol{\Sigma} = Var((\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}) \tag{3.16}$$

In the appendix we show that this can be evaluated as

$$\boldsymbol{\Sigma} = \sigma_e^2(\boldsymbol{X}^T\boldsymbol{X})^{-1} \tag{3.17}$$

The correlation in the inputs introduces a correlation in the weights; for uncorrelated inputs the weights will be uncorrelated. The variance of the jth weight, w_j, is then given by the jth diagonal entry in the covariance matrix

$$\sigma_{w_j}^2 = \boldsymbol{\Sigma}_{jj} \tag{3.18}$$

To see if a weight is significantly non-zero we then compute $CDF_t(t)$ (the cumulative density function; see earlier lecture) where $t = w_j/\sigma_{w_j}$ and if it is above some threshold, say $p = 0.05$, the corresponding feature is removed.

Note that this procedure, which is based on a t-test, is exactly equivalent to a similar procedure based on a partial F-test (see, for example, [32] page 128).

If we do remove a weight then we must recompute all the other weights (and variances) *before* deciding whether or not the other weights are significantly non-zero. This usually proceeds in a stepwise manner where we start with a large number of features and reduce them as necessary (*stepwise backward selection*) or gradually build up the number of features (*stepwise forward selection*) [32].

Note that, if the weights were uncorrelated we could do feature selection in a single step; we would not have to recompute weight values after each weight removal. This provides one motivation for the use of orthogonal transforms in which the weights *are* uncorrelated. Such transforms include Fourier and Wavelet transforms as we shall see in later lectures.

[4] But this type of procedure is the basis of bootstrap estimates of parameter variances. See [17].

3.2.4 Example

Suppose we wish to predict a time series x_3 from two other time series x_1 and x_2. We can do this with the following regression model [5]

$$x_3 = w_0 + w_1 x_1 + w_2 x_2 \tag{3.19}$$

and the weights can be found using the previous formulae. To cope with the constant, w_0, we augment the $\boldsymbol{X}$ vector with an additional column of 1's.

We analyse data having covariance matrix $\boldsymbol{C}_1$ and mean vector $\boldsymbol{m}_1$ (see equations 2.15 and 2.14 in an earlier lecture). $N = 50$ data points were generated and are shown in Figure 3.1. The weights were then estimated from equation 3.7 as

$$\begin{aligned} \hat{\boldsymbol{w}} &= [w_1, w_2, w_0]^T \\ &= [1.7906, -0.0554, 0.6293]^T \end{aligned} \tag{3.20}$$

Note that w_1 is much bigger than w_2. The weight covariance matrix was estimated from equation B.27 as

$$\boldsymbol{\Sigma} = \begin{bmatrix} 0.0267 & 0.0041 & -0.4197 \\ 0.0041 & 0.0506 & -0.9174 \\ -0.4197 & -0.9174 & 21.2066 \end{bmatrix} \tag{3.21}$$

giving $\sigma_{w_1} = 0.1634$ and $\sigma_{w_2} = 0.2249$. The corresponding t-statistics are $t_1 = 10.96$ and $t_2 = -0.2464$ giving p-values of 10^{-15} and 0.4032. This indicates that the first weight is significantly different from zero but the second weight is not ie. x_1 is a good predictor of x_3 but x_2 is not. We can therefore remove x_2 from our regression model.

Question: But what does linear regression tell us about the data that the correlation/covariance matrix does'nt ? *Answer:* Partial correlations.

3.2.5 Partial Correlation

Remember (see eg. equation 1.36 from lecture 1), the square of the correlation coefficient between two variables x_1 and y is given by

$$r^2_{x_1 y} = \frac{\sigma_y^2 - \sigma_e^2(x_1)}{\sigma_y^2} \tag{3.22}$$

where $\sigma_e^2(x_1)$ is the variance of the errors from using a linear regression model based on x_1 to predict y. Writing $\sigma_y^2 = \sigma_e^2(0)$, ie. the error with no predictive variables

$$r^2_{x_1 y} = \frac{\sigma_e^2(0) - \sigma_e^2(x_1)}{\sigma_e^2(0)} \tag{3.23}$$

[5] Strictly, we can only apply this model if the samples *within* each time series are independent (see later). To make them independent we can randomize the time index thus removing any correlation between lagged samples. We therefore end up with a random variables rather than time series.

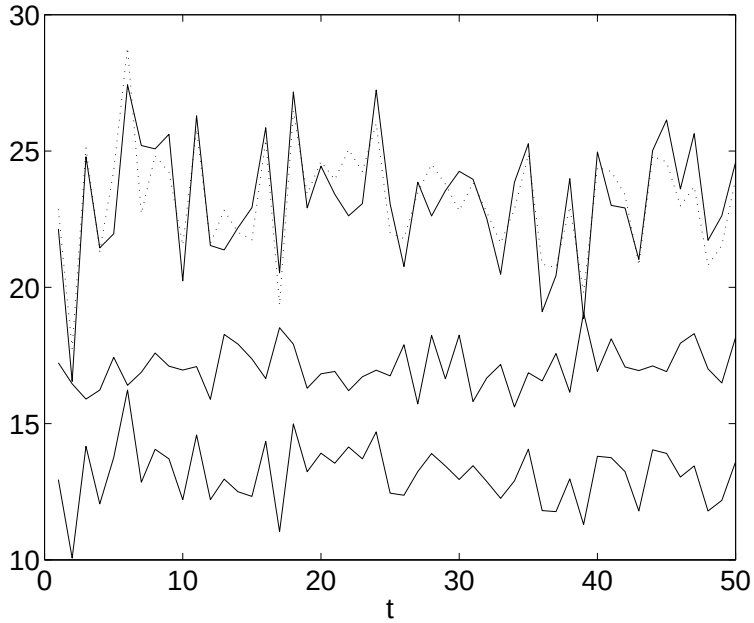


Figure 3.1: *Three time series having the correlation matrix* $\boldsymbol{C}_1$ *and mean vector* $\boldsymbol{m}_1$ *shown in the text. The dotted line shows the value of the third time series as predicted from the other two using a regression model.*

When we have a second predictive variable x_2, the square of the *partial correlation* between x_2 and y is defined as

$$r^2_{x_2 y|x_1} = \frac{\sigma_e^2(x_1) - \sigma_e^2(x_1, x_2)}{\sigma_e^2(x_1)} \tag{3.24}$$

where $\sigma_e^2(x_1, x_2)$ is the variance of the errors from the regression model based on x_1 and x_2. It's the extra proportion of variance in y explained by x_2. It's different to $r^2_{x_2 y}$ because x_2 may be correlated to x_1 which itself explains some of the variance in y. After *controlling* for this, the resulting proportionate reduction in variance is given by $r^2_{x_2 y|x_1}$. More generally, we can define pth order partial correlations which are the correlations between two variables after controlling for p variables.

The sign of the partial correlation is given by the sign of the corresponding regression coefficient.

Relation to regression coefficients

Partial correlations are to regression coefficients what the correlation is to the slope in univariate linear regression. If the partial correlation is significantly non-zero then the corresponding regression coefficient will also be. And vice-versa.

3.3 Principal Component Analysis

Given a set of data vectors $\{\boldsymbol{x}_n\}$ we can construct a covariance matrix

$$\boldsymbol{C} = \frac{1}{N}\sum_n (\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^T \tag{3.25}$$

or, if we construct a matrix $\boldsymbol{X}$ with rows equal to $\boldsymbol{x}_n - \bar{\boldsymbol{x}}$ then

$$\boldsymbol{C} = \frac{1}{N}\boldsymbol{X}^T\boldsymbol{X} \tag{3.26}$$

Because covariance matrices are real and symmetric we can apply the spectral theorem

$$\boldsymbol{C} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T \tag{3.27}$$

If the eigenvectors (columns of $\boldsymbol{Q}$) are normalised to unit length, they constitute an orthonormal basis. If the eigenvalues are then ordered in magnitude such that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_d$ then the decomposition is known as Principal Component Analysis (PCA). The projection of a data point $\boldsymbol{x}_n$ onto the principal components is

$$\boldsymbol{y}_n = \boldsymbol{Q}^T\boldsymbol{x}_n \tag{3.28}$$

The mean projection is

$$\bar{\boldsymbol{y}} = \boldsymbol{Q}^T\bar{\boldsymbol{x}} \tag{3.29}$$

The covariance of the projections is given by the matrix

$$\boldsymbol{C}_y = \frac{1}{N}\sum_n (\boldsymbol{y}_n - \bar{\boldsymbol{y}})(\boldsymbol{y}_n - \bar{\boldsymbol{y}})^T \tag{3.30}$$

Substituting in the previous two expressions gives

$$\begin{aligned} \boldsymbol{C}_y &= \frac{1}{N}\sum_n \boldsymbol{Q}^T(\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^T\boldsymbol{Q} \\ &= \boldsymbol{Q}^T\boldsymbol{C}\boldsymbol{Q} \\ &= \boldsymbol{\Lambda} \end{aligned} \tag{3.31}$$

where $\boldsymbol{\Lambda}$ is the diagonal eigenvalue matrix with entries λ_k ($\sigma_k^2 = \lambda_k$). This shows that the variance of the kth projection is given by the kth eigenvalue. Moreover, it says that the projections are uncorrelated. PCA may therefore be viewed as a linear transform

$$\boldsymbol{y} = \boldsymbol{Q}^T\boldsymbol{x} \tag{3.32}$$

which produces uncorrelated data.

3.3.1 The Multivariate Gaussian Density

In d dimensions the general multivariate normal probability density can be written

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{C}|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \bar{\boldsymbol{x}})^T C^{-1}(\boldsymbol{x} - \bar{\boldsymbol{x}})\right) \tag{3.33}$$

where the mean $\bar{\boldsymbol{x}}$ is a d-dimensional vector, $\boldsymbol{C}$ is a $d \times d$ covariance matrix, and $|\boldsymbol{C}|$ denotes the determinant of $\boldsymbol{C}$. Because the determinant of a matrix is the product of its eigenvalues then for covariance matrices, where the eigenvalues correspond to variances, the determinant is a single number which represents the total volume of variance. The quantity

$$M(\boldsymbol{x}) = (\boldsymbol{x} - \bar{\boldsymbol{x}})^T C^{-1} (\boldsymbol{x} - \bar{\boldsymbol{x}}) \tag{3.34}$$

which appears in the exponent is called the *Mahalanobis distance* from $\boldsymbol{x}$ to $\bar{\boldsymbol{x}}$. This is the equation for an ellipse (see earlier). The directions of the axes are given by the principal components and the lengths are given by $\sigma_i M(\boldsymbol{x})$ where σ_i is the standard deviation of the data in the ith direction (see earlier section on quadratic forms and note that $\lambda_i = \sigma_i^2$). We can therefore map a given probability $p(\boldsymbol{x})$ to a Mahalanobis distance (using equation E.9) and from that plot the ellipse axes. See the figure in the appendix.

3.3.2 Dimensionality Reduction

Given that the eigenvalues in PCA are ordered and that they correspond to the variance of the data in orthogonal directions then it would seem plausible that a reasonable data reconstruction could be obtained from just a few of the larger components and this is indeed the case.

If we retain only a subset $M < d$ of the basis vectors then a data point can be reconstructed as

$$\hat{\boldsymbol{x}}_n = \sum_{k=1}^{M} w_k^n \boldsymbol{q}_k + \sum_{k=M+1}^{d} b_k \boldsymbol{q}_k \tag{3.35}$$

where the b_k are constants (they don't depend on n) and, as we have seen, $w_k^n = \boldsymbol{q}_k^T \boldsymbol{x}_n$. If we keep only the projections w_k^n and the associated eigenvectors $\boldsymbol{q}_k$ we have reduced the dimension of our data set from d to M. Now, given that the actual data point can be written as

$$\boldsymbol{x}_n = \sum_{k=1}^{d} w_k^n \boldsymbol{q}_k \tag{3.36}$$

where the sum is over all d components (not just M) then the reconstruction error is

$$\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n = \sum_{k=M+1}^{d} (w_k^n - b_k) \boldsymbol{q}_k \tag{3.37}$$

It is the cost of replacing the variable w_k^n by a constant b_k. The reconstruction error averaged over the whole data set is

$$\begin{aligned} E_M &= \frac{1}{N} \sum_{n=1}^{N} ||\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n|| \\ &= \frac{1}{N} \sum_{n=1}^{N} \sum_{k=M+1}^{d} (w_k^n - b_k)^2 \end{aligned} \tag{3.38}$$

where the $\boldsymbol{q}_k$'s disappear because $\boldsymbol{q}_k^T \boldsymbol{q}_k = 1$. We can minimise E_M by setting

$$\begin{aligned} b_k &= \frac{1}{N} \sum_{n=1}^{N} w_k^n \\ &= \boldsymbol{q}_k^T \bar{\boldsymbol{x}} \end{aligned} \tag{3.39}$$

which is the mean projection in direction $\boldsymbol{q}_k$. The error is therefore

$$\begin{aligned} E_M &= \frac{1}{N} \sum_{n=1}^{N} \sum_{k=M+1}^{d} \left[\boldsymbol{q}_k^T (\boldsymbol{x}_n - \bar{\boldsymbol{x}}) \right]^2 \\ &= \frac{N}{N} \sum_{k=M+1}^{d} \boldsymbol{q}_k^T \boldsymbol{C} \boldsymbol{q}_k \\ &= \sum_{k=M+1}^{d} \lambda_k \end{aligned} \tag{3.40}$$

The reconstruction error is therefore minimised, for a given M, by throwing away the $d - M$ smallest components, as you would expect. The corresponding error is just the sum of the corresponding eigenvalues.

3.3.3 Singular Value Decomposition

The eigenvalue-eigenvector factorisation (see equation 2.85)

$$\boldsymbol{A} = \boldsymbol{Q} \boldsymbol{\Lambda} \boldsymbol{Q}^T \tag{3.41}$$

applies to square symmetric matrices only. There is an equivalent factorisation for rectangular matrices, having N rows and d columns, called Singular Value Decomposition (SVD)

$$\boldsymbol{A} = \boldsymbol{Q}_1 \boldsymbol{D} \boldsymbol{Q}_2^T \tag{3.42}$$

where $\boldsymbol{Q_1}$ is an orthonormal N-by-N matrix, $\boldsymbol{Q_2}$ is an orthonormal d-by-d matrix, $\boldsymbol{D}$ is a diagonal matrix of dimension N-by-d and the kth diagonal entry in $\boldsymbol{D}$ is known as the kth singular value, σ_k.

If we substitute the SVD of $\boldsymbol{A}$ into $\boldsymbol{A}^T \boldsymbol{A}$, after some rearranging, we get

$$\boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{Q}_2 \boldsymbol{D}^T \boldsymbol{D} \boldsymbol{Q}_2^T \tag{3.43}$$

which is of the form $\boldsymbol{A} = \boldsymbol{Q} \boldsymbol{\Lambda} \boldsymbol{Q}^T$ where $\boldsymbol{Q} = \boldsymbol{Q}_2$ and $\boldsymbol{\Lambda} = \boldsymbol{D}^T \boldsymbol{D}$. This shows that the columns of $\boldsymbol{Q}_2$ contain the eigenvectors of $\boldsymbol{A}^T \boldsymbol{A}$ and that $\boldsymbol{D}$ contains the square roots of the corresponding eigenvalues. Similarly, by substituting the SVD of $\boldsymbol{A}$ into $\boldsymbol{A} \boldsymbol{A}^T$ we can show that the columns of $\boldsymbol{Q}_1$ are the eigenvectors of $\boldsymbol{A} \boldsymbol{A}^T$.

Relation to PCA

Given a data matrix $\boldsymbol{X}$ constructed as before (see PCA section), except that the matrix is scaled by a normalisation factor $\sqrt{1/N}$, then $\boldsymbol{X}^T\boldsymbol{X}$ is equivalent to the covariance matrix $\boldsymbol{C}$. If we therefore decompose $\boldsymbol{X}$ using SVD, the principal components will apear in $\boldsymbol{Q}_2$ and the square roots of the corresponding eigenvalues will appear in $\boldsymbol{D}$.

Therefore we can implement PCA in one of two ways (i) compute the covariance matrix and perform an eigendecomposition or (ii) use SVD directly on the (normalised) data matrix.

The Pseudo-Inverse

Given the SVD of a matrix

$$\boldsymbol{A} = \boldsymbol{Q}_1\boldsymbol{D}\boldsymbol{Q}_2^T \tag{3.44}$$

the *Pseudo-Inverse* of $\boldsymbol{A}$ is defined as

$$\boldsymbol{A}^+ = \boldsymbol{Q}_2\boldsymbol{D}^+\boldsymbol{Q}_1^T \tag{3.45}$$

where $\boldsymbol{D}^+$ is a d-by-N matrix with diagonal entries $1/\sigma_1, 1/\sigma_2, ..., 1/\sigma_d$. The matrix $\boldsymbol{D}^+$ can be computed as

$$\boldsymbol{D}^+ = (\boldsymbol{D}^T\boldsymbol{D})^{-1}\boldsymbol{D}^T \tag{3.46}$$

The Pseudo-Inverse is used in the solution of the multivariate linear regression problem (see equation 3.7)

$$\hat{\boldsymbol{w}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{3.47}$$

We can substitute the SVD for $\boldsymbol{X}$ into the above expression in a series of steps to give

$$\boldsymbol{X}^T\boldsymbol{X} = \boldsymbol{Q}_2\boldsymbol{D}^T\boldsymbol{D}\boldsymbol{Q}_2^T \tag{3.48}$$

The inverse is

$$(\boldsymbol{X}^T\boldsymbol{X})^{-1} = \boldsymbol{Q}_2(\boldsymbol{D}^T\boldsymbol{D})^{-1}\boldsymbol{Q}_2^T \tag{3.49}$$

Hence

$$(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T = \boldsymbol{Q}_2(\boldsymbol{D}^T\boldsymbol{D})^{-1}\boldsymbol{D}^T\boldsymbol{Q}_1^T \tag{3.50}$$

Substituting for $\boldsymbol{D}^+$ gives

$$\begin{aligned}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T &= \boldsymbol{Q}_2\boldsymbol{D}^+\boldsymbol{Q}_1^T \\ &= \boldsymbol{X}^+\end{aligned} \tag{3.51}$$

Therefore, the linear regression weights can be computed by projecting the targets onto the Pseudo-Inverse of the input data matrix

$$\hat{\boldsymbol{w}} = \boldsymbol{X}^+\boldsymbol{y} \tag{3.52}$$

Chapter 4

Information Theory

4.1 Introduction

This lecture covers entropy, joint entropy, mutual information and minimum description length. See the texts by Cover [12] and Mackay [36] for a more comprehensive treatment.

4.2 Measures of Information

Information on a computer is represented by binary bit strings. Decimal numbers can be represented using the following encoding. The position of the binary digit

Bit 1 ($2^3 = 8$)	Bit 2 ($2^2 = 4$)	Bit 3 ($2^1 = 2$)	Bit 4 ($2^0 = 1$)	Decimal
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
.	.	.	.	.
.	.	.	.	.
0	1	1	1	14
1	1	1	1	15

Table 4.1: *Binary encoding*

indicates its decimal equivalent such that if there are N bits the ith bit represents the decimal number 2^{N-i}. Bit 1 is referred to as the most significant bit and bit N as the least significant bit. To encode M different messages requires $\log_2 M$ bits.

4.3 Entropy

The table below shows the probability of occurrence $p(x_i)$ (to two decimal places) of selected letters x_i in the English alphabet. These statistics were taken from Mackay's book on Information Theory [36]. The table also shows the *information content* of a

x_i	$p(x_i)$	$h(x_i)$
a	0.06	4.1
e	0.09	3.5
j	0.00	10.7
q	0.01	10.3
t	0.07	3.8
z	0.00	10.4

Table 4.2: *Probability and Information content of letters*

letter

$$h(x_i) = \log \frac{1}{p(x_i)} \tag{4.1}$$

which is a measure of *surprise*; if we had to guess what a randomly chosen letter of the English alphabet was going to be, we'd say it was an A, E, T or other frequently occuring letter. If it turned out to be a Z we'd be surprised. The letter E is so common that it is unusual to find a sentence without one. An exception is the 267 page novel 'Gadsby' by Ernest Vincent Wright in which the author deliberately makes no use of the letter E (from Cover's book on Information Theory [12]). The *entropy* is the average information content

$$H(x) = \sum_{i=1}^{M} p(x_i)h(x_i) \tag{4.2}$$

where M is the number of discrete values that x_i can take. It is usually written as

$$H(x) = -\sum_{i=1}^{M} p(x_i) \log p(x_i) \tag{4.3}$$

with the convention that $0 \log 1/0 = 0$. Entropy measures uncertainty.

Entropy is maximised for a uniform distribution $p(x_i) = 1/M$. The resulting entropy is $H(x) = log_2 M$ which is the number of binary bits required to represent M different messages (first section). For $M = 2$, for example, the maximum entropy distribution is given by $p(x_1) = p(x_2) = 0.5$ (eg. an unbiased coin; biased coins have lower entropy).

The entropy of letters in the English language is 4.11 bits [12] (which is less than $log_2 26 = 4.7$ bits). This is however, the information content due to considering just the probability of occurence of letters. But, in language, our expectation of what the next letter will be is determined by what the previous letters have been. To measure this we need the concept of joint entropy. Because $H(x)$ is the entropy of a 1-dimensional variable it is sometimes called the scalar entropy, to differentiate it from the joint entropy.

4.4 Joint Entropy

Table 2 shows the probability of occurence (to three decimal places) of selected pairs of letters x_i and y_i where x_i is followed by y_i. This is called the joint probability $p(x_i, y_i)$. The table also shows the joint information content

x_i	y_j	$p(x_i, y_j)$	$h(x_i, y_j)$
t	h	0.037	4.76
t	s	0.000	13.29
t	r	0.012	6.38

Table 4.3: *Probability and Information content of pairs of letters*

$$h(x_i, y_j) = \log \frac{1}{p(x_i, y_j)} \tag{4.4}$$

The average joint information content is given by the *joint entropy*

$$H(x, y) = -\sum_{i=1}^{M} \sum_{j=1}^{M} p(x_i, y_j) \log p(x_i, y_j) \tag{4.5}$$

If we fix x to, say x_i then the probability of y taking on a particular value, say y_j, is given by the *conditional probability*

$$p(y = y_j | x = x_i) = \frac{p(x = x_i, y = y_j)}{p(x = x_i)} \tag{4.6}$$

For example, if $x_i = t$ and $y_j = h$ then the joint probability $p(x_i, y_j)$ is just the probability of occurrence of the pair (which from table 2 is 0.037). The conditional probability $p(y_j|x_i)$, however, says that, given we've seen the letter t, what's the probability that the next letter will be h (which from tables 1 and 2 is $0.037/0.07 = 0.53$). Re-arranging the above relationship (and dropping the $y = y_j$ notation) gives

$$p(x, y) = p(y|x)p(x) \tag{4.7}$$

Now if y does *not* depend on x then $p(y|x) = p(y)$. Hence, for independent variables, we have

$$p(x, y) = p(y)p(x) \tag{4.8}$$

This means that, for independent variables, the joint entropy is the sum of the individual (or *scalar* entropies)

$$H(x, y) = H(x) + H(y) \tag{4.9}$$

Consecutive letters in the English language are not independent (except either after or during a bout of serious drinking). If we take into account the statistical dependence on the previous letter, the entropy of English reduces to 3.67 bits per letter (from 4.11). If we look at the statistics of not just pairs, but triplets and quadruplets of letters or at the statistics of words then it is possible to calculate the entropy more accurately; as more and more contextual structure is taken into account the estimates of entropy reduce. See Cover's book ([12] page 133) for more details.

4.5 Relative Entropy

The *relative entropy* or *Kullback-Liebler Divergence* between a distribution $q(x)$ and a distribution $p(x)$ is defined as

$$D[q||p] = \sum_x q(x) \log \frac{q(x)}{p(x)} \tag{4.10}$$

Jensen's inequality states that for any convex function [1] $f(x)$ and set of M positive coefficients $\{\lambda_j\}$ which sum to one

$$f(\sum_{j=1}^{M} \lambda_j x_j) \geq \sum_{j=1}^{M} \lambda_j f(x_j) \tag{4.11}$$

A sketch of a proof of this is given in Bishop ([3], page 75). Using this inequality we can show that

$$\begin{aligned} -D[q||p] &= \sum_x q(x) \log \frac{p(x)}{q(x)} \\ &\leq \log \sum_x p(x) \\ &\leq \log 1 \end{aligned} \tag{4.12}$$

Hence

$$D[q||p] \geq 0 \tag{4.13}$$

The KL-divergence will appear again in the discussion of the EM algorithm and Variational Bayesian learning (see later lectures).

4.6 Mutual Information

The *mutual information* is defined [12] as the relative entropy between the joint distribution and the product of individual distributions

$$I(x;y) = D[p(X,Y)||p(X)p(Y)] \tag{4.14}$$

Substuting these distributions into 4.10 allows us to express the mutual information as the difference between the sum of the individual entropies and the joint entropy

$$I(x;y) = H(x) + H(y) - H(x,y) \tag{4.15}$$

Therefore if x and y are independent the mutual information is zero. More generally, $I(x;y)$ is a measure of the *dependence* between variables and this dependence will be captured if the underlying relationship is linear *or* nonlinear. This is to be contrasted with Pearsons correlation coefficient, which measures only linear correlation (see first lecture).

[1] A convex function has a negative second derivative.

4.7 Minimum Description Length

Given that a variable has a determinisitic component and a random component the *complexity* of that variable can be defined as the length of a concise description of that variables regularities [19].

This definition has the merit that both random data and highly regular data will have a low complexity and so we have a correspondence with our everyday notion of complexity [2]

The length of a description can be measured by the number of binary bits required to encode it. If the probability of a set of measurements D is given by $p(D|\boldsymbol{\theta})$ where $\boldsymbol{\theta}$ are the parameters of a probabilistic model then the minimum length of a code for representing D is, from Shannon's coding theorem [12], the same as the information content of that data under the model (see eg. equation 4.1)

$$L = -\log p(D|\boldsymbol{\theta}) \tag{4.16}$$

However, for the recevier to decode the message they will need to know the parameters $\boldsymbol{\theta}$ which, being real numbers are encoded by truncating each to a finite precision $\Delta\theta$. We need a total of $-k \log \Delta\theta$ bits to encode the This gives

$$L_{tx} = -\log p(D|\boldsymbol{\theta}) - k \log \Delta\theta \tag{4.17}$$

The optimal precision can be found as follows. First, we expand the negative log-likelihood (ie. the error) using a Taylor series about the Maximum Likelihood (ML) solution $\hat{\boldsymbol{\theta}}$. This gives

$$L_{tx} = -\log p(D|\hat{\boldsymbol{\theta}}) + \frac{1}{2}\Delta\boldsymbol{\theta}^T H \Delta\boldsymbol{\theta} - k \log \Delta\theta \tag{4.18}$$

where $\Delta\boldsymbol{\theta} = \boldsymbol{\theta} - \hat{\boldsymbol{\theta}}$ and H is the Hessian of the error which is identical to the inverse covariance matrix (the first order term in the Taylor series disappears as the error gradient is zero at the ML solution). The derivative is

$$\frac{\partial L_{tx}}{\partial \Delta\theta} = H\Delta\boldsymbol{\theta} - \frac{k}{\Delta\theta} \tag{4.19}$$

If the covariance matrix is diagonal (and therefore the Hessian is diagonal) then, for the case of linear regression (see equation 1.47) the diagonal elements are

$$h_i = \frac{N\sigma_{x_i}^2}{\sigma_e^2} \tag{4.20}$$

where σ_e^2 is the variance of the errors and $\sigma_{x_i}^2$ is the variance of the ith input. More generally, eg. nonlinear regression, this last variance will be replaced with the variance

[2] This is not the case, however, with measures such as the Algorithm Information Content (AIC) or Entropy as these will be high even for purely random data.

of the derivative of the output wrt. the ith parameter. But the dependence on N remains. Setting the above derivative to zero therefore gives us

$$(\Delta\theta)^2 = \frac{1}{N} \times constant \tag{4.21}$$

where the constant depends on the variance terms (when we come to take logs of $\Delta\theta$ this constant becomes an additive term that does'nt scale with either the number of data points or the number of parameters in the model; we can therefore ignore it). The *Minimum Description Length (MDL)* is therefore given by

$$MDL(k) = -\log p(D|\boldsymbol{\theta}) + \frac{k}{2}\log N \tag{4.22}$$

This may be minimised over the number of parameters k to get the optimal model complexity.

For a linear regression model

$$-\log p(D|\boldsymbol{\theta}) = \frac{N}{2}\log \sigma_e^2 \tag{4.23}$$

Therefore

$$MDL_{Linear}(k) = \frac{N}{2}\log \sigma_e^2 + \frac{k}{2}\log N \tag{4.24}$$

which is seen to consist of an accuracy term and a complexity term. This criterion can be used to select the optimal number of input variables and therefore offers a solution to the bias-variance dilemma (see lecture 1). In later lectures the MDL criterion will be used in autoregressive and wavelet models.

The MDL complexity measure can be further refined by integrating out the dependence on $\boldsymbol{\theta}$ altogether. The resulting measure is known as the *stochastic complexity* [54]

$$I(k) = -\log p(D|k) \tag{4.25}$$

where

$$p(D|k) = \int p(D|\boldsymbol{\theta}, k)p(\boldsymbol{\theta})d\boldsymbol{\theta} \tag{4.26}$$

In Bayesian statistics this quantity is known as the 'marginal likelihood' or 'evidence'. The stochastic complexity measure is thus equivalent (after taking negative logs) to the Bayesian model order selection criterion (see later). See Bishop ([3], page 429) for a further discussion of this relationship.

Chapter 5

Fourier methods

5.1 Introduction

Of the many books on Fourier methods those by Chatfield [11], Proakis and Manolakis [51] and Bloomfield [4] are particularly good.

5.2 Sinewaves and Samples

Sines and cosines can be understood in terms of the vertical and horizontal displacement of a fixed point on a rotating wheel; the wheel has unit length and rotates *anti-clockwise*. The angle round the wheel is measured in degrees or radians ($0 - 2\pi$; for unit radius circles the circumference is 2π, radians tell us how much of the circumference we've got). If we go round the wheel a whole number of times we end up in the same place, eg.$\cos 4\pi = \cos 2\pi = \cos 0 = 1$. Frequency, f, is the number of times round the wheel per second. Therefore, given $x = \cos(2\pi ft)$, $x = 1$ at $t = 1/f, 2/f$ etc. For $x = \cos(2\pi ft + \Phi)$ we get a head start (lead) of Φ radians. Negative frequencies may be viewed as a wheel rotating *clockwise* instead of anti-clockwise.

If we assume we have samples of the signal every T_s seconds and in total we have N such samples then T_s is known as the sampling period and $F_s = 1/T_s$ is the sampling frequency in Hertz (Hz) (samples per second). The nth sample occurs at time $t[n] = nT_s = n/F_s$. The cosine of sampled data can be written

$$x[n] = \cos(2\pi ft[n]) \tag{5.1}$$

When dealing with sampled signals it is important to note that some frequencies become indistinguishable from others; at a sampling frequency F_s the only *unique* frequencies are in the range 0 to $(F_s/2)$Hz. Any frequencies outside this range become *aliases* of one of the unique frequencies.

For example, if we sample at 8Hz then a -6Hz signal becomes indistinguishable from

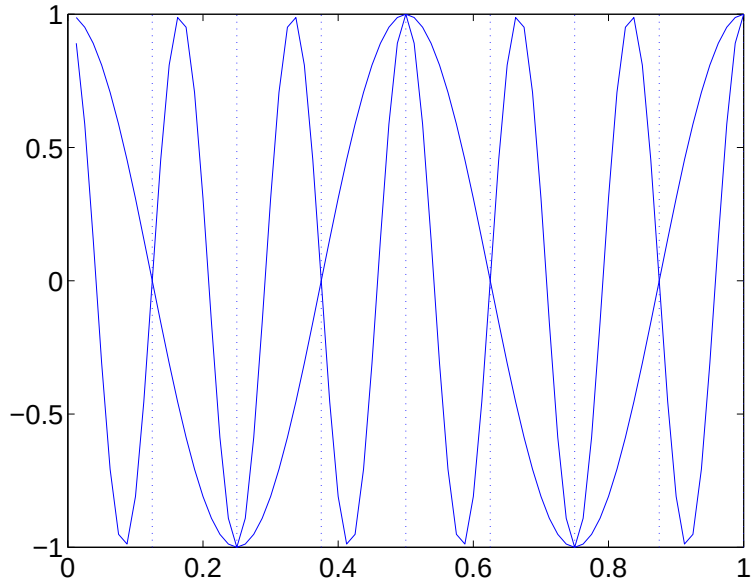


Figure 5.1: **Aliases** *The figure shows a 2Hz cosine wave and a -6Hz cosine wave as solid curves. At sampling times given by the dotted lines, which correspond to a sampling frequency of 8Hz, the* $-6Hz$ *signal is an alias of the 2Hz signal. Other aliases are given by equation 5.2.*

a 2Hz signal. This is shown in figure 5.1. More generally, if f_0 is a unique frequency then its aliases have frequencies given by

$$f = f_0 + kF_s \tag{5.2}$$

where k is any positive or negative integer, eg. for $f_0 = 2$ and $F_s = 8$ the two lowest frequency aliases, given by $k = -1$ and $k = 1$, are -6Hz and 10Hz.

Because of *aliasing* we must be careful when we interpret the results of spectral analysis. This is discussed more at the end of the lecture.

5.3 Sinusoidal models

If our time series has a periodic component in it we might think about modelling it with the equation

$$x[n] = R_0 + Rcos(2\pi f t[n] + \Phi) + e[n] \tag{5.3}$$

where R_0 is the offset (eg. mean value of x[n]), R is the amplitude of the sine wave, f is the frequency and Φ is the phase. What our model does'nt explain will be soaked up in the error term $e[n]$. Because of the trig identity

$$\cos(A + B) = \cos A \cos B - \sin A \sin B \tag{5.4}$$

the model can be written in an alternative form

$$x[n] = R_0 + a\cos(2\pi f t[n]) + b\sin(2\pi f t[n]) + e[n] \tag{5.5}$$

where $a = R\cos(\Phi)$ and $b = -R\sin(\Phi)$. This is the form we consider for subsequent analysis.

This type of model is similar to a class of models in statistics called *Generalised Linear Models* (GLIMS). They perform *nonlinear regression* by, first, taking *fixed* nonlinear functions of the inputs, these functions being called *basis functions*, and second, form an output by taking a linear combination of the basis function outputs. In sinusoidal models the basis functions are sines and cosines. In statistics a much broader class of functions is considered. However, sinewaves have some nice properties as we shall see.

5.3.1 Fitting the model

If we let $\boldsymbol{x} = [x(1), x(2), ..., x(N)]^T$, $\boldsymbol{w} = [R_0, a, b]^T$, $\boldsymbol{e} = [e_1, e_2, ..., e_N]^T$ and

$$\boldsymbol{A} = \begin{bmatrix} 1 & cos2\pi ft[1] & sin2\pi ft[1] \\ 1 & cos2\pi ft[2] & sin2\pi ft[2] \\ 1 & cos2\pi ft[3] & sin2\pi ft[3] \\ .. & .. & .. \\ 1 & cos2\pi ft[N] & sin2\pi ft[N] \end{bmatrix} \tag{5.6}$$

then the model can be written in the matrix form

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{w} + \boldsymbol{e} \tag{5.7}$$

which is in the standard form of a multivariate linear regression problem. The solution is therefore

$$\boldsymbol{w} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{x} \tag{5.8}$$

5.3.2 But sinewaves are orthogonal

Because we are dealing with sinewaves it turns out that the above solution simplifies. We restrict ourselves to a frequency f_p which is an integer multiple of the *base frequency*

$$f_p = pF_b \tag{5.9}$$

where $p = 1..N/2$ and

$$f_b = \frac{F_s}{N} \tag{5.10}$$

eg. for $F_s = 100$ and $N = 100$ (1 seconds worth of data), $f_b = 1Hz$ and we can have f_p from 1Hz up to 50Hz[1]. The *orthogonality* of sinewaves is expressed in the following equations

$$\sum_{n=1}^{N} \cos 2\pi f_k t[n] = \sum_{n=1}^{N} \sin 2\pi f_k t[n] = 0 \tag{5.11}$$

[1] To keep things simple we don't allow f_p where $p = N/2$; if we did allow it we'd get N and 0 in equations 5.14 and 5.15 for the case $k = l$. Also we must have N even.

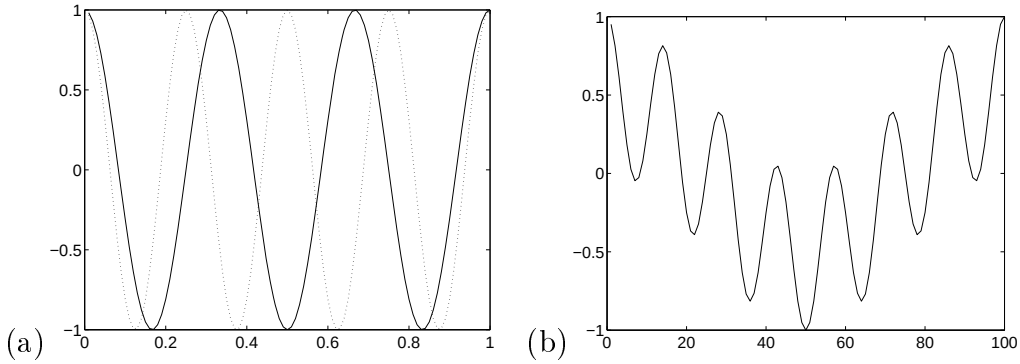


Figure 5.2: **Orthogonality of sinewaves** *Figure (a) shows* $\cos 2\pi 3 f_b t[n]$ *and* $\cos 2\pi 4 f_b t[n]$*, cosines which are 3 and 4 times the base frequency* $f_b = 1Hz$*. For any two integer multiples* k, l *we get* $\sum_{n=1}^{N} cos 2\pi f_k t[n] cos 2\pi f_l t[n] = 0$*. This can be seen from Figure (b) which shows the product* $cos 2\pi 3 f_b t[n] cos 2\pi 4 f_b t[n]$*. Because of the trig identity* $cosAcosB = 0.5cos(A+B) + 0.5cos(A-B)$ *this looks like a 7Hz signal superimposed on a 1Hz signal. The sum of this signal over a whole number of cycles can be seen to be zero; because each cos term sums to zero. If, however,* k *or* l *are not integers the product does not sum to zero and the orthogonality breaks down.*

$$\sum_{n=1}^{N} \cos 2\pi f_k t[n] \sin 2\pi f_l t[n] = 0 \tag{5.12}$$

$$\sum_{n=1}^{N} \cos 2\pi f_k t[n] \sin 2\pi f_l t[n] = 0 \tag{5.13}$$

$$\sum_{n=1}^{N} \cos 2\pi f_k t[n] \cos 2\pi f_l t[n] = \begin{matrix} 0 & k \neq l \\ N/2 & k = l \end{matrix} \tag{5.14}$$

$$\sum_{n=1}^{N} \sin 2\pi f_k t[n] \sin 2\pi f_l t[n] = \begin{matrix} 0 & k \neq l \\ N/2 & k = l \end{matrix} \tag{5.15}$$

These results can be proved by various trig. identities or, more simply, by converting to complex exponentials (see [5] or later in this chapter). The results depend on the fact that all frequencies that appear in the above sums are integer multiples of the base frequency; see figure 5.2.

This property of sinewaves leads to the result

$$\boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{D} \tag{5.16}$$

where $\boldsymbol{D}$ is a diagonal matrix. The first entry is N (from the inner product of two columns of 1's of length N; the 1's are the coefficients of the constant term R_0) and all the other entries are $N/2$. A matrix $\boldsymbol{Q}$ for which

$$\boldsymbol{Q}^T \boldsymbol{Q} = \boldsymbol{D} \tag{5.17}$$

is said to be orthogonal. Therefore our $\boldsymbol{A}$ matrix is orthogonal. This greatly simplifies the fitting of the model which now reduces to

$$\boldsymbol{w} = \boldsymbol{D}^{-1} \boldsymbol{A}^T \boldsymbol{x} \tag{5.18}$$

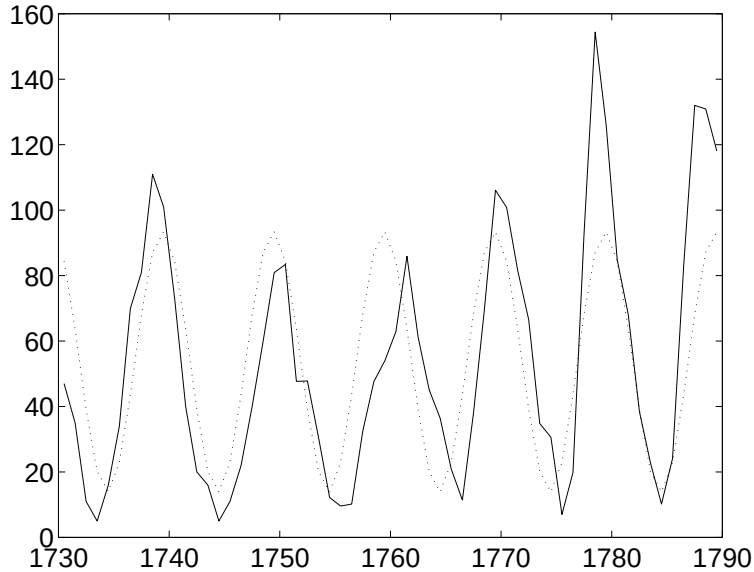


Figure 5.3: *Sunspot index (solid line) and prediction of it from a simple sinusoidal model (dotted line).*

which is simply a *projection* of the signal onto the basis matrix, with some pre-factor ($\boldsymbol{D}^{-1}$; remember the inverse of a diagonal matrix is simply the inverse of each of the diagonal terms, so this is easy to compute). Given that $\boldsymbol{w} = [a, b, R_0]^T$ we can see that, for example, a is computed by simply projecting the data onto the second column of the matrix $\boldsymbol{A}$, eg.

$$a = \frac{2}{N} \sum_{n=1}^{N} \cos(2\pi f t) x_t \tag{5.19}$$

Similarly,

$$b = \frac{2}{N} \sum_{n=1}^{N} \sin(2\pi f t) x_t \tag{5.20}$$

$$R_0 = \frac{1}{N} \sum_{n=1}^{N} x_t \tag{5.21}$$

We applied the simple sinusoidal model to a 'sunspot data set' as follows. We chose 60 samples between the years 1731 and 1790 (because there was a fairly steady mean level in this period). The sampling rate F_s = 1Year. This gives a base frequency of $f_b = 1/60$. We chose our frequency $f = pf_b$ with p=6; giving a complete cycle once every ten years. This gave rise to the following estimates; $R_0 = 53.64$, $a = 39.69$ and $b = -2.36$. The data and predictions are shown in Figure 5.3.

5.4 Fourier Series

We might consider that our signal consists of lots of periodic components in which case the *multiple sinusoidal model* would be more appropriate

$$x(t) = R_0 + \sum_{k=1}^{p} R_k \cos(2\pi f_k t + \Phi_k) + e_t \tag{5.22}$$

where there are p sinusoids with different frequencies and phases. In a discrete Fourier series there are $p = N/2$ such sinusoids having frequencies

$$f_k = \frac{kF_s}{N} \tag{5.23}$$

where $k = 1..N/2$ and F_s is the sampling frequency. Thus the frequencies range from F_s/N up to $F_s/2$. The Fourier series expansion of the signal $x(t)$ is

$$x(t) = R_0 + \sum_{k=1}^{N/2} R_k \cos(2\pi f_k t + \Phi_k) \tag{5.24}$$

Notice that there is no noise term. Because of the trig identity

$$\cos(A + B) = \cos A \cos B - \sin A \sin B \tag{5.25}$$

this can be written in the form

$$x(t) = a_0 + \sum_{k=1}^{N/2} a_k \cos(2\pi f_k t) + b_k \sin(2\pi f_k t) \tag{5.26}$$

where $a_k = R_k \cos(\Phi_k)$ and $b_k = -R_k \sin(\Phi_k)$. Alternatively, we have $R_k^2 = a_k^2 + b_k^2$ and $\Phi = \tan^{-1}(b_k/a_k)$. The signal at frequency f_k is known as the *kth harmonic*. Equivalently, we can write the nth sample as

$$x[n] = a_0 + \sum_{k=1}^{N/2} a_k \cos(2\pi f_k t[n]) + b_k \sin(2\pi f_k t[n]) \tag{5.27}$$

where $t[n] = nT_s$.

The important things to note about the sinusoids in a Fourier series are (i) the frequencies are *equally* spread out, (ii) there are $N/2$ of them where N is the number of samples, (iii) Given F_s and N the frequencies are *fixed*. Also, note that in the Fourier series 'model' there is *no noise*. The Fourier series aims to represent the data *perfectly* (which it can do due to the excessive number of basis functions)[2].

The Fourier coefficients can be computed by a generalisation of the process used to compute the coefficients in the simple sinusoidal model.

$$a_k = \frac{2}{N} \sum_{n=1}^{N} \cos(2\pi f_k t[n]) x[n] \tag{5.28}$$

[2]Statisticians would frown on fitting a model with N coefficients to N data points as the estimates will be very noisy; the Fourier series is a low bias (actually zero), high variance model. This underlines the fact that the Fourier methods are transforms rather than statistical models.

Similarly,

$$b_k = \frac{2}{N} \sum_{n=1}^{N} \sin(2\pi f_k t[n]) x[n] \tag{5.29}$$

$$a_0 = \frac{1}{N} \sum_{n=1}^{N} x[n] \tag{5.30}$$

These equations can be derived as follows. To find, for example, a_k, multiply both sides of equation 5.27 by $\cos(2\pi f_k t[n])$ and sum over n. Due to the orthogonality property of sinusoids (which still holds as all frequencies are integer multiples of a base frequency) all terms on the right go to zero except for the one involving a_k. This just leaves $a_k(N/2)$ on the right giving rise to the above formula.

5.4.1 Example

The plots on the right of Figure 5.4 show four components in a Fourier series expansion. The components have been ordered by amplitude. The plots on the left of the Figure show the corresponding Fourier approximation.

5.5 Fourier Transforms

Fourier series are representations of a signal by combinations of sinewaves of different magnitudes, frequencies and offsets (or *phases*). The magnitudes are given by the *Fourier coefficients*. These sinewaves can also be represented in terms of *complex exponentials* (see the appendix for a quick review of complex numbers); a representation which ultimately leads to algorithms for computing the Fourier coefficients; the Discrete Fourier Transform (DFT) and the Fast Fourier Transform (FFT).

5.5.1 Discrete Fourier Transform

Fourier series can be expressed in terms of complex exponentials. This representation leads to an efficient method for computing the coefficients. We can write the cosine terms as complex exponentials

$$a_k \cos(2\pi f_k t[n]) = a_k \frac{\exp(i2\pi f_k t[n]) + \exp(-i2\pi f_k t[n])}{2} \tag{5.31}$$

where $i^2 = -1$. Picture this as the addition of two vectors; one above the real axis and one below. Together they make a vector on the real axis which is then halved.

We can also write the sine terms as

$$b_k \sin(2\pi f_k t[n]) = b_k \frac{\exp(i2\pi f_k t[n]) - \exp(-i2\pi f_k t[n])}{2i} \tag{5.32}$$

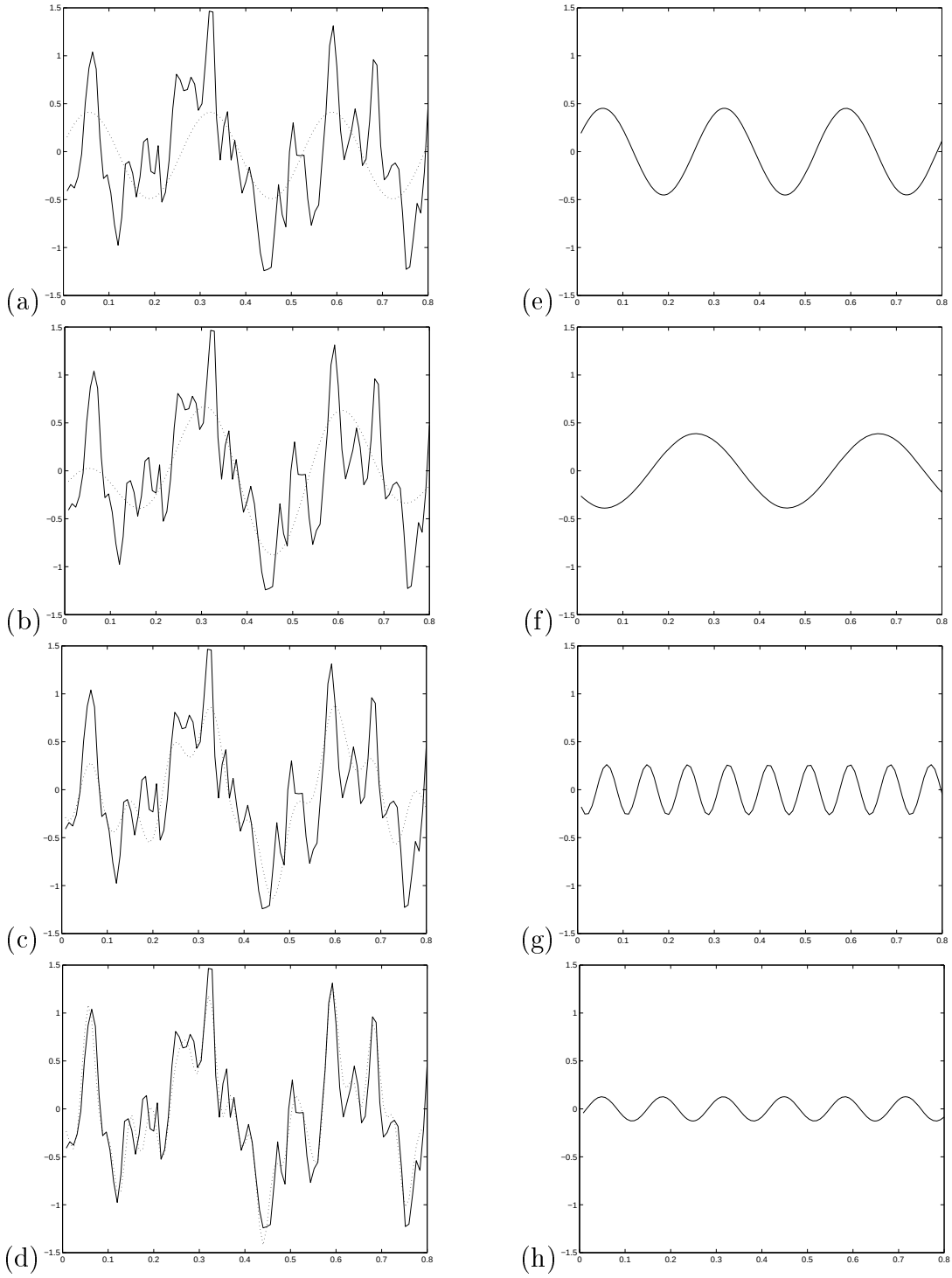


Figure 5.4: *Signal (solid line) and components of the Fourier series approximation* $\sum_{k=1}^{p} R_k cos(2\pi f_k + \Phi_k)$ *(dotted lines) with (a)* $p = 1$*, (b)* $p = 2$*, (c)* $p = 3$ *and (d)* $p = 11$ *where we have ordered the components according to amplitude. The corresponding individual terms are (e)* $R^2 = 0.205, f = 3.75$ *and* $\Phi = 0.437$*, (f)* $R^2 = 0.151$*,* $f = 2.5$ *and* $\Phi = 0.743$*, (g)* $R^2 = 0.069$*,* $f = 11.25$ *and* $\Phi = 0.751$ *and (h)* $R^2 = 0.016$*,* $f = 7.5$ *and* $\Phi = -0.350$*.*

Picture this as one vector above the real axis minus another vector below the real axis. This results in a purely imaginary (and positive) vector. The result is halved and then multiplied by the vector $\exp(3\pi/2)$ ($-i$, from multplying top and bottom by i) which provides a rotation to the real axis.

Adding them (and moving i to the numerator by multiplying b_k top and bottom by i) gives

$$\frac{1}{2}(a_k - b_k i)\exp(i2\pi f_k t[n]) + \frac{1}{2}(a_k + b_k i)\exp(-i2\pi f_k t[n]) \tag{5.33}$$

Note that a single term at frequency k has split into a complex combination (the coefficients are complex numbers) of a positive frequency term and a *negative frequency* term. Substituting the above result into equation 5.27 and noting that $f_k t[n] = kn/N$ we get

$$x[n] = a_0 + \frac{1}{2}\sum_{k=1}^{N/2}(a_k - b_k i)\exp(i2\pi kn/N) + \frac{1}{2}\sum_{k=1}^{N/2}(a_k + b_k i)\exp(-i2\pi kn/N) \tag{5.34}$$

If we now let

$$\tilde{X}(k) = \frac{N}{2}(a_k - b_k i) \tag{5.35}$$

and note that for real signals $\tilde{X}(-k) = \tilde{X}^*(k)$ (negative frequencies are reflections across the real plane, ie. conjugates) then the $(a_k + b_k i)$ terms are equivalent to $\tilde{X}(-k)$. Hence

$$x[n] = a_0 + \frac{1}{2N}\sum_{k=1}^{N/2}\tilde{X}(k)\exp(i2\pi kn/N) + \frac{1}{2N}\sum_{k=1}^{N/2}\tilde{X}(k)\exp(-i2\pi kn/N) \tag{5.36}$$

Now, because $\tilde{X}(N-k) = \tilde{X}(-k)$ (this can be shown by considering the Fourier transform of a signal $x[n]$ and using the decomposition $\exp(-i2\pi(N-k)n/N) = \exp(-i2\pi N/N)\exp(i2\pi kn/N)$ where the first term on the right is unity) we can write the second summation as

$$x[n] = a_0 + \frac{1}{2N}\sum_{k=1}^{N/2}\tilde{X}(k)\exp(i2\pi kn/N) + \frac{1}{2N}\sum_{k=N/2}^{N-1}\tilde{X}(k)\exp(-i2\pi(N-k)n/N) \tag{5.37}$$

Using the same exponential decomposition allows us to write

$$x[n] = a_0 + \frac{1}{N}\sum_{k=1}^{N-1}\tilde{X}(k)\exp(i2\pi kn/N) \tag{5.38}$$

If we now let $X(k+1) = \tilde{X}(k)$ then we can absorb the constant a_0 into the sum giving

$$x[n] = \frac{1}{N}\sum_{k=1}^{N}X(k)\exp(i2\pi(k-1)n/N) \tag{5.39}$$

which is known as the *Inverse Discrete Fourier Transform* (IDFT). The terms $X(k)$ are the *complex valued Fourier coefficients*. We have the relations

$$a_0 = Re\{X(1)\} \tag{5.40}$$

$$
\begin{aligned}
a_k &= \frac{2}{N} Re\{X(k+1)\} \\
b_k &= \frac{-2}{N} Im\{X(k+1)\}
\end{aligned}
$$

The complex valued Fourier coefficients can be computed by first noting the orthogonality relations

$$
\sum_{n=1}^{N} \exp(i2\pi(k-1)n/N) = \begin{array}{ll} N & k = 1, \pm(N+1), \pm(N+2) \\ 0 & otherwise \end{array} \tag{5.41}
$$

If we now multiply equation 5.39 by $\exp(-i2\pi ln/N)$, sum from 1 to N and re-arrange we get

$$
X(k) = \sum_{n=1}^{N} x(n) \exp(-i2\pi(k-1)n/N) \tag{5.42}
$$

which is the Discrete Fourier Transform (DFT).

5.5.2 The Fourier Matrix

If we write $X(k)$ as a vector $\boldsymbol{X} = [X(1), X(2), ..., X(N)]^T$ and the input signal as a vector $\boldsymbol{x} = [x(0), x(1), ..., x(N-1)]^T$ then the above equations can be written in matrix form as follows. The Inverse Discrete Fourier Transform is

$$
\boldsymbol{x} = \boldsymbol{F}\boldsymbol{X} \tag{5.43}
$$

where $\boldsymbol{F}$ is the *Fourier Matrix* and the Discrete Fourier Transform is

$$
\boldsymbol{X} = \boldsymbol{F}^{-1}\boldsymbol{x} \tag{5.44}
$$

If we let

$$
w_N = \exp(i2\pi/N) \tag{5.45}
$$

we can write the *Fourier matrix* as

$$
\boldsymbol{F}_N = \frac{1}{N} \begin{bmatrix} 1 & 1 & 1 & . & 1 \\ 1 & w_N & w_N^2 & . & w_N^{(N-1)} \\ 1 & w_N^2 & w_N^4 & . & w_N^{2(N-1)} \\ . & . & . & . & . \\ 1 & w_N^{(N-1)} & w_N^{2(N-1)} & . & w_N^{(N-1)^2} \end{bmatrix} \tag{5.46}
$$

which has elements[3]

$$
(\boldsymbol{F}_N)_{kn} = w_N^{(k-1)(n-1)} \tag{5.47}
$$

[3] We have re-indexed such that we now have $x(0)$ to $x(N-1)$. Hence we have $(n-1)$ instead of n.

Now, the inverse Fourier matrix is

$$\boldsymbol{F}_N^{-1} = \begin{bmatrix} 1 & 1 & 1 & . & 1 \\ 1 & w_N^{-1} & w_N^{-2} & . & w_N^{-(N-1)} \\ 1 & w_N^{-2} & w_N^{-4} & . & w_N^{-2(N-1)} \\ . & . & . & . & . \\ 1 & w_N^{-(N-1)} & w_N^{-2(N-1)} & . & w_N^{-(N-1)^2} \end{bmatrix} \tag{5.48}$$

where the elements are

$$(\boldsymbol{F}_N^{-1})_{kn} = w_N^{-(k-1)(n-1)} \tag{5.49}$$

In the *Fast Fourier Transform* (FFT) an N-dimensional matrix multiplication can be replaced by 2 M-dimensional multiplications, where $M = N/2$. This is because the exponential elements in the Fourier matrix have the key property

$$w_N^2 = w_M \tag{5.50}$$

eg. $\exp(i2\pi/64)^2 = \exp(i2\pi/32)$. Cooley and Tukey realised you could use this property as follows. If you split the IDFT

$$x_j = \sum_{k=0}^{N-1} w_N^{jk} X_k \tag{5.51}$$

into a summation of even parts and a summation of odd parts

$$x_j = \sum_{k=0}^{M-1} w_N^{2jk} X_{2k} + \sum_{k=0}^{M-1} w_N^{(2k+1)j} X_{2k+1} \tag{5.52}$$

then we can use the identity $w_N^2 = w_M$ to give

$$x_j = \sum_{k=0}^{M-1} w_M^{jk} X_{2k} + w_N^j \sum_{k=0}^{M-1} w_M^{kj} X_{2k+1} \tag{5.53}$$

which is the summation of two IDFTs of dimension M (a similar argument applies for the DFT).

This reduces the amount of computation by, approximately, a factor of 2. We can then replace each M-dimensional multiplication by an $M/2$-dimensional one, etc. FFTs require N to be a power of 2, because at the lowest level we have lots of 2-dimensional operations. For $N = 4096$ we get an overall speed-up by a factor of 680. For larger N the speed-ups are even greater; we are replacing N^2 multiplications by $\frac{N}{2}\log_2 N$.

5.6 Time-Frequency relations

Signals can be operated on in the *time domain* or in the *frequency domain*. We now explore the relations between them.

5.6.1 Power Spectral Density

The power in a signal is given by

$$P_x = \sum_{n=1}^{N} |x[n]|^2 \tag{5.54}$$

We now derive an expression for P_x in terms of the Fourier coefficients. If we note that $|x[n]|$ can also be written in its conjugate form (the conjugate form has the same magnitude; the phase is different but this does'nt matter as we're only interested in magnitude)

$$|x[n]| = \frac{1}{N} \sum_{k=1}^{N} X^*(k) \exp(-i2\pi(k-1)n/N) \tag{5.55}$$

then we can write the power as

$$P_x = \sum_{n=1}^{N} |x[n] \frac{1}{N} \sum_{k=1}^{N} X^*(k) \exp(-i2\pi(k-1)n/N)| \tag{5.56}$$

If we now change the order of the summations we get

$$P_x = \frac{1}{N} \sum_{k=1}^{N} |X^*(k) \sum_{n=1}^{N} x(n) \exp(-i2\pi(k-1)n/N)| \tag{5.57}$$

where the sum on the right is now equivalent to $X(k)$. Hence

$$P_x = \frac{1}{N} \sum_{k=1}^{N} |X(k)|^2 \tag{5.58}$$

We therefore have an equivalence between the power in the time domain and the power in the frequency domain which is known as *Parseval's relation*. The quantity

$$P_x(k) = |X(k)|^2 \tag{5.59}$$

is known as the *Power Spectral Density* (PSD).

5.6.2 Filtering

The filtering process

$$x[n] = \sum_{l=-\infty}^{\infty} x_1(l) x_2(n-l) \tag{5.60}$$

is also known as *convolution*

$$x[n] = x_1(n) * x_2(n) \tag{5.61}$$

We will now see how it is related to frequency domain operations. If we let $w = 2\pi(k-1)/N$, multiply both sides of the above equation by $\exp(-iwn)$ and sum over n the left hand side becomes the Fourier transform

$$X(w) = \sum_{n=-\infty}^{\infty} x[n] \exp(-iwn) \tag{5.62}$$

and the right hand side (RHS) is

$$\sum_{n=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} x_1(l) x_2(n-l) \exp(-iwn) \tag{5.63}$$

Now, we can re-write the exponential term as follows

$$\exp(-iwn) = \exp(-iw(n-l)) \exp(-iwl) \tag{5.64}$$

Letting $n' = n - l$, we can write the RHS as

$$\sum_{l=-\infty}^{\infty} x_1(l) \exp(-iwl) \sum_{n'=-\infty}^{\infty} x_2(n') \exp(-iwn') = X_1(w) X_2(w) \tag{5.65}$$

Hence, the filtering operation is equivalent to

$$X(w) = X_1(w) X_2(w) \tag{5.66}$$

which means that convolution in the time domain is equivalent to multiplication in the frequency domain. This is known as the *convolution theorem.*

5.6.3 Autocovariance and Power Spectral Density

The autocovariance of a signal is given by

$$\sigma_{xx}(n) = \sum_{l=-\infty}^{\infty} x(l) x(l-n) \tag{5.67}$$

Using the same method that we used to prove the convolution theorem, but noting that the term on the right is $x(l-n)$ not $x(n-l)$ we can show that the RHS is equivalent to

$$X(w) X(-w) = |X(w)|^2 \tag{5.68}$$

which is the Power Spectral Density, $P_x(w)$. Combining this with what we get for the left hand side gives

$$P_x(w) = \sum_{n=-\infty}^{\infty} \sigma_{xx}(n) \exp(-iwn) \tag{5.69}$$

which means that the PSD is the Fourier Transform of the autocovariance. This is known as the *Wiener-Khintchine Theorem.* This is an important result. It means that the PSD can be estimated from the autocovariance and vice-versa. It also means that the PSD and the autocovariance contain the same information about the signal.

It is also worth noting that since both contain no information about the phase of a signal then the signal cannot be uniquely constructed from either. To do this we need to know the PSD *and* the *Phase spectrum* which is given by

$$\Phi(k) = \tan^{-1}(\frac{b_k}{a_k}) \tag{5.70}$$

where b_k and a_k are the real Fourier coefficients.

We also note that the Fourier transform of a symmetric function is real. This is because symmetric functions can be represented entirely by cosines, which are themselves symmetric; the sinewaves, which constitute the complex component of a Fourier series, are no longer necessary. Therefore, because the autocovariance is symmetric the PSD is real.

5.7 Spectral Estimation

5.7.1 The Periodogram

The periodogram of a signal x_t is a plot of the normalised power in the kth harmonic versus the frequency, f_k of the kth harmonic. It is calculated as

$$I(f_k) = \frac{N}{4\pi}(a_k^2 + b_k^2) \tag{5.71}$$

where a_k and b_k are the Fourier coefficients.

The periodogram is a low bias (actually unbiased) but high variance [4] estimate of the power at a given frequency. This is therefore a problem if the number of data points is small; the estimated spectrum will be very spiky.

To overcome this, a number of algorithms exist to smooth the periodogram ie. to reduce the variance. The Bartlett method, for example, takes an N-point sequence and subdivides it into K nonoverlapping segments and calculates $I(f_k)$ for each. The final periodogram is just the average over the K estimates. This results in a reduction in variance by a factor K at the cost of reduced spectral resolution (by a factor K).

The Welch method is similar but averages modified periodograms, the modification being a windowing of each segment of data. Also, the segments are allowed to overlap. For further details of this and other smoothing methods see Chapter 12 in Proakis and Manolakis [51]. This smoothing is necessary because at larger lags there are fewer data points, so the estimates of covariance are commensurately more unreliable.

5.7.2 Autocovariance methods

The PSD can be calculated from the autocovariance. However, as the sample autocovariance on short segments of data has a high variance then so will the resulting spectral estimates.

To overcome this a number of proposals have been made. The autocovariance function can first be smoothed and truncated by applying various smoothing windows,

[4] It is an *inconsistent* estimator, because the variance does'nt reduce to zero as the number of samples tends to infinity.

for example Tukey, Parzen, Hanning or Hamming windows. For further details see Chatfield p.114 [11] or Chapter 12 in Proakis and Manolakis [51].

5.7.3 Aliasing

Because of aliasing if we wish to uniquely identify frequencies up to BHz then we must sample the data at a frequency $f_s > 2B$Hz.

Alternatively, given a particular sample rate f_s, in order to uniquely identify frequencies up to $f_s/2$Hz (and not confuse them with higher frequencies) we must ensure that there is no signal at frequencies higher than $f_s/2$. This can be achieved by applying a Low-Pass Filter (LPF).

5.7.4 Filtering

There are two main classes of filters; IIR filters and FIR filters. Their names derive from how the filters respond to a single pulse of input, their so-called impulse response. The output of an Infinite Impulse Response (IIR) filter is fed-back to its input. The response to a single impulse is therefore a gradual decay which, though it may drop rapidly towards zero (no output), will never technically reach zero; hence the name IIR.

In Finite Impulse Response (FIR) filters the output is not fed-back to the input so if there is no subsequent input there will be no output. The output of an FIR filter ([51], page 620) is given by

$$y[n] = \sum_{k=0}^{p-1} b_k x[n-k] \tag{5.72}$$

where $x[n]$ is the original signal and b_k are the filter coefficients.

The simplest FIR filter is the (normalised) rectangular window which takes a moving average of a signal. This smooths the signal and therefore acts a low-pass filter. Longer windows cut down the range of frequencies that are passed.

Other examples of FIR filters are the Bartlett, Blackman, Hamming and Hanning windows shown in Figure 5.5. The curvier shape of the windows means their frequency characteristics are more sharply defined. See Chapter 8 in [51] for more details. FIR filters are also known as Moving Average (MA) models which we will encounter in the next lecture.

The output of an IIR filter is given by

$$y[n] = \sum_{k=0}^{p_a-1} a_k y[n-k] + \sum_{k=0}^{p_b-1} b_k x[n-k] \tag{5.73}$$

where the first term includes the feedback coefficients and the second term is an FIR

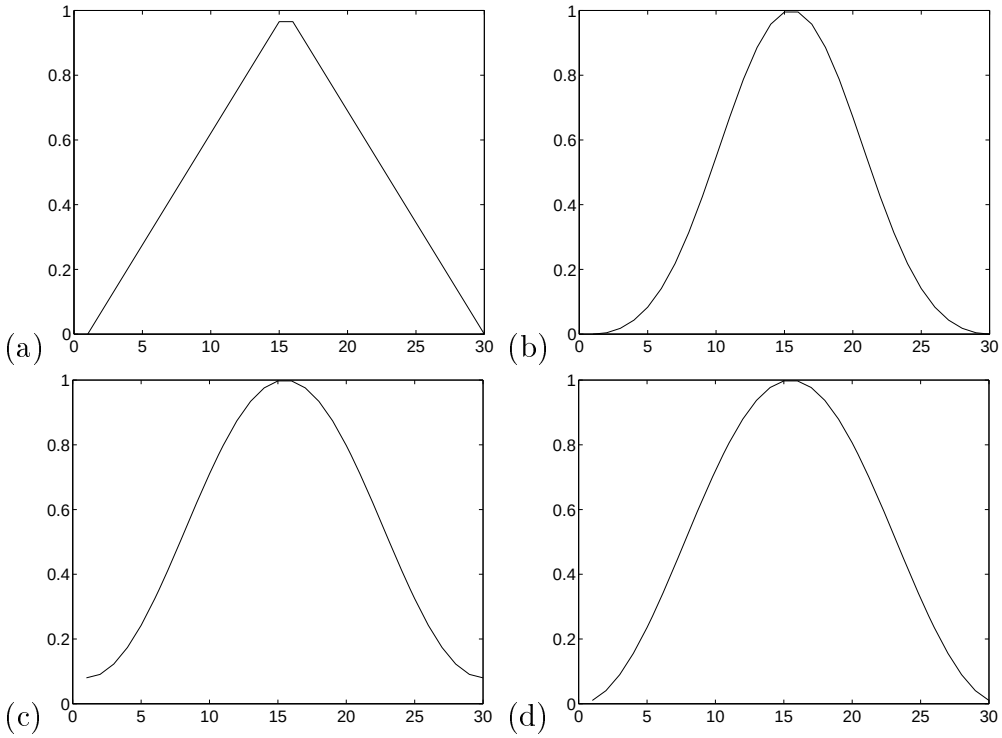


Figure 5.5: *Filter coefficients of (a) Bartlett (triangular), (b) Blackman, (c) Hamming and (d) Hanning windows for* $p = 30$.

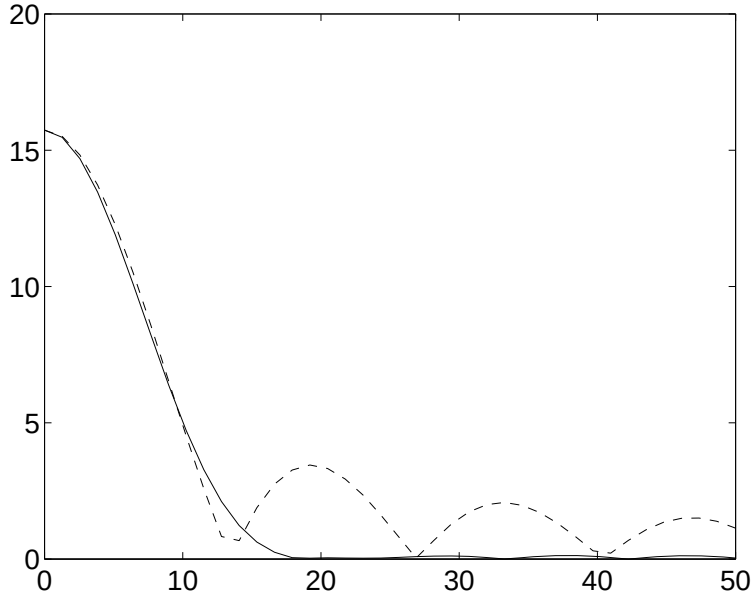


Figure 5.6: *Frequency response of a Hamming window (solid line) and a rectangular window (dotted line). The Hamming window cuts of the higher frequencies more sharply.*

model. This type of filter is also known as a Autoregressive Moving Average (ARMA) model (the first term being the Autoregressive (AR) part).

IIR filters can be designed by converting analog filters into the above IIR digital form. See [51] (section 8.3) for details. Examples of resulting IIR implementations are the Butterworth, Chebyshev, Elliptic and Bessel filters.

Chapter 6

Stochastic Processes

6.1 Introduction

In Lecture 1 we discussed correlation and regression. We now discuss autocorrelation and autoregressive processes; that is, the correlation between successive values of a time series and the linear relations between them. We also show how autoregressive models can be used for spectral estimation. Good textbooks that cover this material are those by Grimmett and Stirzaker [25] and Papoulis [44].

6.2 Autocorrelation

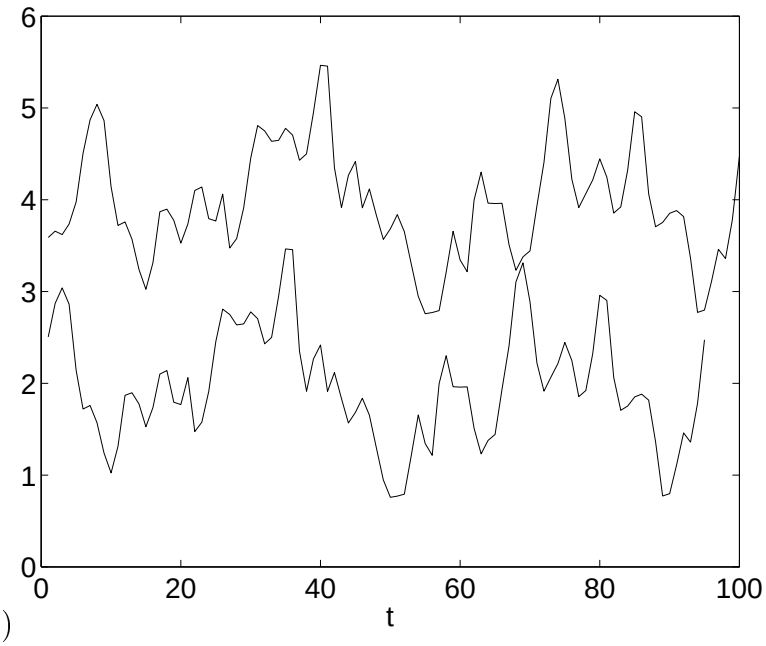
Given a time series x_t we can produce a *lagged* version of the time series x_{t-T} which *lags* the original by T samples. We can then calculate the covariance between the two signals

$$\sigma_{xx}(T) = \frac{1}{N-1}\sum_{t=1}^{N}(x_{t-T} - \mu_x)(x_t - \mu_x) \tag{6.1}$$

where μ_x is the signal mean and there are N samples. We can then plot $\sigma_{xx}(T)$ as a function of T. This is known as the *autocovariance* function. The *autocorrelation* function is a normalised version of the autocovariance

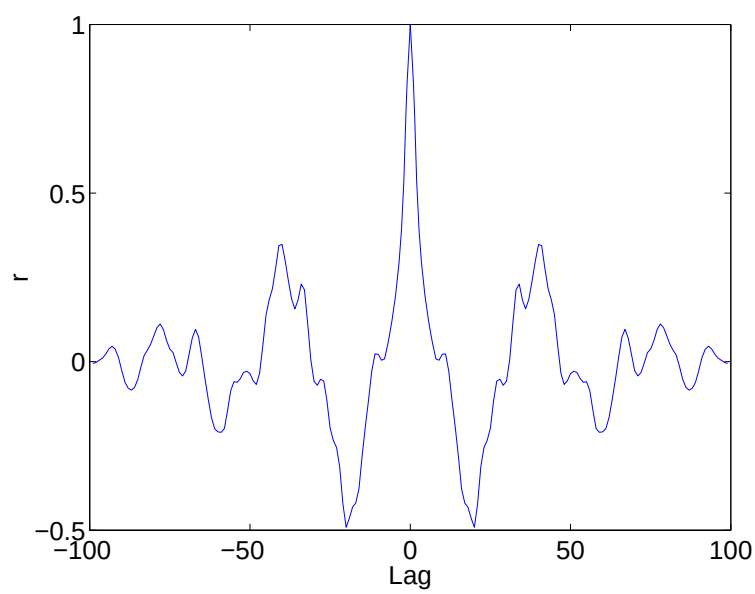
$$r_{xx}(T) = \frac{\sigma_{xx}(T)}{\sigma_{xx}(0)} \tag{6.2}$$

Note that $\sigma_{xx}(0) = \sigma_x^2$. We also have $r_{xx}(0) = 1$. Also, because $\sigma_{xy} = \sigma_{yx}$ we have $r_{xx}(T) = r_{xx}(-T)$; the autocorrelation (and autocovariance) are *symmetric* functions or *even functions*. Figure 6.1 shows a signal and a lagged version of it and Figure 7.2 shows the autocorrelation function.



(a)

Figure 6.1: *Signal x_t (top) and x_{t+5} (bottom). The bottom trace* **leads** *the top trace by 5 samples. Or we may say it* **lags** *the top by -5 samples.*



(a)

Figure 6.2: *Autocorrelation function for x_t. Notice the negative correlation at lag 20 and positive correlation at lag 40. Can you see from Figure 6.1 why these should occur ?*

6.3 Autoregressive models

An autoregressive (AR) model *predicts* the value of a time series from previous values. A pth order AR model is defined as

$$x_t = \sum_{i=1}^{p} x_{t-i} a_i + e_t \tag{6.3}$$

where a_i are the *AR coefficients* and e_t is the prediction error. These errors are assumed to be Gaussian with zero-mean and variance σ_e^2. It is also possible to include an extra parameter a_0 to soak up the mean value of the time series. Alternatively, we can first subtract the mean from the data and then apply the zero-mean AR model described above. We would also subtract any trend from the data (such as a linear or exponential increase) as the AR model assumes stationarity (see later).

The above expression shows the relation for a single time step. To show the relation for all time steps we can use matrix notation.

We can write the AR model in matrix form by making use of the *embedding matrix*, $\boldsymbol{M}$, and by writing the signal and AR coefficients as vectors. We now illustrate this for $p = 4$. This gives

$$\boldsymbol{M} = \begin{bmatrix} x_4 & x_3 & x_2 & x_1 \\ x_5 & x_4 & x_3 & x_2 \\ .. & .. & .. & .. \\ x_{N-1} & x_{N-2} & x_{N-3} & x_{N-4} \end{bmatrix} \tag{6.4}$$

We can also write the AR coefficients as a vector $\boldsymbol{a} = [a_1, a_2, a_3, a_4]^T$, the errors as a vector $\boldsymbol{e} = [e_5, e_6, ..., e_N]^T$ and the signal itself as a vector $\boldsymbol{X} = [x_5, x_6, ..., x_N]^T$. This gives

$$\begin{bmatrix} x_5 \\ x_6 \\ .. \\ x_N \end{bmatrix} = \begin{bmatrix} x_4 & x_3 & x_2 & x_1 \\ x_5 & x_4 & x_3 & x_2 \\ .. & .. & .. & .. \\ x_{N-1} & x_{N-2} & x_{N-3} & x_{N-4} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} + \begin{bmatrix} e_5 \\ e_6 \\ .. \\ e_N \end{bmatrix} \tag{6.5}$$

which can be compactly written as

$$\boldsymbol{X} = \boldsymbol{M}\boldsymbol{a} + \boldsymbol{e} \tag{6.6}$$

The AR model is therefore a special case of the multivariate regression model (compare the above equation to that given in the second lecture). The AR coefficients can therefore be computed from the equation

$$\hat{\boldsymbol{a}} = (\boldsymbol{M}^T \boldsymbol{M})^{-1} \boldsymbol{M}^T \boldsymbol{X} \tag{6.7}$$

The AR predictions can then be computed as the vector

$$\hat{\boldsymbol{X}} = \boldsymbol{M}\hat{\boldsymbol{a}} \tag{6.8}$$

and the error vector is then $\boldsymbol{e} = \boldsymbol{X} - \hat{\boldsymbol{X}}$. The variance of the noise is then calculated as the variance of the error vector.

To illustrate this process we analyse our data set using an AR(4) model. The AR coefficients were estimated to be

$$\hat{\boldsymbol{a}} = [1.46, -1.08, 0.60, -0.186]^T \tag{6.9}$$

and the AR predictions are shown in Figure 6.3. The noise variance was estimated to be $\sigma_e^2 = 0.079$ which corresponds to a standard deviation of 0.28. The variance of the original time series was 0.3882 giving a signal to noise ratio of $(0.3882-0.079)/0.079 = 3.93$.

6.3.1 Random walks

If $p = 1$ and $a_1 = 1$ then the AR model reduces to a random walk model, an example of which is shown in Figure 6.4.

6.3.2 Relation to autocorrelation

The autoregressive model can be written as

$$x_t = a_1 x_{t-1} + a_2 x_{t-2} + ... + a_p x_{t-p} + e_t \tag{6.10}$$

If we multiply both sides by x_{t-k} we get

$$x_t x_{t-k} = a_1 x_{t-1} x_{t-k} + a_2 x_{t-2} x_{t-k} + ... + a_p x_{t-p} x_{t-k} + e_t x_{t-k} \tag{6.11}$$

If we now sum over t and divide by $N-1$ and assume that the signal is zero mean (if it isn't we can easily make it so, just by subtracting the mean value from every sample) the above equation can be re-written in terms of covariances at different lags

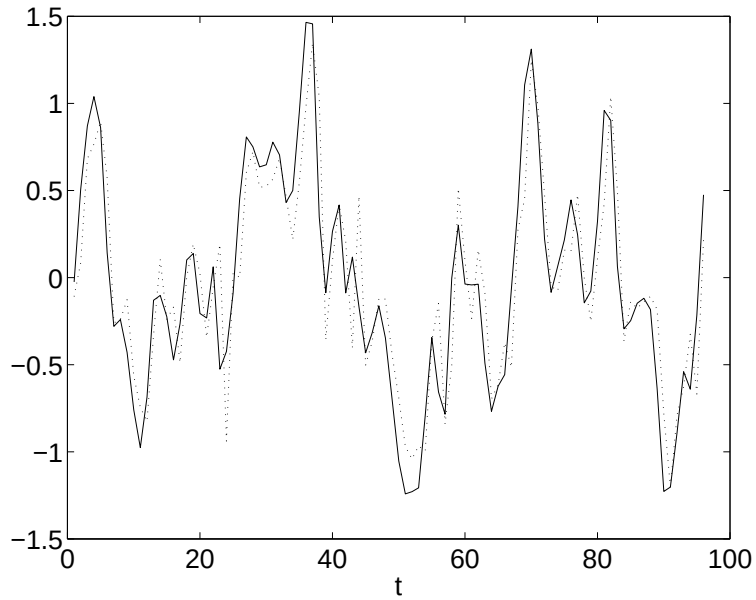
$$\sigma_{xx}(k) = a_1 \sigma_{xx}(k-1) + a_2 \sigma_{xx}(k-2) + ... + a_p \sigma_{xx}(k-p) + \sigma_{e,x} \tag{6.12}$$

where the last term $\sigma_{e,x}$ is the covariance between the noise and the signal. But as the noise is assumed to be independent from the signal $\sigma_{e,x} = 0$. If we now divide every term by the signal variance we get a relation between the correlations at different lags

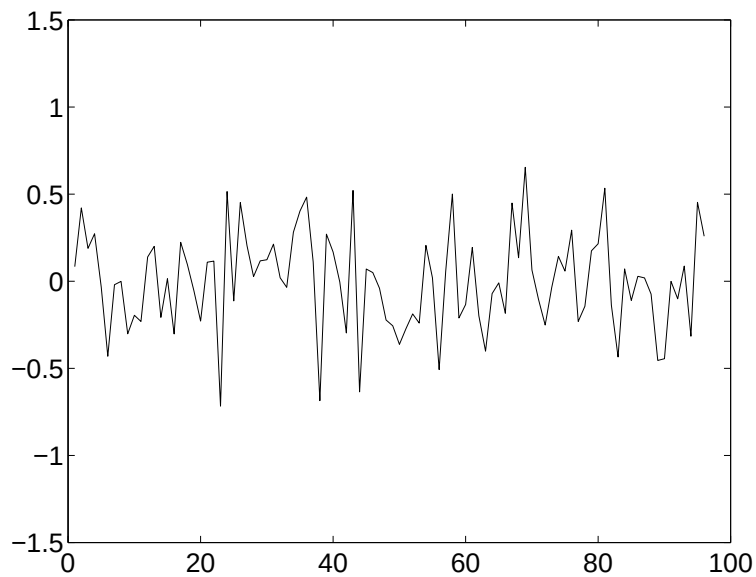
$$r_{xx}(k) = a_1 r_{xx}(k-1) + a_2 r_{xx}(k-2) + ... + a_p r_{xx}(k-p) \tag{6.13}$$

This holds for all lags. For an AR(p) model we can write this relation out for the first p lags. For $p = 4$

$$\begin{bmatrix} r_{xx}(1) \\ r_{xx}(2) \\ r_{xx}(3) \\ r_{xx}(4) \end{bmatrix} = \begin{bmatrix} r_{xx}(0) & r_{xx}(-1) & r_{xx}(-2) & r_{xx}(-3) \\ r_{xx}(1) & r_{xx}(0) & r_{xx}(-1) & r_{xx}(-2) \\ r_{xx}(2) & r_{xx}(1) & r_{xx}(0) & r_{xx}(-1) \\ r_{xx}(3) & r_{xx}(2) & r_{xx}(1) & r_{xx}(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} \tag{6.14}$$



(a)



(b)

Figure 6.3: *(a) Original signal (solid line),* $\boldsymbol{X}$*, and predictions (dotted line),* $\hat{\boldsymbol{X}}$*, from an AR(4) model and (b) the prediction errors,* $\boldsymbol{e}$*. Notice that the variance of the errors is much less than that of the original signal.*

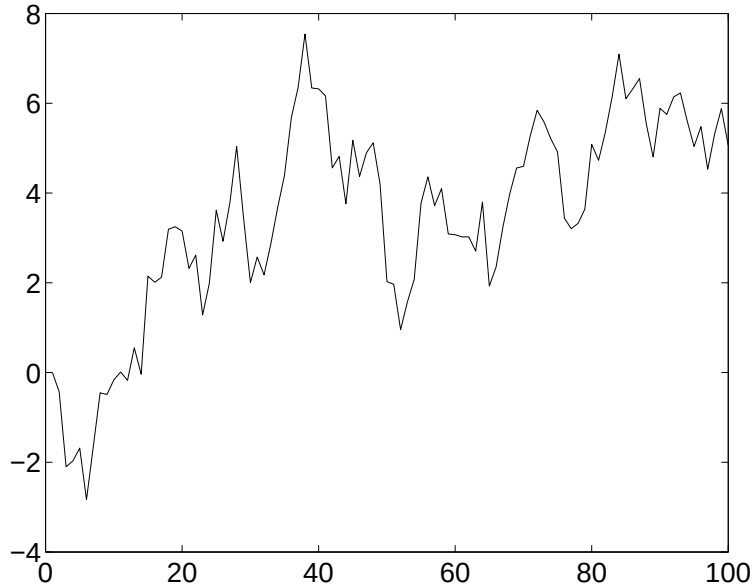


Figure 6.4: *A random walk.*

which can be compactly written as

$$\boldsymbol{r} = \boldsymbol{R}\boldsymbol{a} \tag{6.15}$$

where $\boldsymbol{r}$ is the autocorrelation vector and $\boldsymbol{R}$ is the autocorrelation matrix. The above equations are known, after their discoverers, as the Yule-Walker relations. They provide another way to estimate AR coefficients

$$\boldsymbol{a} = \boldsymbol{R}^{-1}\boldsymbol{r} \tag{6.16}$$

This leads to a more efficient algorithm than the general method for multivariate linear regression (equation 6.7) because we can exploit the structure in the autocorrelation matrix. By noting that $r_{xx}(k) = r_{xx}(-k)$ we can rewrite the correlation matrix as

$$\boldsymbol{R} = \begin{bmatrix} 1 & r_{xx}(1) & r_{xx}(2) & r_{xx}(3) \\ r_{xx}(1) & 1 & r_{xx}(1) & r_{xx}(2) \\ r_{xx}(2) & r_{xx}(1) & 1 & r_{xx}(1) \\ r_{xx}(3) & r_{xx}(2) & r_{xx}(1) & 1 \end{bmatrix} \tag{6.17}$$

Because this matrix is both symmetric and a Toeplitz matrix (the terms along any diagonal are the same) we can use a recursive estimation technique known as the Levinson-Durbin algorithm [1].

6.3.3 Relation to partial autocorrelation

The partial correlation coefficients (see lecture 2) in an AR model are known as *reflection coefficients*. At lag m, the partial correlation between x_{t-m} and x_t, is

[1] This algorithm actually operates on the autocovariance matrix, although some authors, eg. Pardey et al. [45], call it the autocorrelation matrix. What we refer to as autocorrelation, they refer to as normalised autocorrelation.

written as k_m; the mth reflection coefficient. It can be calculated as the relative reduction in prediction error

$$k_m = \frac{E_{m-1} - E_m}{E_{m-1}} \tag{6.18}$$

where E_m is the prediction error from an $AR(m)$ model [2]. The reflection coefficients are to the AR coefficients what the correlation is to the slope in a univariate AR model; if the mth reflection coefficient is significantly non-zero then so is the mth AR coefficient. And vice-versa.

The Levinson-Durbin algorithm computes reflection coefficients as part of a recursive algorithm for computing the AR coefficients. It finds k_1 and from it calculates the AR coefficient for an $AR(1)$ model, a_1. It then computes k_2 and from it calculates the AR coefficients for an AR(2) model (a_2 is computed afresh and a_1 is re-estimated from a_1 for the $AR(1)$ model - as it will be different). The algorithm continues by calculating k_m and the coefficients for $AR(m)$ from $AR(m-1)$. For details, see Pardey *et al.* [45].

6.3.4 Model order selection

Because an AR model is a special case of multivariate regression we can use the same significance testing procedure (see earlier lecture) to determine the relevance or otherwise of our variables. To recap, (i) we compute our coefficients for the AR(p) model, (ii) we estimate the standard deviation of each AR coefficient (see second lecture), (iii) we then perform a double-sided t-test to see if the smallest coefficient is significantly different from zero. If it is, then we might try a larger model order and repeat the process. If it isn't then we might try a smaller model order. We can either start with a model order of 1 and gradually increase it (stepwise forward selection) or start with a very high model order and gradually decrease it (stepwise backward selection), performing the significance test as we increase/decrease the model order.

We note that the above statistical test is identical to seeing whether or not the pth reflection coefficient is significantly non-zero (see earlier lecture).

For our data set both SFS and SBS, with a significance level of 0.05, chose $p = 3$ as the optimal model order. For SFS, for example, when $p = 4$ the smallest coefficient is $a_4 = -0.186$ and the corresponding standard deviation is $\sigma_4 = 0.103$. This gives a t-statistic of $t = -1.8006$ which under a double-sided test gives a probability of 0.0749. We therefore cannot reject the null hypothesis that the coefficient is zero at the 0.05 significance level; the SFS procedure therefore stops at a model order of 3.

Alternatively, we could use other model selection criteria eg. the Minimum Description Length (MDL) (see Lecture 4)

$$MDL(p) = \frac{N}{2} \log \sigma_e^2(p) + \frac{p}{2} \log N \tag{6.19}$$

[2] We have also written $E_m = \sigma_e^2(m)$.

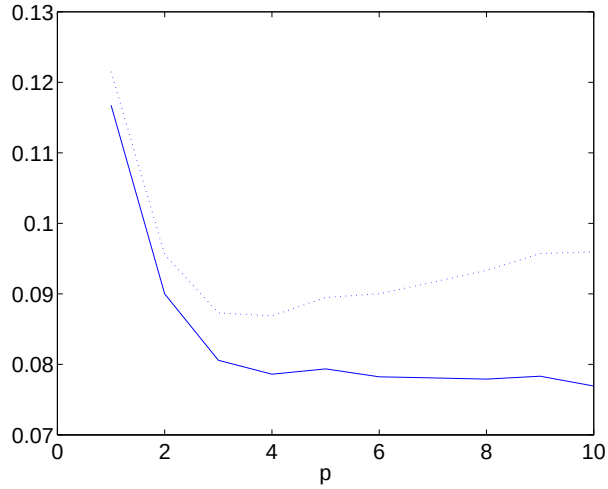


Figure 6.5: *Error variance, $\sigma_e^2(p)$, (solid line) and Final Prediction Error (FPE) (dotted line) versus AR model order, p.*

Another example is the Final Prediction Error

$$FPE(p) = \left(\frac{N+p+1}{N-p-1}\right)\sigma_e^2(p) \tag{6.20}$$

where N is the number of samples and $\sigma_e^2(p)$ is the error variance for model order p. Applying this to our data gives the results shown in Figure 6.5 showing an optimal moder order of 3 or 4.

6.3.5 Example: Sleep EEG

As a subject falls from wakefulness into a deep sleep the EEG increases in amplitude and decreases in the frequency of its oscillations. The optimal AR model order also decreases indicating a decrease in complexity. Using FPE Pardey *et al.* show, for example, that wakefulness, REM sleep and deep sleep have typical optimal model orders of 6, 5 and 3 respectively. It should be noted that these are averages and the optimal order has a high variance. Waking EEG shows the highest variance and deep sleep the least.

6.3.6 Discussion

For a comprehensive introduction to AR modelling see Pardey *at al.* [45]. This paper also contains details of other methods for estimating AR coefficients such as the Burg algorithm, which minimises both a *forwards* prediction error and a *backwards* prediction error.

6.4 Moving Average Models

A Moving Average (MA) model of order q is defined as

$$x_t = \sum_{i=0}^{q} b_i e_{t-i} \tag{6.21}$$

where e_t is Gaussian random noise with zero mean and variance σ_e^2. They are a type of FIR filter (see last lecture). These can be combined with AR models to get Autoregressive Moving Average (ARMA) models

$$x_t = \sum_{i=1}^{p} a_i x_{t-i} + \sum_{i=0}^{q} b_i e_{t-i} \tag{6.22}$$

which can be described as an ARMA(p,q) model. They are a type of IIR filter (see last lecture).

Usually, however, FIR and IIR filters have a set of fixed coefficients which have been chosen to give the filter particular frequency characteristics. In MA or ARMA modelling the coefficients are tuned to a particular time series so as to capture the spectral characteristics of the underlying process.

6.5 Spectral Estimation

Autoregressive models can also be used for spectral estimation. An $AR(p)$ model predicts the next value in a time series as a linear combination of the p previous values

$$x_t = -\sum_{k=1}^{p} a_k x_{t-k} + e_t \tag{6.23}$$

where a_k are the AR coefficients and e_t is IID Gaussian noise with zero mean and variance σ^2.

The above equation can be *solved* by assuming that the solution is in the form of an exponential

$$x_t = z^t \tag{6.24}$$

where z is, generally, a complex number. This form of solution has the property that $x_{t-i} = z^{t-i}$; effectively z^{-i} acts as a delay operator denoting a delay of i time steps. This allows the equation to be written

$$a_p z^{t-p} + a_{p-1} z^{t-(p-1)} + ... + z^t = e_t \tag{6.25}$$

It can then be rewritten

$$z^t = \frac{e_t}{1 + \sum_{k=1}^{p} a_k z^{-k}} \tag{6.26}$$

Given that any complex number can be written in exponential form

$$z = \exp(i2\pi f T_s) \tag{6.27}$$

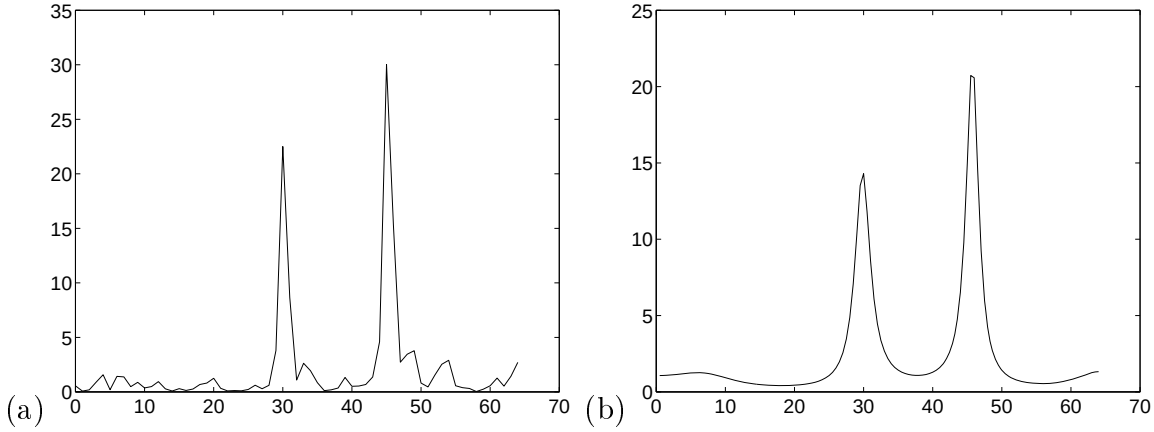


Figure 6.6: *Power spectral estimates of two sinwaves in additive noise using (a) Welch's periodogram method and (b) Autoregressive spectral estimation.*

where f is frequency and T_s is the sampling period we can see that the frequency domain characteristics of an AR model are given by (also see Pardey *et al.* [45])

$$P(f) = \frac{\sigma_e^2 T_s}{|1 + \sum_{k=1}^{p} a_k \exp(-ik2\pi f T_s)|^2} \tag{6.28}$$

An AR(p) model can provide spectral estimates with $p/2$ peaks; therefore if you know how many peaks you're looking for in the spectrum you can define the AR model order. Alternatively, AR model order estimation methods should automatically provide the appropriate level of smoothing of the estimated spectrum.

AR spectral estimation has two distinct advantages over methods based on the periodogram (last lecture) (i) power can be estimated over a continuous range of frequencies (not just at fixed intervals) and (ii) the power estimates have less variance.

Chapter 7

Multiple Time Series

7.1 Introduction

We now consider the situation where we have a number of time series and wish to explore the relations between them. We first look at the relation between cross-correlation and multivariate autoregressive models and then at the cross-spectral density and coherence.

7.2 Cross-correlation

Given *two* time series x_t and y_t we can delay x_t by T samples and then calculate the *cross-covariance* between the pair of signals. That is

$$\sigma_{xy}(T) = \frac{1}{N-1}\sum_{t=1}^{N}(x_{t-T} - \mu_x)(y_t - \mu_y) \tag{7.1}$$

where μ_x and μ_y are the means of each time series and there are N samples in each. The function $\sigma_{xy}(T)$ is the *cross-covariance* function. The *cross-correlation* is a normalised version

$$r_{xy}(T) = \frac{\sigma_{xy}(T)}{\sqrt{\sigma_{xx}(0)\sigma_{yy}(0)}} \tag{7.2}$$

where we note that $\sigma_{xx}(0) = \sigma_x^2$ and $\sigma_{yy}(0) = \sigma_y^2$ are the variances of each signal. Note that

$$r_{xy}(0) = \frac{\sigma_{xy}}{\sigma_x\sigma_y} \tag{7.3}$$

which is the correlation between the two variables. Therefore unlike the autocorrelation, r_{xy} is not, generally, equal to 1. Figure 7.1 shows two time series and their cross-correlation.

7.2.1 Cross-correlation is asymmetric

First, we re-cap as to why the auto-correlation is a *symmetric* function. The autocovariance, for a zero mean signal, is given by

$$\sigma_{xx}(T) = \frac{1}{N-1}\sum_{t=1}^{N} x_{t-T}x_t \tag{7.4}$$

This can be written in the shorthand notation

$$\sigma_{xx}(T) =< x_{t-T}x_t > \tag{7.5}$$

where the angled brackets denote the average value or *expectation*. Now, for negative lags

$$\sigma_{xx}(-T) =< x_{t+T}x_t > \tag{7.6}$$

Subtracting T from the time index (this will make no difference to the expectation) gives

$$\sigma_{xx}(-T) =< x_t x_{t-T} > \tag{7.7}$$

which is identical to $\sigma_{xx}(T)$, as the ordering of variables makes no difference to the expected value. Hence, the autocorrelation is a symmetric function.

The cross-correlation is a normalised cross-covariance which, assuming zero mean signals, is given by

$$\sigma_{xy}(T) =< x_{t-T}y_t > \tag{7.8}$$

and for negative lags

$$\sigma_{xy}(-T) =< x_{t+T}y_t > \tag{7.9}$$

Subtracting T from the time index now gives

$$\sigma_{xy}(-T) =< x_t y_{t-T} > \tag{7.10}$$

which is different to $\sigma_{xy}(T)$. To see this more clearly we can subtract T once more from the time index to give

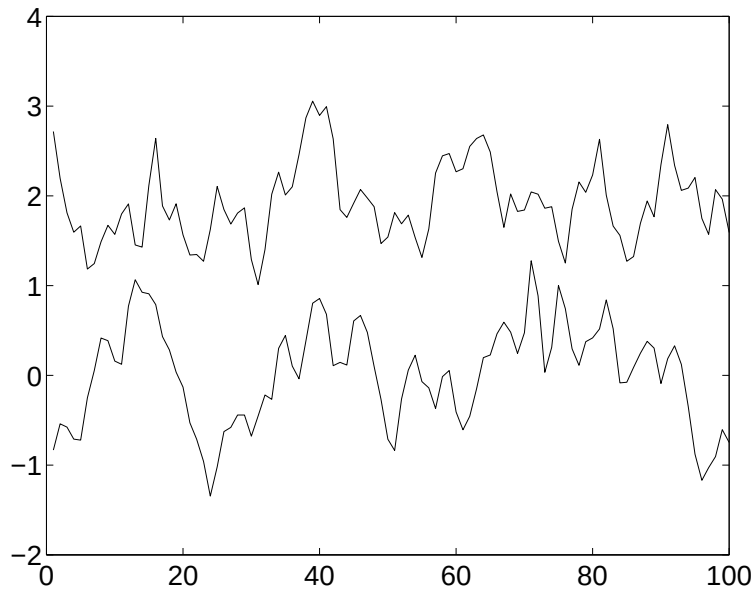
$$\sigma_{xy}(-T) =< x_{t-T}y_{t-2T} > \tag{7.11}$$

Hence, the cross-covariance, and therefore the cross-correlation, is an *asymmetric* function.

To summarise: moving signal A right (forward in time) and multiplying with signal B is not the same as moving signal A left and multiplying with signal B; unless signal A equals signal B.

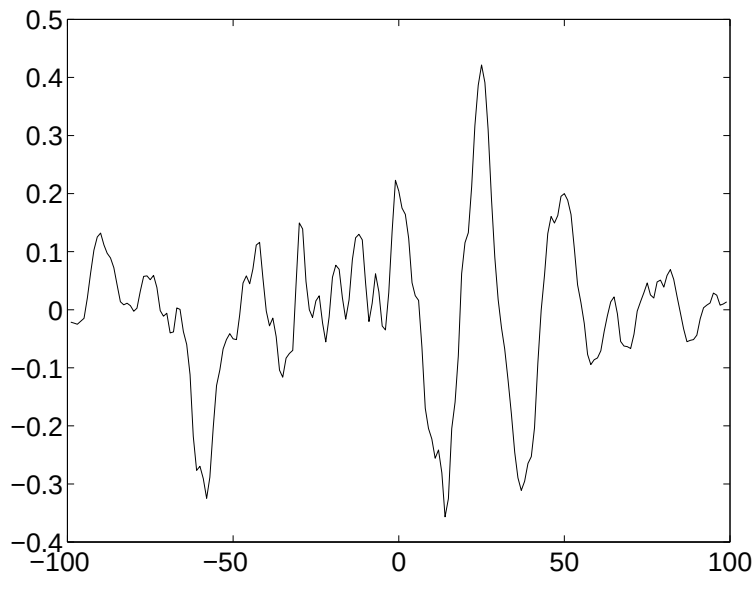
7.2.2 Windowing

When calculating cross-correlations there are fewer data points at larger lags than at shorter lags. The resulting estimates are commensurately less accurate. To take account of this the estimates at long lags can be smoothed using various window operators. See lecture 5.



(a)

Figure 7.1: *Signals x_t (top) and y_t (bottom).*



(a)

Figure 7.2: *Cross-correlation function $r_{xy}(T)$ for the data in Figure 7.1. A lag of T denotes the top series, x, lagging the bottom series, y. Notice the big positive correlation at a lag of 25. Can you see from Figure 7.1 why this should occur ?*

7.2.3 Time-Delay Estimation

If we suspect that one signal is a, possibly noisy, time-delayed version of another signal then the peak in the cross-correlation will identify the delay. For example, figure 7.1 suggests that the top signal lags the bottom by a delay of 25 samples. Given that the sample rate is 125Hz this corresponds to a delay of 0.2 seconds.

7.3 Multivariate Autoregressive models

A multivariate autoregressive (MAR) model is a linear predictor used for modelling multiple time series. An $MAR(p)$ model predicts the next vector value in a d-dimensional time series, $\boldsymbol{x}_t$ (a *row* vector) as a linear combination of the p previous vector values of the time series

$$\boldsymbol{x}(t) = \sum_{k=1}^{p} \boldsymbol{x}(t-k)\boldsymbol{a}(k) + \boldsymbol{e}_t \tag{7.12}$$

where each $\boldsymbol{a}_k$ is a $d - by - d$ matrix of AR coefficients and $\boldsymbol{e}_t$ is an IID Gaussian noise vector with zero mean and covariance $\boldsymbol{C}$. There are a total of $n_p = p \times d \times d$ AR coefficients and the noise covariance matrix has $d \times d$ elements. If we write the lagged vectors as a single augmented row vector

$$\tilde{\boldsymbol{x}}(t) = [\boldsymbol{x}(t-1), \boldsymbol{x}(t-2), ..., \boldsymbol{x}(t-p)] \tag{7.13}$$

and the AR coefficients as a single augmented matrix

$$\boldsymbol{A} = [\boldsymbol{a}(1), \boldsymbol{a}(2), ..., \boldsymbol{a}(p)]^T \tag{7.14}$$

then we can write the MAR model as

$$\boldsymbol{x}(t) = \tilde{\boldsymbol{x}}(t)\boldsymbol{A} + \boldsymbol{e}(t) \tag{7.15}$$

The above equation shows the model at a single time point t.

The equation for the model over all time steps can be written in terms of the embedding matrix, $\tilde{\boldsymbol{M}}$, whose tth row is $\tilde{\boldsymbol{x}}(t)$, the error matrix $\boldsymbol{E}$ having rows $\boldsymbol{e}(t+p+1)$ and the target matrix $\boldsymbol{X}$ having rows $\boldsymbol{x}(t+p+1)$. This gives

$$\boldsymbol{X} = \tilde{\boldsymbol{M}}\boldsymbol{A} + \boldsymbol{E} \tag{7.16}$$

which is now in the standard form of a multivariate linear regression problem. The AR coefficients can therefore be calculated from

$$\hat{\boldsymbol{A}} = \left(\tilde{\boldsymbol{M}}^T\tilde{\boldsymbol{M}}\right)^{-1}\tilde{\boldsymbol{M}}^T\boldsymbol{X} \tag{7.17}$$

and the AR predictions are then given by

$$\hat{\boldsymbol{x}}(t) = \tilde{\boldsymbol{x}}(t)\hat{\boldsymbol{A}} \tag{7.18}$$

The predicion errors are

$$\boldsymbol{e}(t) = \boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t) \tag{7.19}$$

and the noise covariance matrix is estimated as

$$\boldsymbol{C} = \frac{1}{N - n_p} \boldsymbol{e}^T(t)\boldsymbol{e}(t) \tag{7.20}$$

The denominator $N - n_p$ arises because n_p degrees of freedom have been used up to calculate the AR coefficients (and we want the estimates of covariance to be unbiased).

7.3.1 Model order selection

Given that an MAR model can be expressed as a multivariate linear regression problem all the usual model order selection criteria can be employed such as stepwise forwards and backwards selection. Other criteria also exist. Neumaier and Schneider [42] and Lutkepohl [34] investigate a number of methods including the Final Prediction Error

$$FPE(p) = \log \sigma^2 + \log \frac{N + n_p}{N - n_p} \tag{7.21}$$

where

$$\sigma^2 = \frac{1}{N} \left[det((N - n_p)\boldsymbol{C}) \right]^{1/d} \tag{7.22}$$

but they prefer the Minimum Description Length (MDL) criterion[1]

$$MDL(p) = \frac{N}{2} \log \sigma^2 + \frac{n_p}{2} \log N \tag{7.23}$$

7.3.2 Example

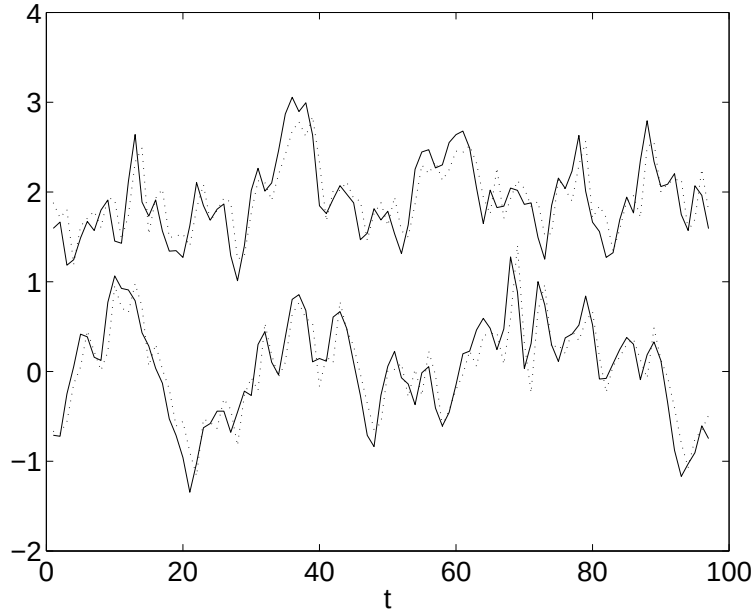
Given two time series and a MAR(3) model, for example, the MAR predictions are

$$\hat{\boldsymbol{x}}(t) = \tilde{\boldsymbol{x}}(t)\boldsymbol{A} \tag{7.24}$$

$$\hat{\boldsymbol{x}}(t) = [\boldsymbol{x}(t-1), \boldsymbol{x}(t-2), \boldsymbol{x}(t-3)] \begin{bmatrix} \boldsymbol{a}(1) \\ \boldsymbol{a}(2) \\ \boldsymbol{a}(3) \end{bmatrix}$$

$$\begin{bmatrix} \hat{x}_1(t) & \hat{x}_2(t) \end{bmatrix} = \begin{bmatrix} x_1(t-1)x_2(t-1)x_1(t-2)x_2(t-2)x_1(t-3)x_2(t-3) \end{bmatrix} \begin{bmatrix} \hat{a}_{11}(1) & \hat{a}_{12}(1) \\ \hat{a}_{21}(1) & \hat{a}_{22}(1) \\ \hat{a}_{11}(2) & \hat{a}_{12}(2) \\ \hat{a}_{21}(2) & \hat{a}_{22}(2) \\ \hat{a}_{11}(3) & \hat{a}_{12}(3) \\ \hat{a}_{21}(3) & \hat{a}_{22}(3) \end{bmatrix} \tag{7.25}$$

[1] The MDL criterion is identical to the negative value of the Bayesian Information Criterion (BIC) ie. $MDL(p) = -BIC(p)$, and Neumaier and Schneider refer to this measure as BIC.



(a)

Figure 7.3: *Signals $x_1(t)$ (top) and $x_2(t)$ (bottom) and predictions from MAR(3) model.*

Applying an MAR(3) model to our data set gave the following estimates for the AR coefficients, $\boldsymbol{a}_p$, and noise covariance $\boldsymbol{C}$, which were estimated from equations 7.17 and 7.20

$$\boldsymbol{a}_1 = \begin{bmatrix} -1.2813 & -0.2394 \\ -0.0018 & -1.0816 \end{bmatrix}$$

$$\boldsymbol{a}_2 = \begin{bmatrix} 0.7453 & 0.2822 \\ -0.0974 & 0.6044 \end{bmatrix}$$

$$\boldsymbol{a}_3 = \begin{bmatrix} -0.3259 & -0.0576 \\ -0.0764 & -0.2699 \end{bmatrix}$$

$$\boldsymbol{C} = \begin{bmatrix} 0.0714 & 0.0054 \\ 0.0054 & 0.0798 \end{bmatrix}$$

7.4 Cross Spectral Density

Just as the Power Spectral Density (PSD) is the Fourier transform of the auto-covariance function we may define the Cross Spectral Density (CSD) as the Fourier transform of the cross-covariance function

$$P_{12}(w) = \sum_{n=-\infty}^{\infty} \sigma_{x_1 x_2}(n) \exp(-iwn) \tag{7.26}$$

Note that if $x_1 = x_2$, the CSD reduces to the PSD. Now, the cross-covariance of a signal is given by

$$\sigma_{x_1 x_2}(n) = \sum_{l=-\infty}^{\infty} x_1(l) x_2(l-n) \tag{7.27}$$

Substituting this into the earlier expression gives

$$P_{12}(w) = \sum_{n=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} x_1(l) x_2(l-n) \exp(-iwn) \tag{7.28}$$

By noting that

$$\exp(-iwn) = \exp(-iwl) \exp(iwk) \tag{7.29}$$

where $k = l - n$ we can see that the CSD splits into the product of two integrals

$$P_{12}(w) = X_1(w) X_2(-w) \tag{7.30}$$

where

$$X_1(w) = \sum_{l=-\infty}^{\infty} x_1(l) \exp(-iwl) \tag{7.31}$$

$$X_2(-w) = \sum_{k=-\infty}^{\infty} x_2(k) \exp(+iwk)$$

For real signals $X_2^*(w) = X_2(-w)$ where * denotes the complex conjugate. Hence, the cross spectral density is given by

$$P_{12}(w) = X_1(w) X_2^*(w) \tag{7.32}$$

This means that the CSD can be evaluated in one of two ways (i) by first estimating the cross-covariance and Fourier transforming or (ii) by taking the Fourier transforms of each signal and multiplying (after taking the conjugate of one of them). A number of algorithms exist which enhance the spectral estimation ability of each method. These algorithms are basically extensions of the algorithms for PSD estimation, for example, for type (i) methods we can perform Blackman-Tukey windowing of the cross-covariance function and for type (ii) methods we can employ Welch's algorithm for averaging modified periodograms before multiplying the transforms. See Carter [8] for more details.

The CSD is complex

The CSD is complex because the cross-covariance is asymmetric (the PSD is real because the auto-covariance is symmetric; in this special case the Fourier transorm reduces to a cosine transform).

7.4.1 More than two time series

The frequency domain characteristics of a multivariate time-series may be summarised by the power spectral density *matrix* (Marple, 1987[39]; page 387). For d time series

$$\boldsymbol{P}(f) = \begin{pmatrix} P_{11}(f) & P_{12}(f) & \cdots & P_{1d}(f) \\ P_{12}(f) & P_{22}(f) & \cdots & P_{2d}(f) \\ \cdots & \cdots & \cdots & \cdots \\ P_{1d}(f) & P_{2d}(f) & \cdots & P_{dd}(f) \end{pmatrix} \tag{7.33}$$

where the diagonal elements contain the spectra of individual channels and the off-diagonal elements contain the cross-spectra. The matrix is called a *Hermitian matrix* because the elements are complex numbers.

7.4.2 Coherence and Phase

The *complex coherence function* is given by (Marple 1987; p. 390)

$$r_{ij}(f) = \frac{P_{ij}(f)}{\sqrt{P_{ii}(f)}\sqrt{P_{jj}(f)}} \tag{7.34}$$

The coherence, or *mean squared coherence* (MSC), between two channels is given by

$$r_{ij}(f) = \mid r_{ij}(f) \mid^2 \tag{7.35}$$

The phase spectrum, between two channels is given by

$$\theta_{ij}(f) = tan^{-1}\left[\frac{Im(r_{ij}(f))}{Re(r_{ij}(f))}\right] \tag{7.36}$$

The MSC measures the linear correlation between two time series at each frequency and is directly analagous to the squared correlation coefficient in linear regression. As such the MSC is intimately related to *linear filtering*, where one signal is viewed as a filtered version of the other. This can be interpreted as a linear regression at each frequency. The optimal regression coefficient, or linear filter, is given by

$$H(f) = \frac{P_{xy}(f)}{P_{xx}(f)} \tag{7.37}$$

This is analagous to the expression for the regression coefficient $a = \sigma_{xy}/\sigma_{xx}$ (see first lecture). The MSC is related to the optimal filter as follows

$$r^2_{xy}(f) = |H(f)|^2 \frac{P_{xx}(f)}{P_{yy}(f)} \tag{7.38}$$

which is analagous to the equivalent expression in linear regression $r^2 = a^2(\sigma_{xx}/\sigma_{yy})$.

At a given frequency, if the phase of one signal is *fixed* relative to the other, then the signals can have a high coherence at that frequency. This holds even if one signal is entirely out of phase with the other (note that this is different from adding up signals which are out of phase; the signals cancel out. We are talking about the coherence *between* the signals).

At a given frequency, if the phase of one signal changes relative to the other then the signals will not be coherent at that frequency. The time over which the phase relationship is constant is known as the *coherence time*. See [46], for an example.

7.4.3 Welch's method for estimating coherence

Algorithms based on Welch's method (such as the cohere function in the matlab system identification toolbox) are widely used [8] [55]. The signal is split up into a number of segments, N, each of length T and the segments may be overlapping. The complex coherence estimate is then given as

$$\hat{r}_{ij}(f) = \frac{\sum_{n=1}^{N} X_i^n(f)(X_j^n(f))^*}{\sqrt{\sum_{n=1}^{N} X_i^n(f)^2}\sqrt{\sum_{n=1}^{N} X_j^n(f)^2}} \tag{7.39}$$

where n sums over the data segments. This equation is exactly the same form as for estimating correlation coefficients (see chapter 1). Note that if we have only $N = 1$ data segment then the estimate of coherence will be 1 regardless of what the true value is (this would be like regression with a single data point). Therefore, we need a number of segments.

Note that this only applies to Welch-type algorithms which compute the CSD from a product of Fourier transforms. We can trade-off good spectral resolution (requiring large T) with low-variance estimates of coherence (requiring large N and therefore small T). To an extent, by increasing the overlap between segments (and therefore the amount of computation, ie. number of FFTs computed) we can have the best of both worlds.

7.4.4 MAR models

Just as the PSD can be calculated from AR coefficients so the PSD's and CSD's can be calculated from MAR coefficients. First we compute

$$\boldsymbol{A}(f) = \boldsymbol{I} + \sum_{k}^{p} \boldsymbol{a}_k \exp(-ik2\pi fT) \tag{7.40}$$

where $\boldsymbol{I}$ is the identity matrix, f is the frequency of interest and T is the sampling period. $\boldsymbol{A}(f)$ will be complex. This is analagous to the denominator in the equivalent AR expression $(1 + \sum_{k=1}^{p} a_k \exp(-ik2\pi ft))$. Then we calculate the PSD matrix as follows (Marple 1987 [39]; page 408)

$$\boldsymbol{P}_{MAR}(f) = T\left[\boldsymbol{A}(f)\right]^{-1} \boldsymbol{C} \left[\boldsymbol{A}(f)\right]^{-H} \tag{7.41}$$

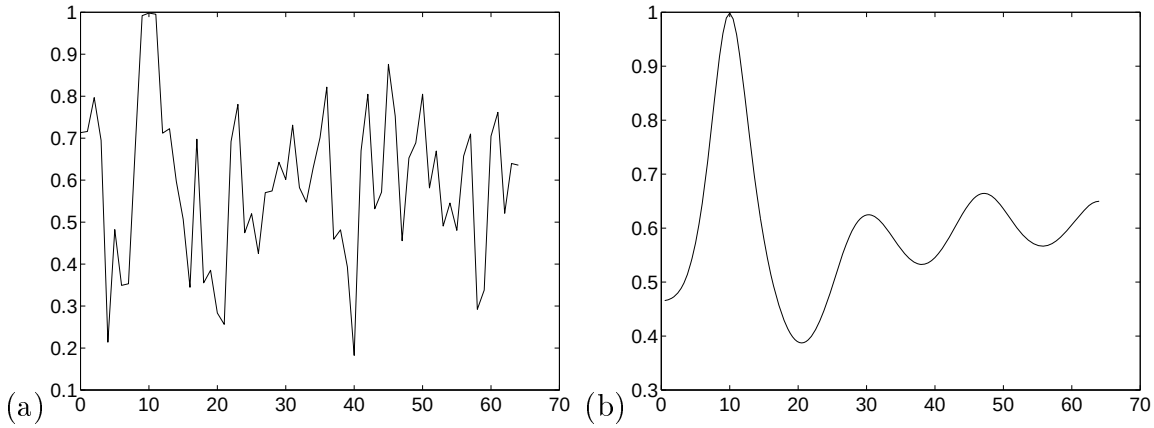


Figure 7.4: *Coherence estimates from (a) Welch's periodogram method and (b) Multivariate Autoregressive model.*

where $\boldsymbol{C}$ is the residual covariance matrix and H denotes the Hermitian transpose. This is formed by taking the complex conjugate of each matrix element and then applying the usual transpose operator.

Just as $\boldsymbol{A}^{-T}$ denotes the transpose of the inverse so $\boldsymbol{A}^{-H}$ denotes the Hermitian transpose of the inverse. Once the PSD matrix has been calculated, we can calculate the coherences of interest using equation 7.35.

7.5 Example

To illustrate the estimation of coherence we generated two signals. The first, x, being a 10Hz sine wave with additive Gaussian noise of standard deviation 0.3 and the second y being equal to the first but with more additive noise of the same standard deviation. Five seconds of data were generated at a sample rate of 128Hz. We then calculated the coherence using (a) Welch's modified periodogram method with $N = 128$ samples per segment and a 50% overlap between segments and smoothing via a Hanning window and (b) an MAR(8) model. Ideally, we should see a coherence near to 1 at 10Hz and zero elsewhere. However, the coherence is highly non-zero at other frequencies. This is because due to the noise component of the signal there is power (and some cross-power) at all frequencies. As coherence is a ratio of cross-power to power it will have a high variance unless the number of data samples is large.

You should therefore be careful when interpreting coherence values. Preferably you should perform a significance test, either based on an assumption of Gaussian signals [8] or using a Monte-Carlo method [38]. See also the text by Bloomfield [4].

7.6 Partial Coherence

There is a direct analogy to partial correlation. Given a target signal y and other signals $x_1, x_2, ..., x_m$ we can calculate the 'error' at a given frequency after including $k = 1..m$ variables $E_m(f)$. The partial coherence is

$$k_m(f) = \frac{E_{m-1}(f) - E_m(f)}{E_{m-1}(f)} \tag{7.42}$$

See Carter [8] for more details.

Chapter 8

Subspace Methods

8.1 Introduction

Principal Component Analysis (PCA) is applied to the analysis of time series data. In this context we discuss measures of complexity and subspace methods for spectral estimation.

8.2 Singular Spectrum Analysis

8.2.1 Embedding

Given a single time series x_1 to x_N we can form an *embedding* of dimension d by taking length d snapshots $\boldsymbol{x}_t = [x_t, x_{t+1}, ..., x_{t+d}]$ of the time series. We form an *embedding matrix* $\boldsymbol{X}$ with different snapshots in different rows. For $d = 4$ for example

$$\boldsymbol{X} = \frac{1}{\sqrt{N}} \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ x_2 & x_3 & x_4 & x_5 \\ .. & .. & .. & .. \\ x_{N-3} & x_{N-2} & x_{N-1} & x_N \end{bmatrix} \tag{8.1}$$

The normalisation factor is there to ensure that $\boldsymbol{X}^T\boldsymbol{X}$ produces the covariance matrix (see PCA section).

$$\boldsymbol{C} = \boldsymbol{X}^T\boldsymbol{X} \tag{8.2}$$

We note that embedding is identical to the procedure used in autoregressive modelling to generate the 'input data matrix'. Similarly, we see that the covariance matrix of embedded data is identical to the autocovariance matrix

$$\boldsymbol{C} = \begin{bmatrix} \sigma_{xx}(0) & \sigma_{xx}(1) & \sigma_{xx}(2) & \sigma_{xx}(3) \\ \sigma_{xx}(1) & \sigma_{xx}(0) & \sigma_{xx}(1) & \sigma_{xx}(2) \\ \sigma_{xx}(2) & \sigma_{xx}(1) & \sigma_{xx}(0) & \sigma_{xx}(1) \\ \sigma_{xx}(3) & \sigma_{xx}(2) & \sigma_{xx}(1) & \sigma_{xx}(0) \end{bmatrix} \tag{8.3}$$

where $\sigma_{xx}(k)$ is the autocovariance at lag k.

The application of PCA to embedded data (using either SVD on the embedding matrix or eigendecomposition on the autocovariance matrix) is known as Singular Spectrum Analysis (SSA) [18] or PCA Embedding.

8.2.2 Noisy Time Series

If we suppose that the observed time series x_n consists of a signal s_n plus additive noise e_n of variance σ_e^2 then

$$x_n = s_n + e_n \tag{8.4}$$

If the noise is uncorrelated from sample to sample (a key assumption) then the noise autocovariance matrix is equal to $\sigma_e^2 \boldsymbol{I}$. If the signal has autocovariance matrix $\boldsymbol{C}_s$ and corresponding singular values s_k then application of SVD to the observed embedding matrix will yield the singular values (see section 8.3 for a proof)

$$\sigma_k = s_k + \sigma_e \tag{8.5}$$

Thus, the biggest singular values correspond to signal plus noise and the smallest to just noise. A plot of the singular values is known as the *singular spectrum*. The value σ_e is the *noise floor*. By reconstructing the time series from only those components above the noise floor we can remove noise from the time series.

Projections and Reconstructions

To find the projection of the data onto the kth principal component we form the projection matrix

$$\boldsymbol{P} = \boldsymbol{Q}^T \boldsymbol{X}^T \tag{8.6}$$

where $\boldsymbol{Q}$ contains the eigenvectors of $\boldsymbol{C}$ ($\boldsymbol{Q}_2$ from SVD) and the kth row of $\boldsymbol{P}$ ends up containing the projection of the data onto the kth component. We can see this more clearly as follows, for $d = 4$

$$\boldsymbol{P} = \begin{bmatrix} - & - & \boldsymbol{q}_1 & - & - \\ - & - & \boldsymbol{q}_2 & - & - \\ - & - & \boldsymbol{q}_3 & - & - \\ - & - & \boldsymbol{q}_4 & - & - \end{bmatrix} \begin{bmatrix} x_1 & x_2 & . & x_{N-3} \\ x_2 & x_3 & . & x_{N-2} \\ x_3 & x_4 & . & x_{N-1} \\ x_4 & x_5 & . & x_N \end{bmatrix} \tag{8.7}$$

We can write the projection onto the kth component explicitly as

$$\boldsymbol{p}_k = \boldsymbol{q}_k^T \boldsymbol{X}^T \tag{8.8}$$

After plotting the singular spectrum and identifying the noise floor the signal can be reconstructed using only those components from the signal subspace. This is achieved by simply summing up the contributions from the first M chosen components

$$\hat{\boldsymbol{x}} = \sum_{k=1}^{M} \boldsymbol{p}_k \tag{8.9}$$

which is a row vector whose nth element, $\hat{x}_n$ contains the reconstruction of the original signal x_n.

From the section on dimensionality reduction (lecture 3) we know that the average reconstruction error will be

$$E_M = \sum_{k=M+1}^{d} \lambda_k \tag{8.10}$$

where $\lambda_k = \sigma_k^2$ and we expect that this error is solely due to the noise, which has been removed by SSA.

The overall process of projection and reconstruction amounts to a *filtering* or *denoising* of the signal. Figure 8.1 shows the singular spectrum (embedding dimension $d = 30$) of a short section of EEG. Figure 8.2 shows the original EEG data and the SSA filtered data using only the first 4 principal components.

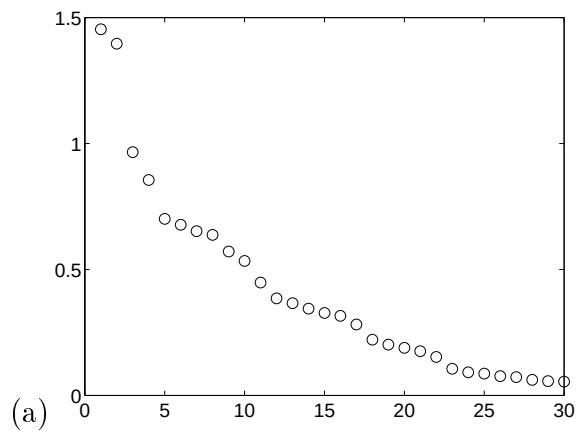


Figure 8.1: *Singular spectrum of EEG data: A plot of λ_k versus k.*

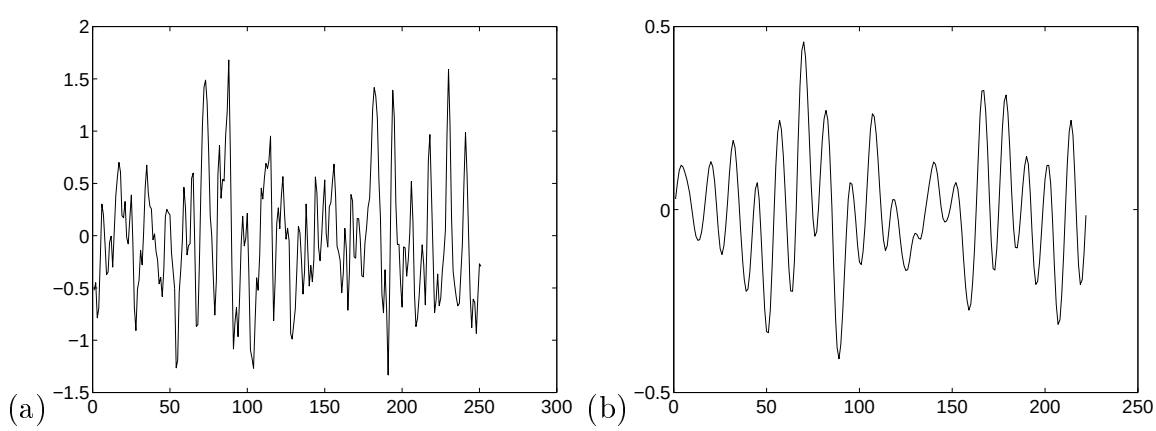


Figure 8.2: *(a) EEG data and (b) SSA-filtered EEG data.*

8.2.3 Embedding Sinewaves

A pure sinewave

If we embed a pure sinewave with embedding dimension $d = 2$ then we can view the data in the 'embedding space'. Figure 8.3 shows two such embeddings; one for a low frequency sinewave and one for a high frequency sinewave. Each plot shows that the data lie on a closed loop. There are two points to note.

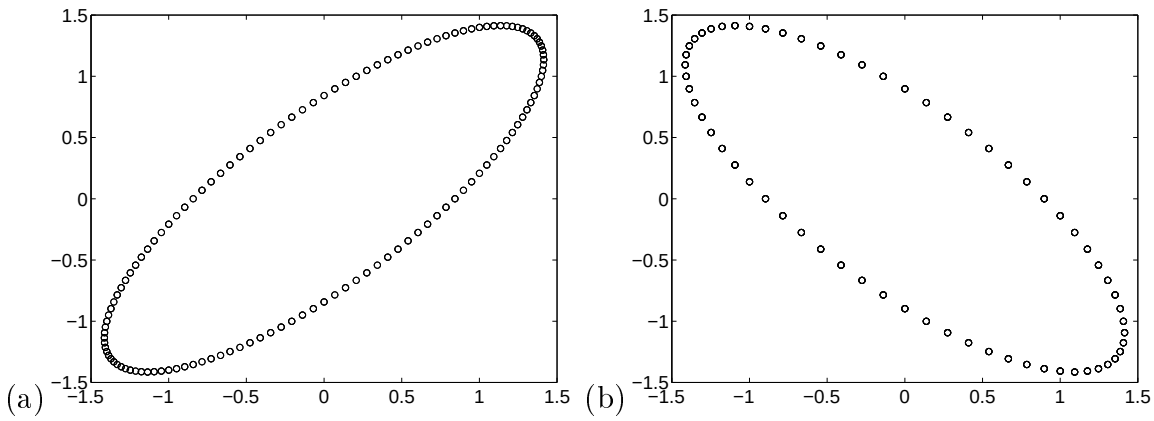


Figure 8.3: *Embedding Sinewaves: Plots of x_{n+1} versus x_n for sinewaves at frequencies of (a) 13Hz and (b) 50Hz.*

Firstly, whilst a loop is intrinsically a 1-dimensional object (any point on the loop can be described by a single number; how far round the loop from an agreed reference point) in terms on linear bases (straight lines and planes) we need *two* basis vectors. If the embedding took place in a higher dimension ($d > 2$) we would still need two basis vectors. Therefore, if we embed a pure sinewave in d dimensions the number of corresponding singular values will be 2. The remaining singular values will be zero.

Secondly, for the higher frequency signal we have fewer data points. This will become relevant when we talk about spectral estimation methods based on SVD.

Multiple sinewaves in noise

We now look at using SSA on data consisting of multiple sinusoids with additive noise. As an example we generated data from four sinusoids of different ampltidues and additive Gaussian noise. The amplitudes and frequencies were $a_1 = 2, a_2 = 4, a_3 = 3, a_4 = 1$ and $f_1 = 13, f_2 = 29, f_3 = 45, f_4 = 6$ and the standard deviation of the noise was $\sigma_e = 2$. We generated 3 seconds of data and sampled at 128Hz. We then embedded the data in dimension $d = 30$. Application of SVD yielded the singular spectrum shown in Figure 8.4; we also show the singular spectrum obtained for a data set containing just the first two sinewaves. The pairs of singular values constitutng the signal are clearly visible. Figure 8.5 shows the Power Spectral Densities (computed using Welch's modified periodogram method; see earlier) of the projections onto the

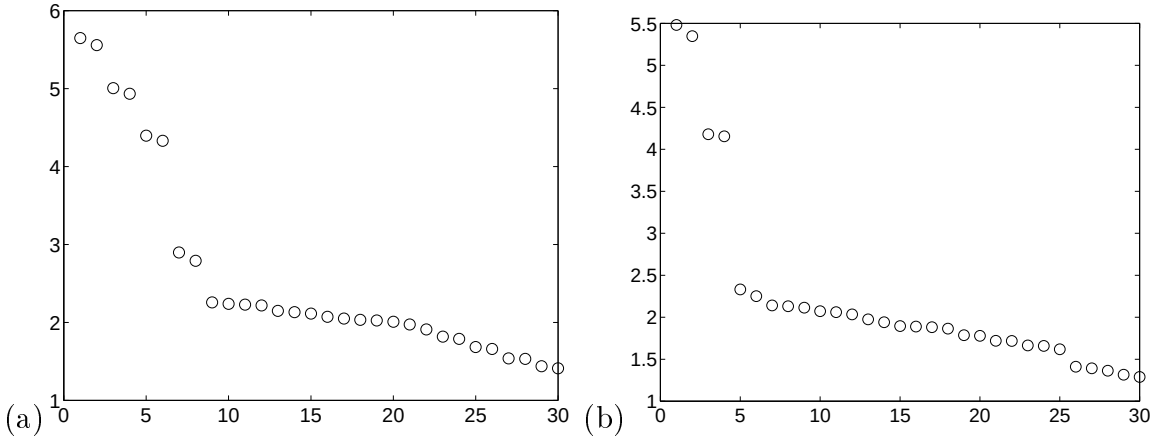


Figure 8.4: *The singular spectrums for (a) $p = 4$ and (b) $p = 2$ sinewaves in additive noise.*

first four pairs of principal components. They clearly pick out the corresponding sinewaves.

8.3 Spectral estimation

If we assume that our signal consists of p complex sinusoids

$$s_k = \exp(i2\pi f_k n) \tag{8.11}$$

where $k = 1..p$ then the signal autocovariance function, being the inverse Fourier transform of the Power Spectral Density, is

$$\sigma_{xx}(m) = \sum_{k=1}^{p} P_k \exp(i2\pi f_k m) \tag{8.12}$$

where m is the lag, P_k and f_k are the power and frequency of the kth complex sinusoid and $i = \sqrt{-1}$. If the signal embedding dimension is d, where $d > p$, then we can compute $\sigma_{xx}(m)$ for $m = 0..d-1$. The corresponding autocovariance matrix, for $d = 4$, for example is given by

$$\boldsymbol{C}_{xx} = \begin{bmatrix} \sigma_{xx}(0) & \sigma_{xx}(1) & \sigma_{xx}(2) & \sigma_{xx}(3) \\ \sigma_{xx}(1) & \sigma_{xx}(0) & \sigma_{xx}(1) & \sigma_{xx}(2) \\ \sigma_{xx}(2) & \sigma_{xx}(1) & \sigma_{xx}(0) & \sigma_{xx}(1) \\ \sigma_{xx}(3) & \sigma_{xx}(2) & \sigma_{xx}(1) & \sigma_{xx}(0) \end{bmatrix} \tag{8.13}$$

The kth sinusoidal component of the signal at these d points is given by the d-dimensional vector

$$\boldsymbol{s}_k = [1, \exp(i2\pi f_k), \exp(i4\pi f_k), ..., \exp(i2\pi(M-1)f_k)]^T \tag{8.14}$$

The autocovariance matrix can now be written as follows

$$\boldsymbol{C}_{xx} = \sum_{k=1}^{p} P_k \boldsymbol{s}_k \boldsymbol{s}_k^H \tag{8.15}$$

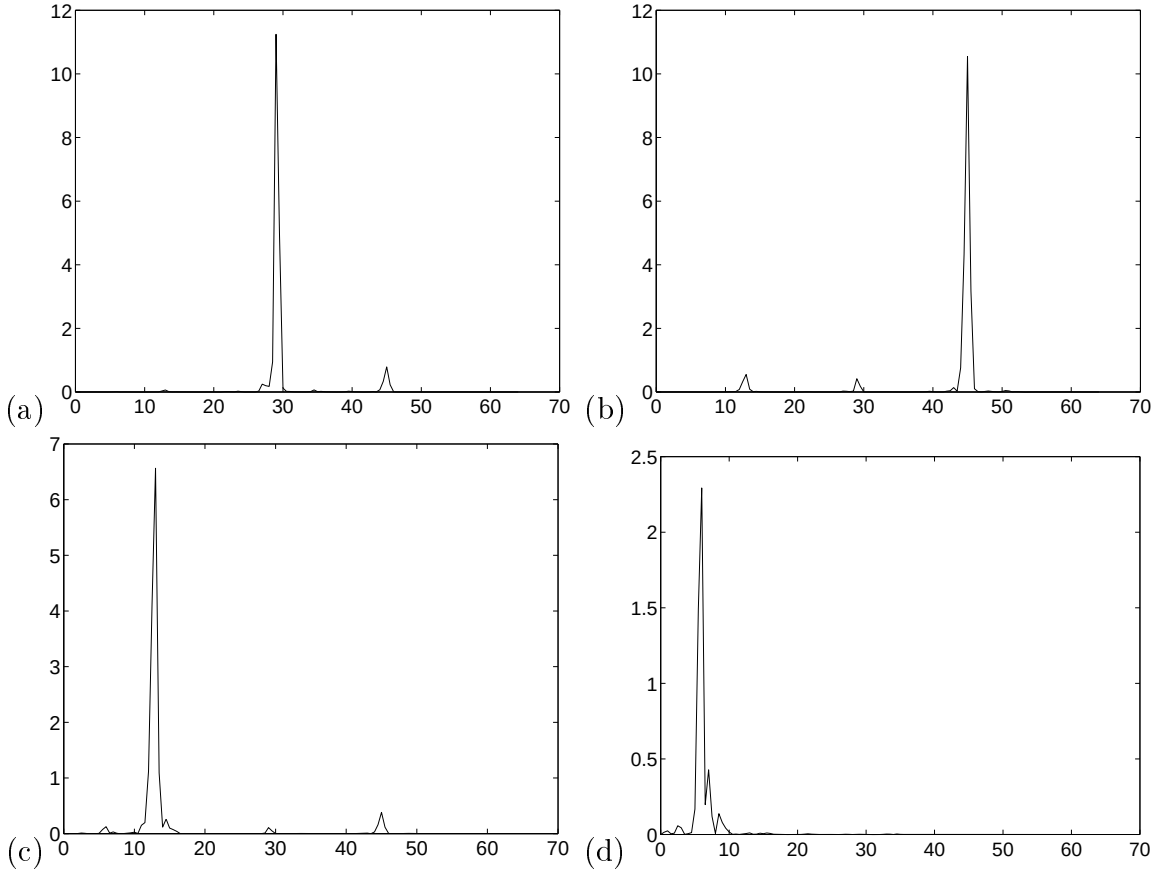


Figure 8.5: *The Power Spectral Densities of the (a) first (b) second (c) third and (d) fourth pairs of projections. They clearly correspond to the original pure sinewaves which were, in order of amplitude, of frequencies 29, 45, 13 and 6Hz. The Fourier transform of the data is the sum of the Fourier transforms of the projections.*

where H is the Hermitian transpose (take the conjugate and then the transpose).

We now model our time series as signal plus noise. That is

$$y[n] = x[n] + e[n] \tag{8.16}$$

where the noise has variance σ_e^2. The autocovariance matrix of the observed time series is then given by

$$\boldsymbol{C}_{yy} = \boldsymbol{C}_{xx} + \sigma_e^2 \boldsymbol{I} \tag{8.17}$$

We now look at an eigenanalysis of $\boldsymbol{C}_{yy}$ where the eigenvalues are ordered $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_M$ where M is the embedding dimension. The corresponding eigenvectors are $\boldsymbol{q}_k$ (as usual, they are normalised). In the absence of noise, the eigenvalues $\lambda_1, \lambda_2, .., \lambda_p$ will be non-zero while $\lambda_{p+1}, \lambda_{p+2}, .., \lambda_M$ will be zero (this is because there are only p degrees of freedom in the data - from the p sinusoids).

The signal autocovariance matrix can therefore be written as

$$\boldsymbol{C}_{xx} = \sum_{k=1}^{p} \lambda_k \boldsymbol{q}_k \boldsymbol{q}_k^H \tag{8.18}$$

(this is the usual $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^H$ eigendecomposition written as a summation) where the sum runs only over the first p components.

In the presence of noise, $\lambda_1, \lambda_2, .., \lambda_p$ and $\lambda_{p+1}, \lambda_{p+2}, .., \lambda_M$ will be non-zero. Using the orthogonality property $\boldsymbol{Q}\boldsymbol{Q}^H = \boldsymbol{I}$ we can write the noise autocovariance as

$$\sigma_e^2 \boldsymbol{I} = \sigma_e^2 \sum_{k=1}^{M} \boldsymbol{q}_k \boldsymbol{q}_k^H \tag{8.19}$$

where the sum runs over *all* M components.

Combining the last two results allows us to write the observed autocovariance matrix as

$$\boldsymbol{C}_{yy} = \sum_{k=1}^{p} (\lambda_k + \sigma_e^2) \boldsymbol{q}_k \boldsymbol{q}_k^H + \sum_{k=p+1}^{M} \sigma_e^2 \boldsymbol{q}_k \boldsymbol{q}_k^H \tag{8.20}$$

We have two sets of eigenvectors. The first p eigenvectors form a basis for the *signal subspace* while the remaining eigenvectors form a basis for the *noise subspace*. This last name is slightly confusing as the noise also appears in the signal subspace; the *signal*, however, does not appear in the noise subspace. In fact, the signal is orthogonal to the eigenvectors constituting the noise subspace. This last fact can be used to estimate the frequencies in the signal.

Suppose, for example, that $d = p + 1$. This means there will be a single vector in the noise subspace and it will be the one with the smallest eigenvalue. Now, because the signal is orthogonal to the noise we can write

$$\boldsymbol{s}_k^H \boldsymbol{q}_{p+1} = 0 \tag{8.21}$$

If we write the elements of $\boldsymbol{q}_{p+1}$ as q_{p+1}^k then we have which can be written as

$$\sum_{k=1}^{d} q_{p+1}^k \exp(-i2\pi(k-1)f_k) = 0 \tag{8.22}$$

Writing $z_k = \exp(-i2\pi k f_k)$ allows the above expression to be written in terms of a polynomial in z. The roots allow us to identify the frequencies. The amplitudes can then be found by solving the usual AR-type equation. This method of spectral estimation is known as *Pisarenko's harmonic decomposition* method.

More generally, if we have $d > p + 1$ (ie. p is unknown) then we can use the Multiple Signal Classification (MUSIC) algorithm. This is essentially the same as Pisarenko's method except that the noise variance is estimated as the average of the $d - p$ smallest eigenvalues. See Proakis [51] for more details. Figure 8.6 compares spectral estimates for the MUSIC algorithm versus Welch's method on synthetic data containing 5 pure sinusoids and additive Gaussian noise.

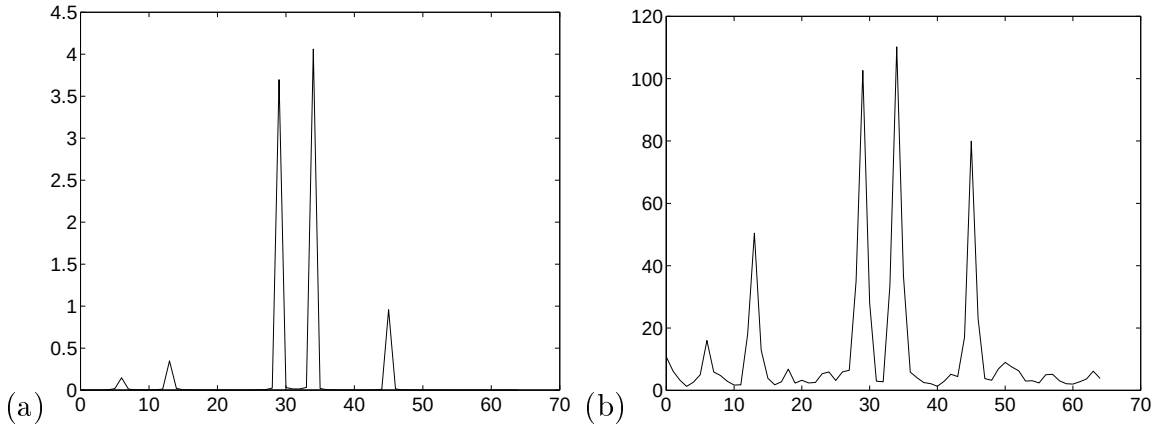


Figure 8.6: *Power Spectral Density estimates from (a) MUSIC and (b) Welch's modified periodogram.*

8.3.1 Model Order Selection

Wax and Kailath [62] suggest the Minimum Description Length (MDL) criterion for selecting p

$$MDL(p) = -N \log \left(\frac{G(p)}{A(p)} \right) + E(p) \tag{8.23}$$

where

$$\begin{aligned} G(p) &= \prod_{k=p+1}^{d} \lambda_k \\ A(p) &= \left[\frac{1}{d-p} \sum_{k=p+1}^{d} \lambda_k \right]^{d-p} \\ E(p) &= \frac{1}{2} p(2d-p) \log N \end{aligned} \tag{8.24}$$

where d is the embedding dimension, N is the number of samples and λ_k are the eigenvalues. The optimal value of p can be used as a measure of signal *complexity*.

8.3.2 Comparison of methods

Kay and Marple [31] provide a comprehensive tutorial on the various spectral estimation methods. Pardey *et. al* [45] show that the AR spectral estimates are typically better than those obtained from periodogram or autocovariance-based methods. Proakis and Manolakis (Chapter 12) [51] tend to agree, although for data containing a small number of sinusoids in additive noise, they advocate the MUSIC algorithm and its relatives.

Chapter 9

Nonlinear Methods

9.1 Introduction

This chapter covers entropy, mutual information, correlation sums, source entropy and nonlinear prediction.

To motivate the use of nonlinear methods we give a simple example of where other methods fail. Our example is the logistic map

$$x_{t+1} = Rx_t(1 - x_t) \tag{9.1}$$

which is nonlinear because of the x_t^2 term. Different values of R are known to produce different dynamics; R=3.5 and 3.6 produce periodic dynamics and R=4 produces *chaotic dynamics*. A 'chaotic' system is a low-dimensional nonlinear determnistic system which is sensitive to initial conditions. Because of the 'folding' in the logistic

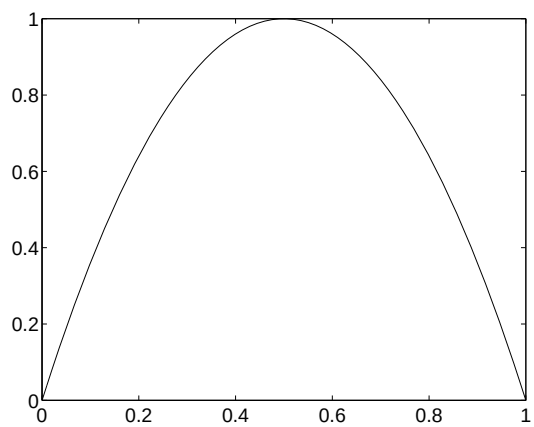


Figure 9.1: *A plot of x_{t+1} versus x_t for logistic map function $x_{t+1} = 4x_t(1 - x_t)$. If $x_{t+1} = 0.7$, then what was x_t ? Was it 0.23 or 0.77 ?*

map, for example, the system quickly forgets where its been before. Also, a slight change in the initial conditions soon leads to a big change in the subsequent state of the system.

For $R = 4$ the Power Spectral Density (PSD) is flat which is reminiscent of white noise (the corresponding autocovariance is only signficiantly non-zero at zero lag). Application of autoregressive models yields prediction errors with the same variance

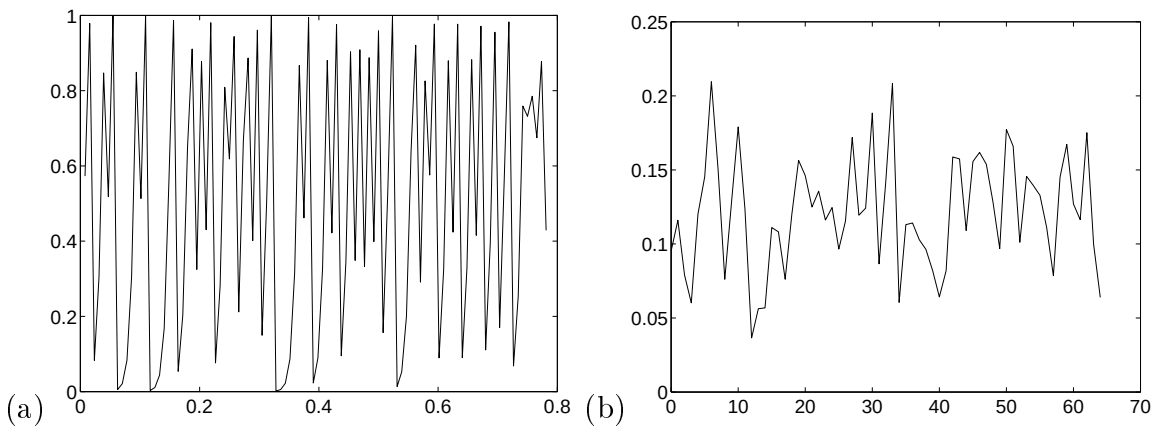


Figure 9.2: *(a) Time series from the logistic map ($R = 4$) and (b) its Power Spectral Density*

as the signal itself; ie. they are unable to detect any deterministic component in the signal. Thus, the application of linear methods would lead us to mistakenly conclude that the signal is purely stochastic when in fact it is purely deterministic.

If we apply nonlinear methods, however, then the underlying determinism can be discovered. This holds the promise of short-term predictability when, under the hypothesis of linear dynamics the system was considered to be unpredictable.

Also most early claims that physiological systems were chaotic have since been discredited. What is a more plausible working hypothesis, however, is that whilst these systems may not be nonlinear and *deterministic* they may very well be nonlinear and *stochastic*, and there is much evidence for this [23].

We look at methods for detecting nonlinear dependencies such as the *mutual information* and *marginal mutual information* and methods for exploiting these dependencies for purposes of prediction, such as *local-linear methods* and *neural networks*.

9.2 Lyapunov Exponents

A defining characteristic of a chaotic system is sensitivity to initial conditions. Points which are near at time 0 become exponentially far apart at time t. This can be captured in the relation

$$d_t = d_0 e^{\lambda t} \tag{9.2}$$

where d_0 is the initial distance, d_t is the distance at time t and λ is the *Lyapunov exponent*. Re-arranging the above equation gives

$$\lambda = \lim_{t \to \infty} \log \frac{d_t}{d_0} \tag{9.3}$$

λ_1	λ_2	λ_3	Attractor
-	-	-	Fixed Point
0	-	-	Cycle
0	0	-	Torus
+	0	-	Chaotic

Table 9.1: *Relation of sign of Lyaponov exponents to type of attractor.*

Negative λ's indicate convergence (damping) and positive λ's indicate divergence. Exponents equal to zero indicate cycles.

If the points are in a d-dimensional embedding space then neighboring points will initially be contained in a small multidimensional sphere. As time progresses this sphere will be stretched to form an ellipsoid with the length of the ith principal axis at time t given by $d_i(t)$. There is a corresponding *spectrum* of Lyapunov exponents; one for each axis. If we consider a 3-dimensional system, for example, then the relation between the signs of the Lyapunov exponents and the type of attractors is shown in Table 9.2. See [41] for more details.

The exponents can be calculate from a data set using the relation

$$\lambda_i = \lim_{t \to \infty} \log \frac{d_i(t)}{d_0} \tag{9.4}$$

Lyapunov exponents can be calculated from box-counting algorithms or from predictive models. In the last approach, for example, we can fit a neural network to the data, calculate the networks *Jacobian* matrix $\boldsymbol{J}$ (the derivative of the network's output with respect to its inputs - see Bishop [3] for details) and find λ_i from an eigendecomposition of $\boldsymbol{J}$ ([30] page 174). See also [13].

9.3 Measures of Information

See earlier lecture on Information Theory.

9.3.1 Continuous variables

In order to apply information theory to continuous variables we can partition continuous space into a number of discrete bins [1]. If we use M bins and observe n_i occurences in the ith bin then the probability of the value x_i occuring is

$$p(x_i) = \frac{n_i}{N} \tag{9.5}$$

[1] An alternative is to use a parametric model to estimate the probability density $p(\boldsymbol{x})$ from which $H(\boldsymbol{x})$ can be calculated. The entropy of such a continuous variable is known as the differential entropy [12].

where N is the total number of samples.

As we increase the number of bins, so the entropy increases.

If we have two continuous variables x and y and partition the two-dimensional space into bins where the number of levels in each dimension is M then the probability of a vector is given by

$$p(x_i, y_i) = \frac{n_{ij}}{N} \tag{9.6}$$

where there are n_{ij} samples in the i, jth bin and a total of N samples. The total number of bins will be M^2. The entropy of the above distribution is the joint entropy (see equation 4.5) and the mutual information can be calculated from 4.15. In general, these discretization procedures can be applied to d variables. But because the number of bins is M^d we need a lot of data to estimate the probabilities. As an alternative to box-counting algorithms we could use tree search algorithms or correlation sum methods (see later). See Pineda and Sommerer [48] for a review.

9.3.2 Measures of Information for Time Series

If our d continuous variables have come from a d-dimensional embedding of a time series eg.

$$\boldsymbol{x}_i = [x_i, x_{i-1}, ..., x_{i-d+1}] \tag{9.7}$$

and we partition the d-dimensional space into bins where the number of levels in each dimension is M then the probability of a vector is given by

$$p_d(\boldsymbol{x}_i) = \frac{n_i}{N - d + 1} \tag{9.8}$$

where there are n_i samples in the ith bin and a total of $N - d + 1$ samples. The total number of bins will be M^d so we need long time series to get good probability estimates.

Given a signal that has a range V the bin width will be $r = V/M$. The entropy of the above distribution is the joint entropy

$$H_d(\tau, r) = -\sum_{i=1}^{M^d} p_d(\boldsymbol{x}_i) \log p_d(\boldsymbol{x}_i) \tag{9.9}$$

where τ is the lag between samples. The *mutual information*, defined for $d = 2$, is

$$I(\tau, r) = 2H_1(\tau, r) - H_2(\tau, r) \tag{9.10}$$

It tells us about the nonlinear (or linear) correlation between $x_{t-\tau}$ and x_t and by varing τ we can plot an autocorrelation function. Figure 9.3 shows a plot of this for the logistic map time series. The entropies were calculated using a correlation sum method (see later) rather than a box-counting method. The mutual information reduces from about 4 at a lag of zero to nearly zero after 5 time steps. This makes sense as with the logistic map we lose about 1 bit of information per iteration. The

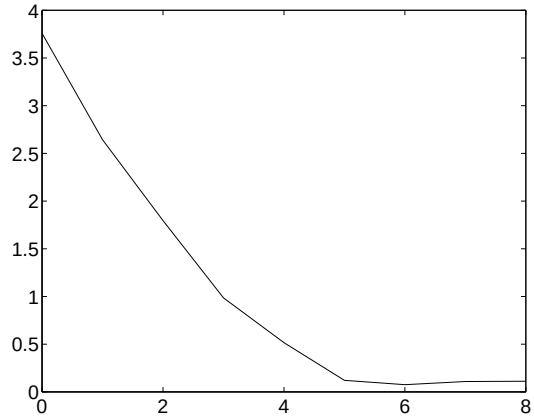


Figure 9.3: *Mutual Information, $I(\tau, r)$ versus lag τ for Logistic Map data. A resolution $r = 0.1\sigma_x$ was used where σ_x is the standard deviation of the data*

folding of the attractor acts like a switch and we lose about 1 bit of information per switch press.

For general d we can define the *joint mutual information* as the difference between the scalar entropies and the joint entropy

$$I_d(\tau, r) = dH_1(\tau, r) - H_d(\tau, r) \tag{9.11}$$

The joint mutual information measures the amount of information about x_t contained *independently* in the previous d samples ie. if we were to build a predictor, each of the previous d samples could be used but no interaction terms would be allowed.

9.3.3 Marginal Mutual Information

The joint mutual information measures the difference between the measured joint entropy of d variables and their joint entropy as if they were independent. For the special case $d = 2$ it therefore measures the amount of information about x_t contained in the previous sample $x_{t-\tau}$. For $d = 3$ and above, however, the corresponding measure is the *marginal mutual information* (or incremental mutual information or redundancy)

$$R_d(\tau, r) = I_d(\tau, r) - I_{d-1}(\tau, r) \tag{9.12}$$

We can re-write this in terms of joint entropies

$$R_d(\tau, r) = H_1(\tau, r) + H_{d-1}(\tau, r) - H_d(\tau, r) \tag{9.13}$$

Here the effect of the $d - 1$ previous variables is considered jointly (in the second term) whereas in the joint mutual information they were considered independently. The marginal mutual information, $R_d(\tau, r)$ measures the amount of information about x_t contained in the previous d samples. For $d = 2$ the marginal mutual information reduces to the mutual information.

9.3.4 Source Entropy

The *Approximate Source Entropy* statistics [47] are defined as

$$ApEn(d, r, N) = H_d(\tau, r) - H_{d-1}(\tau, r) \tag{9.14}$$

and

$$ApEn(d, r) = \lim_{N \to \infty} [H_d(\tau, r) - H_{d-1}(\tau, r)] \tag{9.15}$$

They are approximations to the *source entropy* or *KS-entropy* (from Mr. Kolmogorov and Mr Sinai) which is defined as

$$h_{KS}(\tau) = \lim_{r \to 0} \lim_{d \to \infty} ApEn(d, r) \tag{9.16}$$

Now, because of the limits, the $KS-Entropy$ can never be estimated experimentally (and, besides, it is only really of interest for purely deterministic sytems). But $ApEn$ can, and as long as the embedding dimension is large enough and the resolution fine enough it will provide a good approximation. That is,

$$h_{KS}(\tau) \approx ApEn(d, r) \tag{9.17}$$

Moreover, we can relate it to the marginal mutual information. If we substitute the above relation into equation 9.13 we get

$$R_d(\tau, r) = H_1(\tau, r) - h_{KS}(\tau) \tag{9.18}$$

Given that (see Weigend [63] page 50, or equation 9.42 later on)

$$h_{KS}(\tau) = \tau h_{KS} \tag{9.19}$$

then we have

$$R_d(\tau, r) = H_1(\tau, r) - \tau h_{KS} \tag{9.20}$$

Thus h_{KS} is the gradient of a plot of $R_d(\tau, r)$ versus τ. The d previous samples contain an amount of information $R_d(\tau, r)$ about the present sample which decreases as the time lag τ is increased. The rate of decrease is governed by the source entropy.

So, at a time lag of zero, the second term on the right is zero. The marginal mutual information is equal to the scalar entropy of the signal and the signal is completely predictable.

At each additional time step our predictive accuracy (which is governed by the marginal mutual information) loses h_{KS} bits. After a certain number of time steps, p_t, the marginal mutual information will fall to zero and all prediction accuracy will be lost.

In practice, zero prediction accuracy occurs when the the variance of the prediction error equals the variance of the signal σ_x^2. Given a prediction accuracy at zero lag of e_0 (equal to the resolution of the signal) after p_t time steps the accuracy will be

$$\sigma_x = e_0 2^{p_t h_{KS}} \tag{9.21}$$

Taking logs (to the base 2) gives

$$p_t = \frac{\log(\sigma_x/e_0)}{h_{KS}} \tag{9.22}$$

Therefore we must know the initial conditions exponentially more accurately (exponential decrease in e_0) to get a linear increase of the prediction horizon p_t. By measuring h_{KS} we can estimate the prediction horizon. Conversely, by measuring the prediction horizon, from a predictive model (see later), we can estimate h_{KS}.

9.3.5 Correlation Sums

As an alternative to box-counting algorithms we can use *correlation sums* to estimate the joint entropy (and therefore the mutual information and the source entropy). If we embed a time series in d-dimensional lag space such that

$$\boldsymbol{x}_i = [x_i, x_{i-1}, ..., x_{i-d+1}] \tag{9.23}$$

then we can measure the maximum distance between two points as

$$|\boldsymbol{x}_i - \boldsymbol{x}_j| = \max_k \{x_{i-k+1} - x_{j-k+1}\} \tag{9.24}$$

ie. look along the k out of d dimensions and pick the biggest distance. If we define the step function (or *Heaviside* function) as $h(x) = 1$ for $x \geq 0$ and $h(x) = 0$ for $x < 0$ then the indicator function

$$I_r(\boldsymbol{x}_i, \boldsymbol{x}_j) = h(r - |\boldsymbol{x}_i - \boldsymbol{x}_j|) \tag{9.25}$$

is 1 if the maximum distance between two points is less than r, and zero otherwise. We can now define the *pointwise correlation sum* as

$$C_i^d(r) = \frac{1}{N-d+1} \sum_{j=1}^{N-d+1} I_r(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{9.26}$$

which is the proportion of points within distance r of the point $\boldsymbol{x}_i$. As such this provides a good estimate for the probability density at point i

$$p_d(\boldsymbol{x}_i) = C_i^d(r) \tag{9.27}$$

The joint entropy can be approximated as the average log of this inverse probability [16]

$$H_d(r) = \frac{-1}{N-d+1} \sum_{i=1}^{N-d+1} \log p_d(\boldsymbol{x}_i) \tag{9.28}$$

Note that the sum is now over i whereas before it was over j. This method was used to calculate the mutual information in the earlier example. Now the probability $p_d(\boldsymbol{x}_i)$ can be decomposed as

$$\begin{aligned} p_d(\boldsymbol{x}_i) &= p(x_i^1, x_i^2, .., x_i^d) \\ &= p(x_i^d | x_i^1, x_i^2, .., x_i^{d-1}) p(x_i^1, x_i^2, .., x_i^{d-1}) \\ &= p(x_i^d | x_i^1, x_i^2, .., x_i^{d-1}) p_{d-1}(\boldsymbol{x}_i) \end{aligned} \tag{9.29}$$

Substituting this into the definitions for the joint entropies gives an expression for the approximate source entropy

$$ApEn(d, r, N) = \frac{-1}{N-d+1} \sum_{i=1}^{N-d+1} \log p(x_i^d | x_i^1, x_i^2, .., x_i^{d-1}) \tag{9.30}$$

Therefore, the approximate source entropy can be interpreted as the average log of a conditional probability; the probability that points are within distance r in embedding dimension d *given* that they were within this distance in embedding dimension $d-1$. Application of $ApEn$ to the logistic map shows that it is able to detect the difference between the 'simpler' periodic regime and the more complex 'chaotic' regime. Application of $ApEn$ to physiological signals is discussed in [23, 52, 47]. See Pincus

R	$ApEn$
3.5	0.0
3.6	0.229
3.8	0.425

Table 9.2: *Approximate entropy of the logistic map time series with* $d = 3$, $N = 300$, $r = 0.1\sigma_x$. *Increasing* R *increases the complexity of the time series which is reflected in higher values of ApEn.*

[47] for a discussion on how to select r.

9.4 Nonlinear Prediction

Given a time series x_n where $n = 1..N$ we wish to predict future values of the series ie x_{N+1}, x_{N+2} etc. If we view the time series up to time N as a fixed data set D then this can be achieved by inferring a statistical model from the data and using this model to predict future values of the signal.

This could, for example, be achieved by an autoregressive model which predicts the next value in the time series eg x_{N+1} as a linear combination of the p previous values

$$\hat{x}_{N+1} = w_1 x_N + w_2 x_{N-1} + ... + w_k x_{N-k+1} \tag{9.31}$$

where w_k are the autoregressive coefficients (see earlier lecture). These can be 'learnt' by tuning the model to the data set D.

This same process can be repeated but with a more powerful class of predictive models; nonlinear predictors. These replace the linear function in the above equation with a nonlinear function

$$\hat{x}_{N+1} = f(\boldsymbol{w}, x_N, x_{N-1}, .., x_{N-k+1}) \tag{9.32}$$

having parameters $\boldsymbol{w}$. Nonlinear predictors may be categorized into two broad classes (i) Local methods and (ii) Global methods.

9.4.1 Local methods

Given a data set of N embedded points $D = \{\boldsymbol{x}_n\}$ we can make a nonlinear prediction of a future time series value x_{p+T} from the embedded data point $\boldsymbol{x}_p$ as follows. Firstly, we find the k-nearest neighbours amongst D. That is, the k points in D which minimise the distance

$$||\boldsymbol{x}_n - \boldsymbol{x}_p|| \tag{9.33}$$

Put these points, $\tilde{\boldsymbol{x}}_n$, in rows of a matrix $\boldsymbol{X}$ and put the corresponding 'future' values $\tilde{x}_{n+T}$ into the vector Y. We now fit a linear model

$$\boldsymbol{Y} = \boldsymbol{w}\boldsymbol{X} \tag{9.34}$$

in the usual manner

$$\boldsymbol{w} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y} \tag{9.35}$$

and we can then use it to make the prediction

$$\hat{x}_{p+T} = \boldsymbol{w}\boldsymbol{x}_p \tag{9.36}$$

This constitutes a *local autoregressive* model since only points in the neighbourhood of the predicting region have been used. As $k \to N$ we get the usual (global) autoregresive model.

A plot of prediction error versus k shows whether a local linear model (which is globally nonlinear) or a global linear model is appropriate. These plots are known as Deterministic versus Stochastic (DVS) plots [9]. For stochastic linear dynamics $k \to N$ gives the smallest error and for deterministic nonlinear dynamics $k \to 2d+1$, where d is the dimension of the attractor, gives the smallest error. Physiological data, such as heart rate or EEG, is in-between; it varies from nonlinear-stochastic to linear stochastic.

A cautionary note in the interpretation of these plots is due to the issue of *stationarity*. This is because a nonstationary linear system may be viewed as a stationary nonlinear system. The two viewpoints are both valid descriptions of the same dynamics.

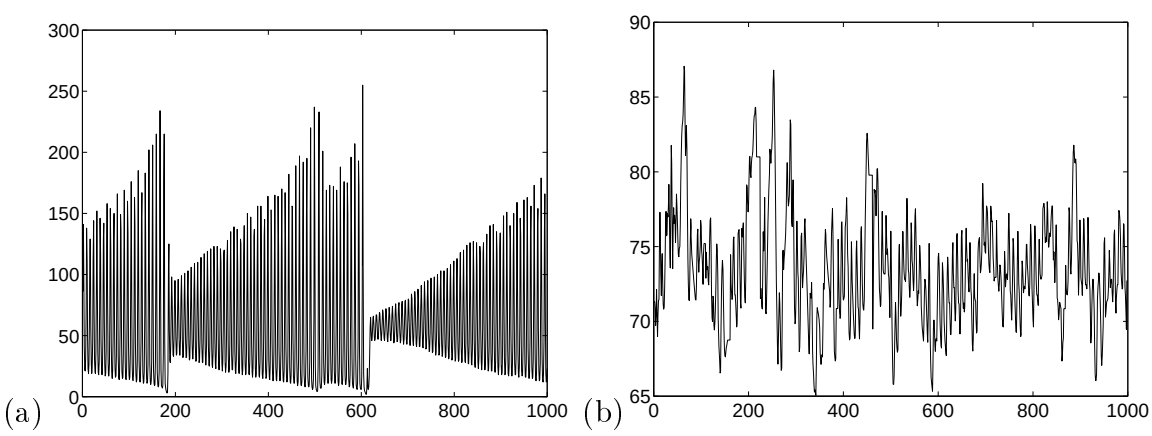


Figure 9.4: *(a) Intensity pulsations of a laser and (b) heart rate.*

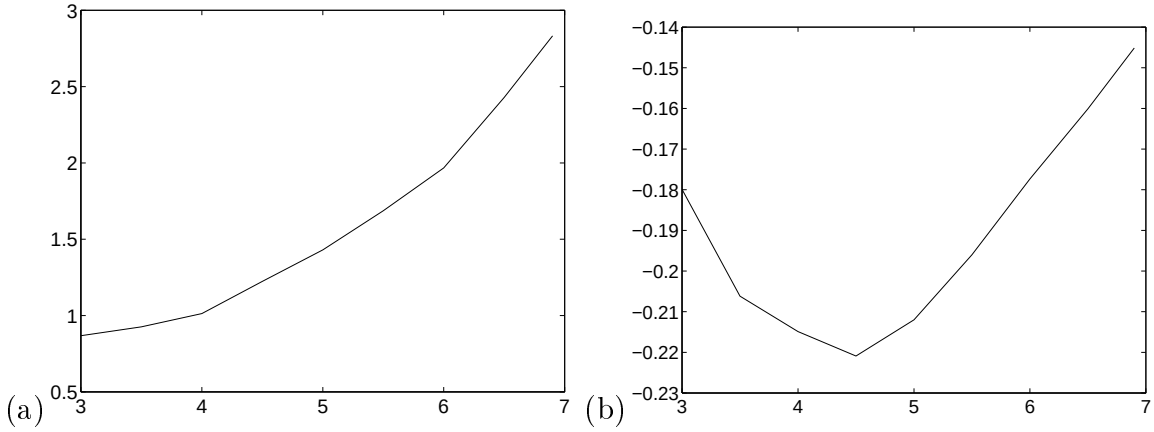


Figure 9.5: *Plots of (log) prediction error, E, versus (log) neighbourhood size, k, for (a) laser data and (b) heart-rate data. The minimum error points are at (a) $\log k = 3$, $k = 21$ and (b) $\log k = 4.5$, $k = 91$. These indicate that (a) the laser data is nonlinear and deterministic and (b) the heart-rate data is nonlinear and stochastic.*

Denoising

Not only can local methods be used for nonlinear prediction but also for nonlinear denoising. If, for example, the above linear prediction step is replaced by an SVD step we have a local-SVD denoising algorithm. This can also be used in combination with local prediction methods - see Sauer et. al in [63].

9.4.2 Global methods

Probably the most powerful nonlinear predictor is a *Neural Network* and the most commonly used network is the *Multi-Layer Perceptron* (MLP). This consists of a number of layers of processing elements (usually only two). The first layer consists of a number of linear transforms which are then operated on by a nonlinearity. There are $j = 1..p$ such functions each called a *hidden unit*

$$h_j = f(\sum_{i=1}^{d} w_{ij} x_{n-i}) \tag{9.37}$$

where i sums over the embedding and f is usually a sigmoidal nonlinearity

$$f(a) = \frac{1}{1 + e^{-a}} \tag{9.38}$$

The output of the second layer gives the networks prediction which is a linear combination of hidden unit responses

$$\hat{x}_{n+T} = \sum_{j=p}^{d} v_j h_j \tag{9.39}$$

Given a data set of of embedded vectors $\boldsymbol{x}_n$ and corresponding future values x_{n+T} (often $T = 1$) the parameters of the model can be set so as to minimise the prediction

error

$$E = \sum_{n=1}^{N} (x_{n+T} - \hat{x}_{n+T})^2 \tag{9.40}$$

This can be achieved by various non-linear optimisation algorithms. The number of hidden units can be chosen according to various model order selection criterion. See Bishop [3] for details.

Application of neural nets to some time series, eg. the laser data, shows them to be better predictors than linear methods by several orders of magnitude [63].

Other global nonlinear methods involve the use of polynomial functions or *Volterra series*. Predictions are formed from linear combinations of quadratic and higher order terms eg.

$$\hat{x}_{n+T} = w_1 x_n + w_2 x_n^2 + w_3 x_n x_{n-1} + w_4 x_{n-1} + ... \tag{9.41}$$

The number and order of such functions can be found empirically or from prior knowledge of the possible interactions.

9.5 Discusion

A nonlinear dynamical system, with or without added stochastic noise, can thus be characterised by a number of measures: (i) source entropy, (ii) prediction error and (iii) Lyapunov exponents and there are relations between them. There are also many more measures that we have'nt discussed. Most of these are relevant to nonlinear *deterministic* systems rather than nonlinear *stochastic* ones. (the most prominent being *correlation dimension* [24]).

To use them to, say, differentiate between different physiological states or experimental conditions requires not just estimating the measures themselves but also providing error bars so we can apply significance tests.

For these 'nonlinear' statistics, these most often take the form of Monte-Carlo estimates. Given a particular time series we compute our measure of interest, say $ApEn$. We then shuffle the data and recompute the statistic. If we do this for a number of shuffles then where on the resulting PDF our original value falls is the significance value.

The sum of the positive Lyapunov exponents is equal to the source entropy

$$h_{KS} = \sum_{\lambda_i > 0} \lambda_i \tag{9.42}$$

This is known as *Pesin's Identity* [2]. This completes the circle: Source Entropy → Nonlinear Prediction → Lyapunov Exponents → Source Entropy etc.

[2] In fact, it is an upper bound on the source entropy [30]

Chapter 10

Bayesian Methods

10.1 Introduction

See [33] or Box and Tiao [6] for a general introduction to Bayesian statistics and [43] for applications of Bayesian methods in signal processing.

10.2 Bayes Rule

The distribution of a variable x conditioned on a variable y is

$$p(x \mid y) = \frac{p(x, y)}{p(y)} \tag{10.1}$$

Given that $p(y \mid x)$ can be expressed similarly we can write

$$p(x \mid y) = \frac{p(y \mid x)p(x)}{p(y)} \tag{10.2}$$

which is Baye's rule. The density $p(x)$ is known as the *prior*, $p(y \mid x)$ as the *likelihood* and $p(y)$ as the *evidence* or *marginal likelihood*. Baye's rule shows how a prior distribution can be turned into a posterior distribution ie. how we update our distribution in the light of new information. To do this it is necessary to calculate the normalising term; the evidence

$$p(y) = \int p(y \mid x)p(x)dx \tag{10.3}$$

which, being an integral, can sometimes be problematic to evaluate.

10.2.1 Example

For discrete variables. Given a disease D with a prevalence of ten percent, a test for it T having a sensitivity of 95% and a specificity of 85% we have

$$p(D=1) = 0.1 \tag{10.4}$$
$$p(T=1|D=1) = 0.95 \tag{10.5}$$
$$p(T=0|D=0) = 0.85 \tag{10.6}$$

The probability that subjects who test positive for D actually have D is then given by Bayes' rule

$$p(D=1|T=1) = \frac{p(T=1|D=1)p(D=1)}{p(T=1|D=1)p(D=1)+p(T=1|D=0)p(D=0)} \tag{10.7}$$
$$= \frac{0.95 \times 0.1}{0.95 \times 0.1 + 0.15 \times 0.9} \tag{10.8}$$
$$= 0.413 \tag{10.9}$$

10.3 Gaussian Variables

A Gaussian random variable x has the probability density function (PDF)

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{-(x-\mu)^2}{2\sigma^2}\right] \tag{10.10}$$

where the mean is μ and the variance is σ^2. The inverse of the variance is known as the precision $\beta = 1/\sigma^2$. The Gaussian PDF is written in shorthand as

$$p(x) = N(x; \mu, \sigma^2) \tag{10.11}$$

If the prior is Gaussian

$$p(x) = N(x; x_0, 1/\beta_0) \tag{10.12}$$

where x_0 is the prior mean and β_0 is the prior precision and the likelihood is also Gaussian

$$p(y \mid x) = N(y; x, 1/\beta_D) \tag{10.13}$$

where the variable x is the mean of the likelihood and β_D is the data precision then the posterior distribution is also Gaussian (see eg. [33],page 37).

$$p(x \mid y) = N(x; m, 1/\beta) \tag{10.14}$$

where the mean and precision are given by

$$\beta = \beta_0 + \beta_D \tag{10.15}$$

and

$$m = \frac{\beta_0}{\beta} x_0 + \frac{\beta_D}{\beta} y \tag{10.16}$$

Thus, the posterior precision is given by the sum of the prior precision and the data precision and the posterior mean is given by the sum of the prior data mean and the new data value each weighted by their relative precisions [1].

[1] This is the same as *inverse variance* weighting where the weights sum to one.

10.3.1 Combining Estimates

This type of updating is relevant to the *sensor fusion* problem, where we have information about a variable from two different sources and we wish to combine that information.

Say, for example, we had two estimates for the amount of carbon in a given compound; method 1 estimates the percentage to be 35 ± 4 units and method 2 estimates it to be 40 ± 7 units. Before observing the second result we have a prior belief that the mean percentage is $x_0 = 35$ and the variance is $4^2 = 16$ which corresponds to a precision of $\beta_0 = 0.0625$. Whilst the first result is viewed as the prior, the second result is viewed as the 'data', which has mean $y = 40$ and precision $\beta_D = 1/7^2 = 0.0204$. Our posterior estimate for the amount of carbon is then estimated as

$$m = \frac{0.0625}{0.0829} \times 35 + \frac{0.0204}{0.0829} \times 40 = 36.2 \tag{10.17}$$

and the posterior standard deviation is 3.5. If the results of method 2 were chosen as the prior (instead of method 1) we'd get the same result.

The equation for the posterior mean can be re-arranged as

$$m = x_0 + \frac{\beta_D}{\beta}(y - x_0) \tag{10.18}$$

showing that the new estimate is the old estimate plus some fraction (which may be viewed as a learning rate) of an error term $e = y - x_0$.

10.3.2 Sequential Estimation

Also, this type of update is particularly suited to *sequential* estimation, where data comes in a sample at a time and we update our estimates at each time step. Baye's rule is perfect for this because today's posterior becomes tomorrow's prior.

Say, for example, we have a random variable x which we observe sequentially - the value at time t being x_t ie. a time series - and that we wish to estimate the mean, without storing all the data points. At time t our estimate for the mean is μ_t and our estimate for the variance is σ_t^2. Now our prior distribution for μ_t (ie. prior to observing x_t) is

$$p(\mu_t) = N(\mu_t; \mu_t, \sigma_t^2/t) \tag{10.19}$$

where the variance is given by the usual standard error formula (see lecture 1). The likelihood of the new data point is

$$p(x_t|\mu_t) = N(x_t; \mu_t, \sigma_t^2) \tag{10.20}$$

Adding the precisions to get the posterior precision gives (from equation 10.15)

$$\beta = \frac{t}{\sigma_t^2} + \frac{1}{\sigma_t^2} = \frac{t+1}{\sigma_t^2} \tag{10.21}$$

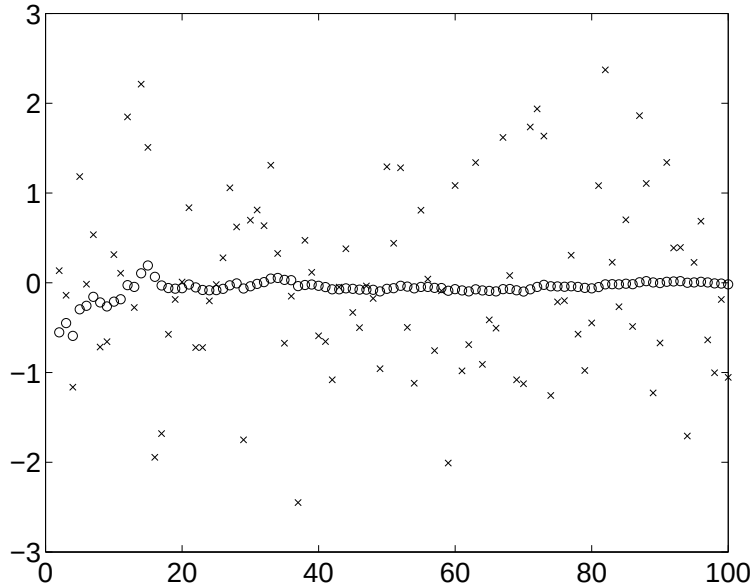


Figure 10.1: **Sequential estimation of stationary mean.** The graph plots data values x_t (crosses) and the estimated mean value μ_t (circles) versus iteration number t.

The posterior mean is then given by equation 10.16

$$\mu_{t+1} = \frac{t}{t+1}\mu_t + \frac{1}{t+1}x_t \tag{10.22}$$

Re-arranging gives

$$\mu_{t+1} = \mu_t + \frac{1}{t+1}(x_t - \mu_t) \tag{10.23}$$

In the above procedure we have implicitly assumed that the data x_t is *stationary* ie. that the mean at time t is equal to the mean at time $t+T$ for all T (a more formal definition of stationarity will be given later). This results in our estimate for the mean converging to a steady value as t increases. The final value is exactly the same as if we'd stored all the data and calculated it in the usual way.

But what if the signal is non-stationary ? See the chapter on Kalman filters.

10.4 Multiple Gaussian Variables

A d-dimensional Gaussian random vector $\boldsymbol{x}$ has a PDF given by

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{C}|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\bar{\boldsymbol{x}})^T C^{-1}(\boldsymbol{x}-\bar{\boldsymbol{x}})\right) \tag{10.24}$$

where the mean $\bar{\boldsymbol{x}}$ is a d-dimensional vector, $\boldsymbol{C}$ is a $d \times d$ covariance matrix, and $|\boldsymbol{C}|$ denotes the determinant of $\boldsymbol{C}$. The multivariate Gaussian PDF is written in shorthand as

$$p(\boldsymbol{x}) = N(\boldsymbol{x}; \bar{\boldsymbol{x}}, \boldsymbol{C}) \tag{10.25}$$

If the prior distribution is Gaussian

$$p(\boldsymbol{x}) = N(\boldsymbol{x}; \boldsymbol{x}_0, \Sigma_0) \tag{10.26}$$

where $\boldsymbol{x}_0$ is the prior mean and Σ_0 is the prior covariance and the likelihood is

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = N(\boldsymbol{y}; \boldsymbol{x}, \Sigma_D) \tag{10.27}$$

where the variable $\boldsymbol{x}$ is the mean of the likelihood and Σ_D is the data covariance then the posterior distribution is given by [40]

$$p(\boldsymbol{x} \mid \boldsymbol{y}) = N(\boldsymbol{x}; \boldsymbol{m}, \boldsymbol{\Sigma}) \tag{10.28}$$

where the mean and covariance are given by

$$\boldsymbol{\Sigma}^{-1} = \boldsymbol{\Sigma}_0^{-1} + \boldsymbol{\Sigma}_D^{-1} \tag{10.29}$$

and

$$\boldsymbol{m} = \boldsymbol{\Sigma}\boldsymbol{\Sigma}_0^{-1}\boldsymbol{x}_0 + \boldsymbol{\Sigma}\boldsymbol{\Sigma}_D^{-1}\boldsymbol{y} \tag{10.30}$$

These updates are similar in form to the updates for the univariate case. Again, these update formulae are useful for both sequential estimation and sensor fusion. In the sequential estimation case we have a *Kalman filter* (see next lecture).

10.5 General Linear Models

Given a set of input variables $\boldsymbol{z}_n$ (a row vector) where $n = 1..N$ and a fixed, possibly nonlinear, function of them

$$\boldsymbol{x}_n = F(\boldsymbol{z}_n) \tag{10.31}$$

the output variable is then given as a linear combination

$$y_n = \boldsymbol{x}_n \boldsymbol{w} + e_n \tag{10.32}$$

where $\boldsymbol{w}$ is a column vector of coefficients and e is zero mean Gaussian noise with precision β. This type of model is sufficiently general to include (i) autoregressive models if F is the identity function and $\boldsymbol{x}_n = [y_{n-1}, y_{n-2}, ..., y_{n-p}]$, (ii) Fourier-type models if F are sine and cosine functions and (iii) wavelet models if F are the wavelet bases.

Given a data set $D = \{\boldsymbol{z}_n, y_n\}$ where $n = 1..N$ the likelihood of the data is given by

$$p(D \mid \boldsymbol{w}, \beta) \quad = \quad \left(\frac{\beta}{2\pi}\right)^{N/2} \exp(-\beta E_D) \tag{10.33}$$

where

$$E_D = \frac{1}{2}(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{w})^T(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{w}) \tag{10.34}$$

and $\boldsymbol{Y}$ is a column vector with entries y_n and the nth row of the matrix $\boldsymbol{X}$ contains $\boldsymbol{x}_n$. The weights are drawn from a zero-mean Gaussian prior with an isotropic covariance having precision α

$$p(\boldsymbol{w} \mid \alpha) = \left(\frac{\alpha}{2\pi}\right)^{p/2} \exp(-\alpha E_W) \tag{10.35}$$

where

$$\begin{aligned} E_W &= \frac{1}{2}\sum_{i=1}^{p} w_i^2 \\ &= \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} \end{aligned} \tag{10.36}$$

The posterior distribution over the unknown coefficients is then given by Bayes' rule

$$p(\boldsymbol{w}|D,\alpha,\beta) = \frac{p(D|\boldsymbol{w},\beta)p(\boldsymbol{w}|\alpha)}{\int p(D|\boldsymbol{w},\beta)p(\boldsymbol{w}|\alpha)d\boldsymbol{w}} \tag{10.37}$$

As the prior is normal with mean $\boldsymbol{w}_0 = \boldsymbol{0}$ and covariance $\boldsymbol{\Sigma}_0 = (1/\alpha)\boldsymbol{I}$, the likelihood is normal with mean $\boldsymbol{w}_D = \boldsymbol{X}^{-1}\boldsymbol{Y}$ and covariance $\boldsymbol{\Sigma}_D = (\beta\boldsymbol{X}^T\boldsymbol{X})^{-1}$ then the posterior is also a normal with mean and covariance given by equations 10.30 and 10.29. The posterior is therefore given by

$$p(\boldsymbol{w}|D,\alpha,\beta) = N(\boldsymbol{w};\hat{\boldsymbol{w}},\hat{\boldsymbol{\Sigma}}) \tag{10.38}$$

where

$$\begin{aligned} \hat{\boldsymbol{\Sigma}} &= (\beta\boldsymbol{X}^T\boldsymbol{X} + \alpha\boldsymbol{I})^{-1} \\ \hat{\boldsymbol{w}} &= \hat{\boldsymbol{\Sigma}}\boldsymbol{X}^T\beta\boldsymbol{Y} \end{aligned} \tag{10.39}$$

10.5.1 The evidence framework

If the 'hyperparameters' α and β are unknown (they almost always are) they can be set according to following method known as either the *evidence framework* [35] or *Maximimum Likelihood II (ML II)* [2]. In this approach α and β are set so as to maximise the evidence (also known as marginal likelihood)

$$p(D|\alpha,\beta) = \int p(D|\boldsymbol{w},\beta)p(\boldsymbol{w}|\alpha)d\boldsymbol{w} \tag{10.40}$$

Substituting in our earlier expressions for the prior and likelihood gives

$$p(D|\alpha,\beta) = \left(\frac{\beta}{2\pi}\right)^{-N/2}\left(\frac{\alpha}{2\pi}\right)^{-p/2}\int \exp(-E(\boldsymbol{w}))d\boldsymbol{w} \tag{10.41}$$

where

$$E(\boldsymbol{w}) = \beta E_D + \alpha E_w \tag{10.42}$$

Bishop shows that ([3], page 398 and further details in Appendix B) the integral in equation 10.41 can be evaluated as

$$\int \exp(-E(\boldsymbol{w}))d\boldsymbol{w} = (2\pi)^{p/2}|\boldsymbol{\Sigma}|^{1/2}\exp(-E(\boldsymbol{w})) \tag{10.43}$$

The log of the evidence can then be written as

$$EV(p) = -\alpha E_W - \beta E_D + 0.5\log|\boldsymbol{\Sigma}| + \frac{p}{2}\log\alpha + \frac{N}{2}\log\beta - \frac{N}{2}\log 2\pi \tag{10.44}$$

The values of α and β which maximise the evidence are

$$\alpha = \frac{\gamma}{2E_W} \tag{10.45}$$

$$\beta = \frac{N-\gamma}{2E_D} \tag{10.46}$$

where γ, the number of 'well-determined' coefficients, is given by

$$\gamma = p - \alpha Trace(\boldsymbol{\Sigma}) \tag{10.47}$$

which is calculated using the 'old' value of α. The update for α is therefore an implicit equation. We can also write it as the explicit update

$$\alpha = \frac{p}{2E_W + Trace(\boldsymbol{\Sigma})} \tag{10.48}$$

See Bishop ([3], chapter 10) or Mackay [35] for a derivation of the above equations.

To summarise, the evidence framework works as follows. The weights are first estimated using equation 10.40. The hyperparmeters are then estimated using equations 10.46 and 10.48. This weights are then re-estimated and so are the hyperparameters until the procedure converges. This usually takes ten or so cycles.

Once the above procedure has converged we can use the evidence as a model order selection criterion.

10.5.2 Example

The following figures compare the MDL and Bayesian Evidence model order selection criteria. The first figure shows that, for low model order (relative to the number of data samples) both methods work equally well. The second figure shows that, at high model order, the Bayesian evidence is superior. The last figure shows that EEG recordings from an awake subject can be differentiated from those of an anaesthetised subject. Differentiation was good using the Bayesian evidence criterion but insignificant using MDL.

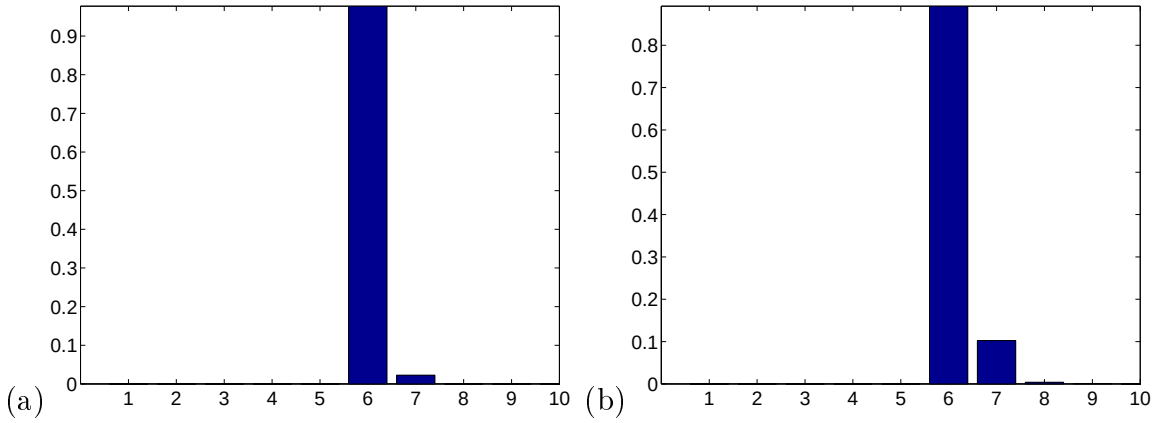


Figure 10.2: Model Order Selection for AR(6) data with (a) MDL and (b) Bayesian Evidence with 3-second blocks of data.

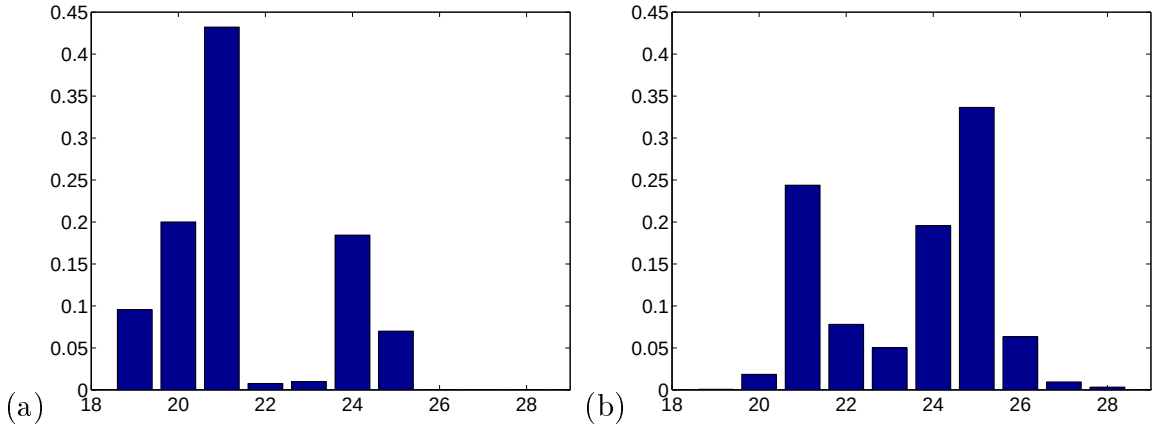


Figure 10.3: Model Order Selection for AR(25) data with (a) MDL and (b) Bayesian Evidence with 3-second blocks of data.

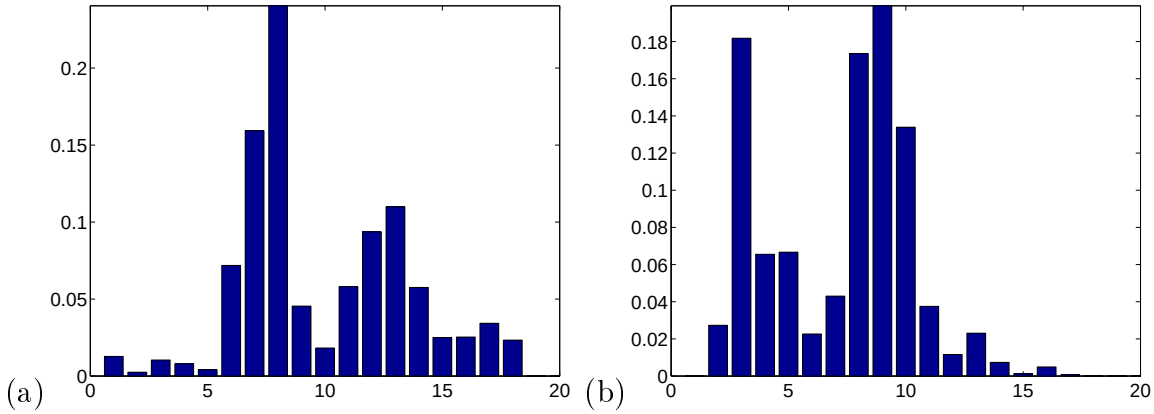


Figure 10.4: Bayesian Evidence model order selection on EEG data from (a) awake subject and (b) anaesthetised subject.

Chapter 11

Kalman Filters

11.1 Introduction

We describe Bayesian Learning for sequential estimation of parameters (eg. means, AR coefficients). The update procedures are known as Kalman Filters. We show how Dynamic Linear Models, Recursive Least Squares and Steepest Descent algorithms are all special cases of the Kalman filter.

11.1.1 Sequential Estimation of Nonstationary Mean

In the lecure on Bayesian methods we described the sequential estimation of a stationary mean. We now extend that analysis to the nonstationary case.

A reasonable model of a time varying mean is that it can drift from sample to sample. If the drift is random (later on we will also consider deterministic drifts) then we have

$$\mu_t = \mu_{t-1} + w_t \tag{11.1}$$

where the random drift is Gaussian $p(w_t) = N(w_t; 0, \sigma_w^2)$ with drift variance σ_w^2. The data points are then Gaussian about mean μ_t. If they have a *fixed* variance σ_x^2 (later on we will also consider time-varing variance)

$$x_t = \mu_t + e_t \tag{11.2}$$

where $e_t = x_t - \mu_t$. Hence $p(e_t) = N(e_t; 0, \sigma_x^2)$.

At time $t-1$ our estimate of μ_{t-1} has a Gaussian distribution with mean $\hat{\mu}_{t-1}$ and variance $\hat{\sigma}_{t-1}^2$. We stress that this is the variance of our mean estimate and not the variance of the data. The standard error estimate for this variance (σ_t^2/t) is no longer valid as we have nonstationary data. We therefore have to estimate it as we go along.

This means we keep running estimates of the distribution of the mean. At time $t-1$ this distribution has a mean $\hat{\mu}_{t-1}$ and a variance $\hat{\sigma}_{t-1}^2$. The distribution at time t is

then found from Bayes rule. Specifically, the prior distribution is given by

$$p(\mu_t) = N(\mu_t; \hat{\mu}_{t-1}, r_t) \tag{11.3}$$

where r_t is the prior variance (we add on the random drift variance to the variance from the previous time step)

$$r_t = \hat{\sigma}^2_{t-1} + \sigma^2_w \tag{11.4}$$

and the likelihood is

$$p(x_t|\mu_t) = N(x_t; \hat{\mu}_{t-1}, \sigma^2_x) \tag{11.5}$$

The posterior is then given by

$$p(\mu_t|x_t) = N(\mu_t; \hat{\mu}_t, \hat{\sigma}^2_t) \tag{11.6}$$

where the mean is

$$\hat{\mu}_t = \hat{\mu}_{t-1} + \frac{r_t}{\sigma^2_x + r_t}(x_t - \hat{\mu}_{t-1}) \tag{11.7}$$

and the variance is

$$\hat{\sigma}^2_t = \frac{r_t \sigma^2_x}{r_t + \sigma^2_x} \tag{11.8}$$

We now write the above equations in a slightly different form to allow for comparison with later estimation procedures

$$\begin{aligned} \hat{\mu}_t &= \hat{\mu}_{t-1} + K_t e_t \\ \hat{\sigma}^2_t &= r_t(1 - K_t) \end{aligned} \tag{11.9}$$

where

$$K_t = \frac{r_t}{\sigma^2_x + r_t} \tag{11.10}$$

and

$$e_t = x_t - \hat{\mu}_{t-1} \tag{11.11}$$

In the next section we will see that our update equations are a special case of a *Kalman filter* where e_t is the prediction error and K_t is the *Kalman gain*.

In figure 11.1 we give a numerical example where 200 data points were generated; the first 100 having a mean of 4 and the next 100 a mean of 10. The update equations have two paramaters which we must set (i) the data variance σ^2_x and (ii) the drift variance σ^2_w. Together, these parameters determine (a) how responsive the tracking will be and (b) how stable it will be. The two plots are for two different values of σ^2_w and $\sigma^2_x = 1$. Later we will see how these two parameters can be learnt.

11.1.2 A single state variable

We now look at a general methodology for the sequential estimation of a nonstationary parameter (this can be anything - not necesarily the data mean).

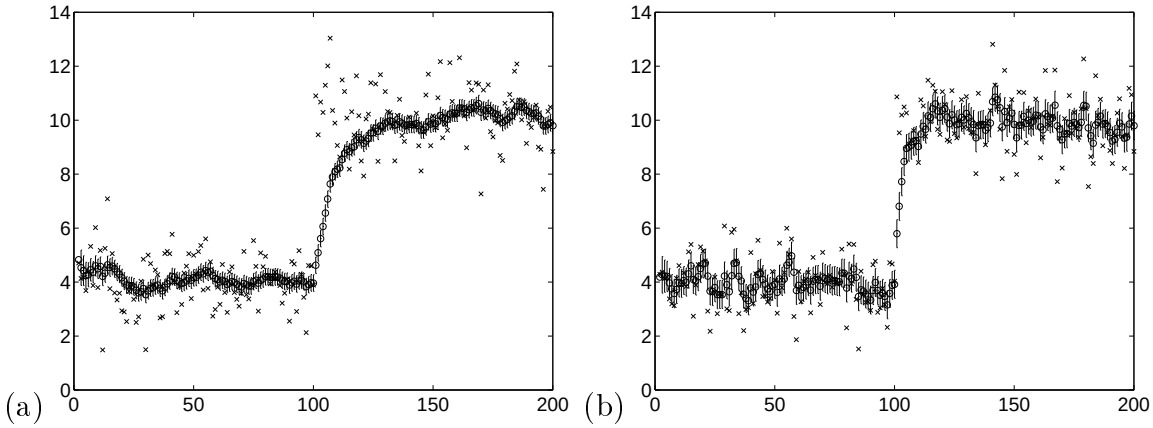


Figure 11.1: **Sequential estimation of nonstationary mean.** The graphs plot data values x_t (crosses) and estimated mean values $\hat{\mu}_t$ (circles) along with error bars $\hat{\sigma}_t$ (vertical lines) versus iteration number t for two different drift noise values (a) $\sigma_w^2 = 0.01$ and (b) $\sigma_w^2 = 0.1$.

The parameter's evolution is modelled as a *linear dynamical system.* The *state-space* equations are

$$\begin{array}{rcl} \theta_t & = & g_t\theta_{t-1} + w_t, \qquad w_t \sim N(w_t; 0, \sigma_w^2) \\ x_t & = & f_t\theta_t + e_t, \qquad e_t \sim N(e_t; 0, \sigma_x^2) \end{array} \tag{11.12}$$

The value of the parameter at time t is referred to as the *state* of the system θ_t. This state can change deterministically, by being multiplied by g_t, and stochastically by added a random drift w_t. This drift is referred to as *state noise.* The observed data (eg. time series values) are referred to as *observations* x_t which are generated from the state according to the second equation. This allows for a linear transformation plus the addition of *observation noise.*

At time $t-1$ our estimate of θ_{t-1} has a Gaussian distribution with mean $\hat{\theta}_{t-1}$ and variance $\hat{\sigma}_{t-1}^2$. The prior distribution is therefore given by

$$p(\theta_t) = N(\theta_t; g_t\hat{\theta}_{t-1}, r_t) \tag{11.13}$$

where r_t is the prior variance

$$r_t = g_t^2\hat{\sigma}_{t-1}^2 + \sigma_w^2 \tag{11.14}$$

and the likelihood is

$$p(x_t|\theta_t) = N(x_t; f_t\hat{\theta}_{t-1}, \sigma_x^2) \tag{11.15}$$

The posterior is then given by

$$p(\theta_t|x_t) = N(\theta_t; \hat{\theta}_t, \hat{\sigma}_t^2) \tag{11.16}$$

where

$$\begin{array}{rcl} \hat{\theta}_t & = & g_t\hat{\theta}_{t-1} + K_t e_t \\ \hat{\sigma}_t^2 & = & r_t(1 - K_t f_t) \end{array} \tag{11.17}$$

and

$$K_t = \frac{r_t}{\sigma_x^2 + f_t^2 r_t} f_t \tag{11.18}$$

The above equations constitute a 1-dimensional *Kalman Filter* (the state is 1-dimensional because there is only 1 state variable). Next we consider many state variables.

11.1.3 Multiple state variables

We now consider linear dynamical systems where data is generated according to the model

$$\begin{aligned} \boldsymbol{\theta}_t &= \boldsymbol{G}_t\boldsymbol{\theta}_{t-1} + \boldsymbol{w}_t, & \boldsymbol{w}_t &\sim N(\boldsymbol{w}_t; 0, \boldsymbol{W}_t) \\ \boldsymbol{y}_t &= \boldsymbol{F}_t\theta_t + \boldsymbol{v}_t, & \boldsymbol{v}_t &\sim N(\boldsymbol{v}_t; 0, \boldsymbol{V}_t) \end{aligned} \tag{11.19}$$

where $\boldsymbol{\theta}_t$ are 'state' or 'latent' variables, $\boldsymbol{G}_t$ is a 'flow' matrix, $\boldsymbol{w}_t$ is 'state noise' distributed according to a normal distribution with zero mean and covariance matrix $\boldsymbol{W}_t$, $\boldsymbol{y}_t$ are the multivariate observations, $\boldsymbol{F}_t$ is a transformation matrix and $\boldsymbol{v}_t$ is 'observation noise' distributed according to a normal distribution with zero mean and covariance matrix $\boldsymbol{V}_t$. The model is parameterised by the matrices $\boldsymbol{G}_t$, $\boldsymbol{W}_t$, $\boldsymbol{F}_t$ and $\boldsymbol{V}_t$. These parameters may depend on t (as indicated by the subscript).

The Kalman filter is a recursive procedure for estimating the latent variables, $\boldsymbol{\theta}_t$ [29]. Meinhold and Singpurwalla [40] show how this estimation procedure is derived (also see lecture on Bayesian methods). The latent variables are normally distributed with a mean and covariance that can be estimated with the following recursive formulae

$$\begin{aligned} \hat{\boldsymbol{\theta}}_t &= \boldsymbol{G}_t\hat{\boldsymbol{\theta}}_{t-1} + \boldsymbol{K}_t\boldsymbol{e}_t \\ \boldsymbol{\Sigma}_t &= \boldsymbol{R}_t - \boldsymbol{K}_t\boldsymbol{F}_t\boldsymbol{R}_t \end{aligned} \tag{11.20}$$

where $\boldsymbol{K}_t$ is the 'Kalman gain' matrix, $\boldsymbol{e}_t$ is the prediction error and $\boldsymbol{R}_t$ is the 'prior covariance' of the latent variables (that is, prior to $\boldsymbol{y}_t$ being observed). These quantities are calculated as follows

$$\begin{aligned} \boldsymbol{K}_t &= \boldsymbol{R}_t\boldsymbol{F}_t^T\left(\boldsymbol{V}_t + \boldsymbol{F}_t\boldsymbol{R}_t\boldsymbol{F}_t^T\right)^{-1} \\ \boldsymbol{e}_t &= \boldsymbol{y}_t - \boldsymbol{F}_t\boldsymbol{G}_t\hat{\boldsymbol{\theta}}_{t-1} \\ \boldsymbol{R}_t &= \boldsymbol{G}_t\boldsymbol{\Sigma}_{t-1}\boldsymbol{G}_t^T + \boldsymbol{W}_t \end{aligned} \tag{11.21}$$

To apply these equations you need to know the parameters $\boldsymbol{G}_t$, $\boldsymbol{W}_t$,$\boldsymbol{F}_t$ and $\boldsymbol{V}_t$ and make initial guesses for the state mean and covariance; $\hat{\boldsymbol{\theta}}_0$ and $\boldsymbol{\Sigma}_0$. Equations (3) and (2) can then be applied to estimate the state mean and covariance at the next time step. The equations are then applied recursively.

A useful quantity is the likelihood of an observation given the model parameters before they are updated

$$p(\boldsymbol{y}_t) = N\left(\boldsymbol{y}_t; \boldsymbol{F}_t\hat{\boldsymbol{\theta}}_{t-1}, \boldsymbol{V}_t + \boldsymbol{F}_t\left(\boldsymbol{G}_t^T\boldsymbol{\Sigma}_{t-1}\boldsymbol{G}_t\right)\boldsymbol{F}_t^T\right) \tag{11.22}$$

In Bayesian terminology this likelihood is known as the evidence for the data point [14]. Data points with low evidence correspond to periods when the statistics of the underlying system are changing (non-stationarity) or, less consistently, to data points having large observation noise components.

The state-space equations may be viewed as a dynamic version of factor analysis where the factor, $\boldsymbol{\theta}_t$, evolves over time according to linear dynamics. Shumway and Stoffer [56] derive an Expectation-Maximisation (EM) algorithm (see next lecture) in which the parameters of the model $\boldsymbol{G}$, $\boldsymbol{W}$ and $\boldsymbol{V}$ can all be learnt. Only $\boldsymbol{F}$ is assumed known. Note that these parameters are no longer dependent on t. This does not, however, mean that the model is no longer dynamic; the state, $\boldsymbol{\theta}_t$, is still time dependent. Ghahramani and Hinton [22] have recently extended the algorithm to allow $\boldsymbol{F}$ to be learnt as well. These learning algorithms are batch learning algorithms rather than recursive update procedures. They are therefore not suitable for 'on-line' learning (where the learning algorithm has only one 'look' at each observation).

In the engineering and statistical forecasting literature [44] [11] the transformation matrix, $\boldsymbol{F}_t$, is known. It is related to the observed time series (or other observed time series) according to a known deterministic function set by the statistician or 'model builder'. Assumptions are then made about the flow matrix, $\boldsymbol{G}_t$. Assumptions are also made about the state noise covariance, $\boldsymbol{W}_t$, and the observation noise covariance, $\boldsymbol{V}_t$, or they are estimated on-line. We now look at a set of assumptions which reduces the Kalman filter to a 'Dynamic Linear Model'.

11.1.4 Dynamic Linear Models

In this section we consider Dynamic Linear Models (DLMs) [11] which for a univariate time series are

$$\begin{aligned} \boldsymbol{\theta}_t &= \boldsymbol{\theta}_{t-1} + \boldsymbol{w}_t, \qquad \boldsymbol{w}_t \sim N(\boldsymbol{w}_t; 0, \boldsymbol{W}_t) \\ y_t &= \boldsymbol{F}_t\theta_t + \boldsymbol{v}_t, \qquad \boldsymbol{v}_t \sim N(\boldsymbol{v}_t; 0, \sigma_t^2) \end{aligned} \tag{11.23}$$

This is a linear regression model with time-varying coefficients. It is identical to the generic Kalman filter model with $\boldsymbol{G}_t = \boldsymbol{I}$. Substituting this into the update equations gives

$$\begin{aligned} \hat{\boldsymbol{\theta}}_t &= \hat{\boldsymbol{\theta}}_{t-1} + \boldsymbol{K}_t e_t \\ \boldsymbol{\Sigma}_t &= \boldsymbol{R}_t - \boldsymbol{K}_t\boldsymbol{F}_t\boldsymbol{R}_t \end{aligned} \tag{11.24}$$

where

$$\begin{aligned}
\boldsymbol{K}_t &= \frac{\boldsymbol{R}_t \boldsymbol{F}_t^T}{\sigma_{\hat{y}_t}^2} \qquad (11.25)\\
\boldsymbol{R}_t &= \boldsymbol{\Sigma}_{t-1} + \boldsymbol{W}_t \\
\sigma_{\hat{y}_t}^2 &= \sigma_t^2 + \sigma_\theta^2 \\
\sigma_\theta^2 &= \boldsymbol{F}_t \boldsymbol{R}_t \boldsymbol{F}_t^T \\
e_t &= y_t - \hat{y}_t \\
\hat{y}_t &= \boldsymbol{F}_t \hat{\boldsymbol{\theta}}_{t-1}
\end{aligned}$$

(11.26)

where $\hat{y}_t$ is the prediction and $\sigma_{\hat{y}_t}^2$ is the estimated prediction variance. This is composed of two terms; the observation noise, σ_t^2, and the component of prediction variance due to state uncertainty, σ_θ^2. The likelihood of a data point under the old model (or evidence) is

$$p(y_t) = N\left(y_t; \hat{y}_t, \sigma_{\hat{y}_t}^2\right) \qquad (11.27)$$

If we make the further assumption that the transformation vector (its no longer a matrix because we have univariate predictions) is equal to $\boldsymbol{F}_t = -[y_{t-1}, y_{t-2}, ..., y_{t-p}]$ then we have a Dynamic Autoregressive (DAR) model.

To apply the model we make initial guesses for the state (AR parameters) mean and covariance ($\hat{\boldsymbol{\theta}}_0$ and $\boldsymbol{\Sigma}_0$) and use the above equations. We must also plug in guesses for the state noise covariance, $\boldsymbol{W}_t$, and the observation noise variance, σ_t^2. In a later section we show how these can be estimated on-line. It is also often assumed that the state noise covariance matrix is the isotropic matrix, $\boldsymbol{W}_t = q\boldsymbol{I}$. Next, we look at a set of assumptions that reduce the Kalman filter to Recursive Least Squares.

11.1.5 Recursive least squares

If there is no state noise ($\boldsymbol{w}_t = 0$,$\boldsymbol{W}_t = 0$) and no state flow ($\boldsymbol{G}_t = \boldsymbol{I}$) then the linear dynamical system in equation (1) reduces to a static linear system ($\boldsymbol{\theta}_t = \boldsymbol{\theta}$). If we further assume that our observations are univariate we can re-write the state-space equations as

$$y_t = \boldsymbol{F}_t \theta + \boldsymbol{v}_t, \qquad \boldsymbol{v}_t \sim N(\boldsymbol{v}_t; 0, \sigma_t^2) \qquad (11.28)$$

This is a regression model with constant coefficients. We can, however, estimate these coefficients in a recursive manner by substituting our assumptions about $\boldsymbol{W}_t$, $\boldsymbol{G}_t$ and $\boldsymbol{V}_t$ into the Kalman filter update equations. This gives

$$\hat{\boldsymbol{\theta}}_t = \hat{\boldsymbol{\theta}}_{t-1} + \boldsymbol{K}_t e_t \tag{11.29}$$

$$\boldsymbol{\Sigma}_t = \boldsymbol{\Sigma}_{t-1} - \boldsymbol{K}_t \boldsymbol{F}_t \boldsymbol{\Sigma}_{t-1} \tag{11.30}$$

where

$$\boldsymbol{K}_t = \frac{\boldsymbol{\Sigma}_{t-1}\boldsymbol{F}_t^T}{\sigma^2_{\hat{y}_t}} \tag{11.31}$$

$$\begin{aligned} \sigma^2_{\hat{y}_t} &= \sigma^2_t + \sigma^2_\theta \\ \sigma^2_\theta &= \boldsymbol{F}_t \boldsymbol{\Sigma}_{t-1} \boldsymbol{F}_t^T \\ e_t &= y_t - \hat{y}_t \\ \hat{y}_t &= \boldsymbol{F}_t \hat{\boldsymbol{\theta}}_{t-1} \end{aligned} \tag{11.32}$$

where $\hat{y}_t$ is the prediction and $\sigma^2_{\hat{y}_t}$ is the estimated prediction variance. This is composed of two terms; the observation noise, σ^2_t, and the component of prediction variance due to state uncertainty, σ^2_θ.

The above equations are identical to the update equations for recursive least squares (RLS) as defined by Abraham and Ledolter (equation (8.60) in [1]).

The likelihood of a data point under the old model (or evidence) is

$$p(y_t) = N\left(y_t; \hat{y}_t, \sigma^2_{\hat{y}_t}\right) \tag{11.33}$$

If we make the further assumption that the transformation vector (its no longer a matrix because we have univariate predictions) is equal to $\boldsymbol{F}_t = -[y_{t-1}, y_{t-2}, ..., y_{t-p}]$ then we have a recursive least squares estimation procedure for an autoregressive (AR) model.

To apply the model we make initial guesses for the state (AR parameters) mean and covariance ($\hat{\boldsymbol{\theta}}_0$ and $\boldsymbol{\Sigma}_0$) and use the above equations. We must also plug in our guess for the observation noise variance, σ^2_t. In a later section we show how this can be estimated on-line.

11.1.6 Estimation of noise parameters

To use the DLM update equations it is necessary to make guesses for the state noise covariance, $\boldsymbol{W}_t$, and the observation noise variance, σ^2_t. In this section we show how these can be estimated on-line. Note, we either estimate the state noise or the observation noise - not both.

Jazwinski's method for estimating state noise

This method, reviewed in [14] is ultimately due to Jazwinski [28] who derives the following equations using the MLII approach (see Bayes lecture). We assume that the state noise covariance matrix is the isotropic matrix, $\boldsymbol{W} = q\boldsymbol{I}$. The parameter q can be updated according to

$$q = h\left(\frac{e^2 - \sigma_{q0}^2}{\boldsymbol{F}_t \boldsymbol{F}_t^T}\right) \tag{11.34}$$

where $h(x)$ is the 'ramp' function

$$h(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{11.35}$$

and σ_{q0}^2 is the estimated prediction variance assuming that $q = 0$

$$\sigma_{q0}^2 = \sigma_t^2 + \boldsymbol{F}_t \boldsymbol{\Sigma}_{t-1} \boldsymbol{F}_t^T \tag{11.36}$$

Thus, if our estimate of prediction error assuming no state noise is smaller than our observed error (e^2) we should infer that the state noise is non-zero. This will happen when we transit from one stationary regime to another; our estimate of q will increase. This, in turn, will increase the learning rate (see later section). A smoothed estimate is

$$q_t = \alpha q_{t-1} + (1-\alpha) h\left(\frac{e^2 - \sigma_{q0}^2}{\boldsymbol{F}_t \boldsymbol{F}_t^T}\right) \tag{11.37}$$

where α is a smoothing parameter. Alternatively, equation 11.34 can be applied to a window of samples [14].

Jazwinski's method for estimating observation noise

This method, reviewed in [14] is ultimately due to Jazwinski [28] who derives the following equations by applying the MLII framework (see Bayes lecture). Equation 11.26 shows that the estimated prediction variance is composed of two components; the observation noise and the component due to state uncertainty. Thus, to estimate the observation noise one needs to subtract the second component from the measured squared error

$$\sigma_t^2 = h\left(e_t^2 - \boldsymbol{F}_t \boldsymbol{R}_{t-1} \boldsymbol{F}_t^T\right) \tag{11.38}$$

This estimate can be derived by setting σ_t^2 so as to maximise the evidence (likelihood) of a new data point (equation 11.27). A smoothed estimate is

$$\sigma_t^2 = \alpha\sigma_{t-1}^2 + (1-\alpha)h\left(e_t^2 - \boldsymbol{F}_t\boldsymbol{R}_{t-1}\boldsymbol{F}_t^T\right) \tag{11.39}$$

where α is a smoothing parameter. Alternatively, equation 11.38 can be applied to a window of samples [14].

For RLS these update equations can be used by substituting $\boldsymbol{R}_t = \Sigma_{t-1}$. We stress, however, that this estimate is especially unsuitable for RLS applied to non-stationarity data (but then you should only use RLS for stationary data, anyway). This is because the learning rate becomes dramatically decreased.

We also stress that Jazwinski's methods cannot both be applied at the same time; the 'extra' prediction error is explained *either* as greater observation noise *or* as greater state noise.

Skagens' method

Skagen [57] lets $W = \rho\sigma_t^2\boldsymbol{I}$ ie. assumes the state noise covariance is isotropic with a variance that is proportional to the observation noise σ_t^2.

He observes that if ρ is kept fixed then varying σ_t^2 over six orders of magnitude has little or no effect on the Kalman filter updates. He therefore sets σ_t^2 to an arbitrary value eg. 1.

He then defines a measure R as the relative reduction in prediction error due to adaption and chooses ρ to give a value of $R = 0.5$.

11.1.7 Comparison with steepest descent

For a linear predictor, the learning rule for 'on-line' steepest descent is [3]

$$\hat{\boldsymbol{\theta}}_t = \hat{\boldsymbol{\theta}}_{t-1} + \alpha\boldsymbol{F}_t^T e_t \tag{11.40}$$

where α is the learning rate, which is fixed and chosen arbitrarily beforehand. This method is otherwise known as Least Mean Squares (LMS). Haykin [27] (page 362) discusses the conditions on α which lead to a convergent learning process. Comparison of the above rule with the DLM learning rule in equation 11.25 shows that DLM has a learning rate *matrix* equal to

$$\boldsymbol{\alpha} = \frac{\boldsymbol{\Sigma}_{t-1} + q_t\boldsymbol{I}}{\sigma_t^2 + \sigma_\theta^2} \tag{11.41}$$

The average learning rate, averaged over all state variables, is given by

$$\alpha_{DLM} = \frac{1}{p} \frac{Tr\left(\mathbf{\Sigma}_{t-1} + q_t \boldsymbol{I}\right)}{\left(\sigma_t^2 + \sigma_\theta^2\right)} \tag{11.42}$$

where $Tr()$ denotes the trace of the covariance matrix and p is the number of state variables.

DLM thus uses a learning rate which is directly proportional to the variance of the state variables and is inversely proportional to the estimated prediction variance.

If the prediction variance due to state uncertainty is significantly smaller than the prediction variance due to state noise ($\sigma_\theta^2 \ll \sigma_t^2$), as it will be once the filter has reached a steady solution, then increasing the state noise parameter, q_t, will increase the learning rate. This is the mechanism by which DLM increases its learning rate when a new dynamic regime is encountered.

The average learning rate for the RLS filter is

$$\alpha_{RLS} = \frac{1}{p} \frac{Tr\left(\mathbf{\Sigma}_{t-1}\right)}{\left(\sigma_t^2 + \sigma_\theta^2\right)} \tag{11.43}$$

As there is no state noise ($q_t = 0$) there is no mechanism by which the learning rate can be increased when a new dynamic regime is encountered. This underlines the fact that RLS is a stationary model. In fact, RLS behaves particularly poorly when given non-stationary data. When a new dynamic regime is encountered, σ_θ^2 will increase (and so may σ_t^2 if we're updating it online). This leads not to the desired increase in learning rate, but to a decrease.

For stationary data, however, the RLS model behaves well. As the model encounters more data the parameter covariance matrix decreases which in turn leads to a decrease in learning rate. In on-line gradient descent learning it is desirable to start with a high learning rate (to achieve faster convergence) but end with a low learning rate (to prevent oscillation). RLS exhibits the desirable property of adapting its learning rate in exactly this manner. DLM also exhibits this property when given stationary data, but when given non-stationary data, has the added property of being able to increasing its learning rate when necessary.

We conclude this section by noting that DLM and RLS may be viewed as linear on-line gradient descent estimators with *variable* learning rates; RLS for stationary data and DLM for non-stationary data.

11.1.8 Other algorithms

The Least Mean Squares (LMS) algorithm [27] (Chapter 9) is identical to the steepest-descent method (as described in this paper) - both methods have constant learning rates.

Our comments on the RLS algorithm are relevant to RLS as defined by Abraham and Ledolter [1]. There are, however, a number of variants of RLS. Haykin [27] (page 564) defines an exponentially weighted RLS algorithm, where past samples are given exponentially less attention than more recent samples. This gives rise to a *limited* tracking ability (see chapter 16 in [27]). The tracking ability can be further improved by adding state noise (Extended RLS-1 [27], page 726) or a non-constant state transition matrix (Extended RLS-2 [27], page 727). The Extended RLS-1 algorithm is therefore similar to the DAR model described in this paper.

11.1.9 An example

This example demonstrates the basic functioning of the dynamic AR model and compares it to RLS.

A time series was generated consisting of a 10Hz sine wave in the first second, a 20Hz sinewave in the second second and a 30Hz sine wave in the third second. All signals contained additive Gaussian noise with standard deviation 0.1. One hundred samples were generated per second.

A DAR model with $p = 8$ AR coefficients was trained on the data. The algorithm was given a fixed value of observation noise ($\sigma_t^2 = 0.2$). The state noise was initially set to zero and was adapted using Jazwinski's algorithm described in equation 11.34, using a smoothing value of $\alpha = 0.1$. The model was initialised using linear regression; the first p data points were regressed onto the $p+1$th data point using an SVD implementation of least squares, resulting in the linear regression weight vector $\boldsymbol{w}_{LR}$. The state at time step $t = p+1$ was initialised to this weight vector; $\boldsymbol{\theta}_{p+1} = \boldsymbol{w}_{LR}$. The initial state covariance matrix was set to the linear regression covariance matrix, $\Sigma_{p+1} = \sigma_t^2 \boldsymbol{F}_{p+1} \boldsymbol{F}_{p+1}^T$. Model parameters before time $p+1$ were set to zero.

An RLS model (with $p = 8$ AR coefficients) was also trained on the data. The algorithm was given a fixed value of observation noise ($\sigma_t^2 = 0.2$). The model was initilised by setting $\boldsymbol{\theta}_{p+1} = \boldsymbol{w}_{LR}$ and $\Sigma_{p+1} = \boldsymbol{I}$ (setting $\Sigma_{p+1} = \sigma_t^2 \boldsymbol{F}_{p+1} \boldsymbol{F}_{p+1}^T$ resulted in an initial learning rate that was'nt sufficiently large for the model to adapt to the data - see later).

Figure 11.2 shows the original time series and the evidence of each point in the time series under the DAR model. Data points occuring at the transitions between different dynamic regimes have low evidence.

Figure 11.3 shows that the state noise parameter, q, increases by an amount necessary for the estimated prediction error to equal the actual prediction error. The state noise is high at transitions between different dynamic regimes. Within each dynamic regime the state noise is zero.

Figure 11.4 shows that the variance of state variables reduces as the model is exposed to more data from the same stationary regime. When a new stationary regime is encountered the state variance increases (because q increases).

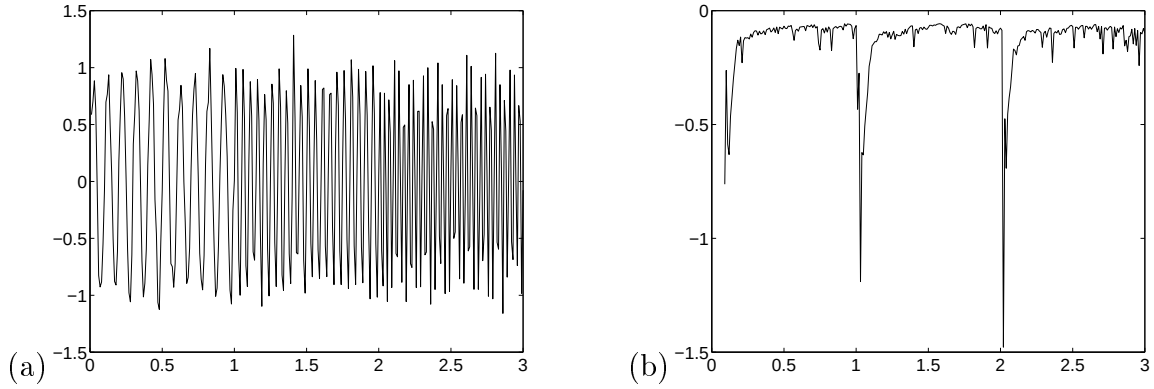


Figure 11.2: (a) Original time series (b) Log evidence of data points under DAR model, $\log p(y_t)$.

Figure 11.5 shows that the learning rate of the DAR model increases when the system enters a new stationary regime, whereas the learning rate of RLS actually decreases. The RLS learning rate is initially higher because the state covariance matrix was initialised differently (initialising it in the same way gave much poorer RLS spectral estimates).

Figure 11.6 shows the spectral estimates obtained from the DAR and RLS models. The learning rate plots and spectrogram plots show that DAR is suitable for non-stationary data whereas RLS is not.

11.1.10 Discussion

Dynamic Linear Models, Recursive Least Squares and Steepest-Descent Learning. are special cases of linear dynamical systems and their learning rules are special cases of the Kalman filter. Steepest-Descent Learning is suitable for modelling stationary data. It uses a learning rate parameter which needs to be high at the beginning of learning (to ensure fast learning) but low at the end of learning (to prevent oscillations). The learning rate parameter is usually hand-tuned to fulfill these criteria. Recursive Least Squares is also suitable for modelling stationary data. It has the advantage of having an adaptive learning rate that reduces gradually as learning proceeds. It reduces in response to a reduction in the uncertainty (covariance) of the model parameters. Dynamic Linear Models are suitable for stationary and non-stationary enviroments. The models possess state-noise and observation noise parameters which can be updated on-line so as to maximise the evidence of the observations.

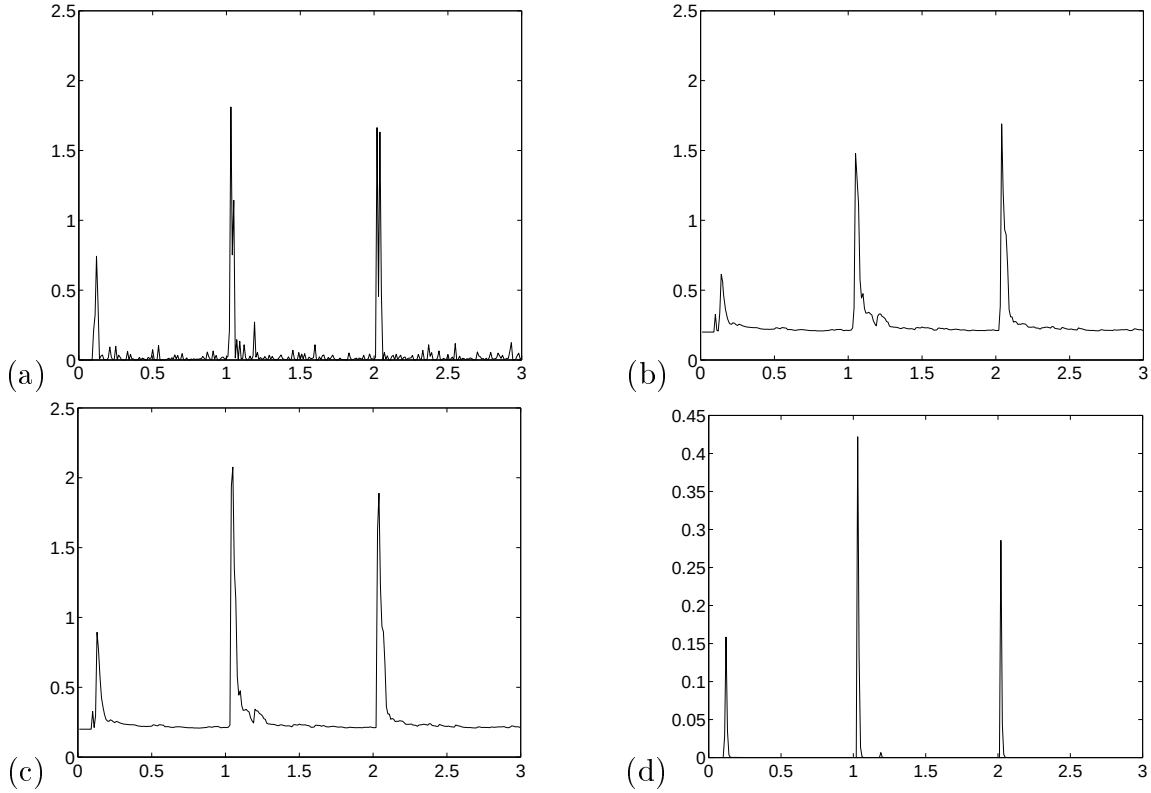


Figure 11.3: (a) Squared prediction error, e_t^2, (b) Estimated prediction error with $q_t = 0$, σ_{q0}^2, (c) Estimated prediction error, $\sigma_{\hat{y}_t}^2$ (the baseline level is due to the fixed observation noise component, $\sigma_t^2 = 0.2$) and (d) Estimate of state noise variance, q_t. The state noise, q_t, increases by an amount necessary for the estimated prediction error (plot c) to equal the actual predicition error (plot a) - see equation 11.34.

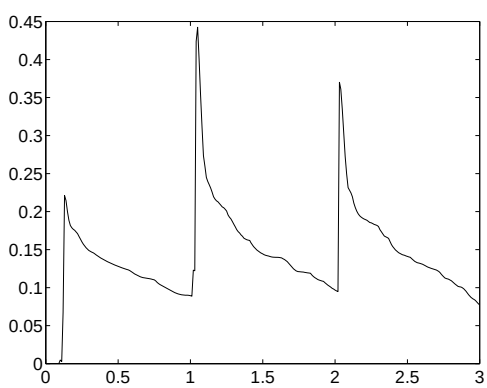


Figure 11.4: Average prior variance of state variables, $\frac{1}{p}Tr(R_t)$. As the model is exposed to more data from the same stationary regime the estimates of the state variables become more accurate (less variance). When a new stationary regime is encountered the state variance increases (because q increases).

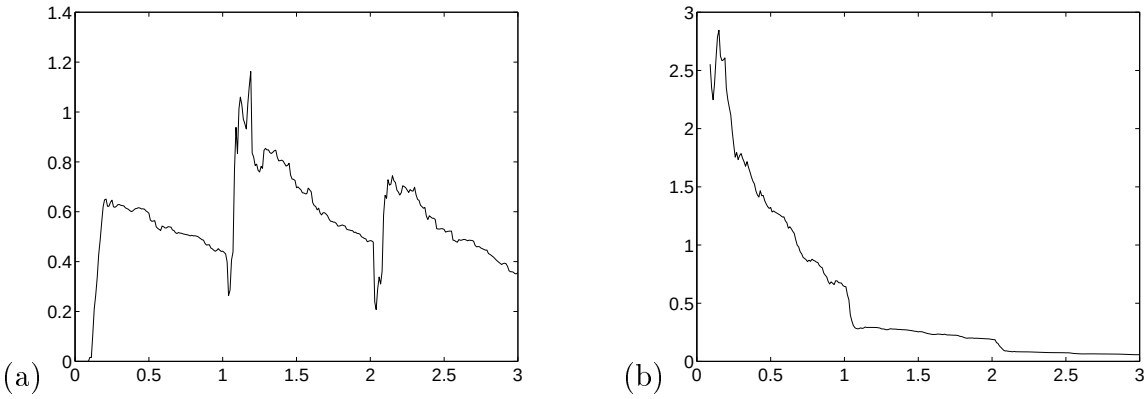


Figure 11.5: Average learning rates for (a) DAR model (b) RLS model. The learning rate for RLS is set to a higher initial value (indirectly by setting $\mathbf{\Sigma}$ to have larger entries) to give it a better chance of tracking the data. The DAR model responds to a new dynamic regime by increasing the learning rate. The RLS responds by decreasing the learning rate and is therefore unable to track the nonstationarity.

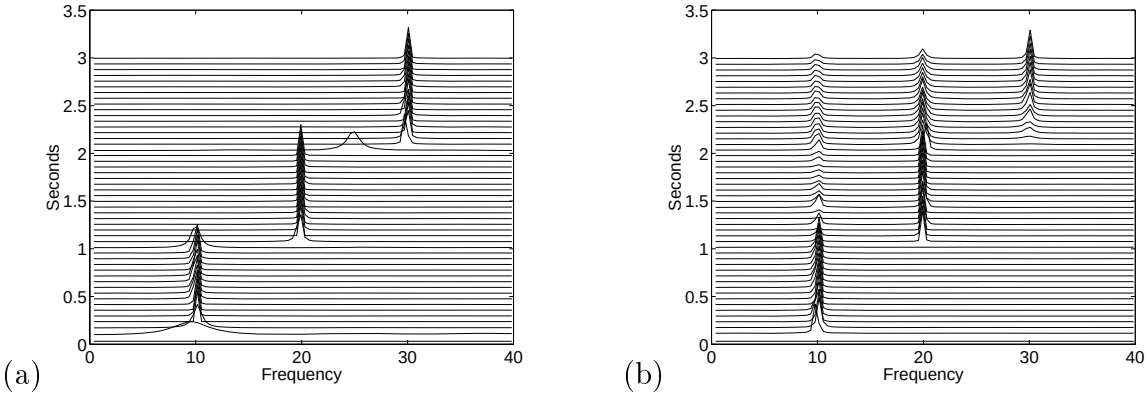


Figure 11.6: Spectrograms for (a) DAR model (b) RLS model.

Chapter 12

EM algorithms

The Expectation-Maximization (EM) algorithm is a maximum likelihood method for models that have hidden variables eg. Gaussian Mixture Models (GMMs), Linear Dynamic Systems (LDSs) and Hidden Markov Models (HMMs).

12.1 Gaussian Mixture Models

Say we have a variable which is multi-modal ie. it separates into distinct clusters. For such data the mean and variance are not very representative quantities.

In a 1-dimensional Gaussian Mixture Model (GMM) with m-components the likelihood of a data point x_n is given by

$$p(x_n) = \sum_{k=1}^{m} p(x_n|k)p(s^n = k) \tag{12.1}$$

where s_n is an indicator variable indicating which component is selected for which data point. These are chosen probabilistically according to

$$p(s^n = k) = \pi_k \tag{12.2}$$

and each component is a Gaussian

$$p(x_n|k) = \frac{1}{(2\pi\sigma_k^2)^{1/2}} \exp\left(-\frac{(x_n - \mu_k)^2}{2\sigma_k^2}\right) \tag{12.3}$$

To generate data from a GMM we pick a Gaussian at random (according to 12.2) and then sample from that Gaussian. To fit a GMM to a data set we need to estimate π_k, μ_k and σ_k^2. This can be achieved in two steps. In the 'E-Step' we soft-partition the data among the different clusters. This amounts to calculating the probability that data point n belongs to cluster k which, from Baye's rule, is

$$\gamma_k^n \equiv p(s_n = k|x_n) = \frac{p(x_n|k)p(s_n = k)}{\sum_{k'} p(x_n|k')p(s_n = k')} \tag{12.4}$$

In the 'M-Step' we re-estimate the parameters using Maximum Likelihood, but the data points are weighted according to the soft-partitioning

$$\begin{aligned} \pi_k &= \sum \gamma_k^n \\ \mu_k &= \frac{\sum \gamma_k^n x_n}{\sum \gamma_k^n} \\ \sigma_k^2 &= \frac{\sum \gamma_k^n (x_n - \mu_k)^2}{\sum \gamma_k^n} \end{aligned} \tag{12.5}$$

These two steps constitute an EM algorithm. Summarizing:

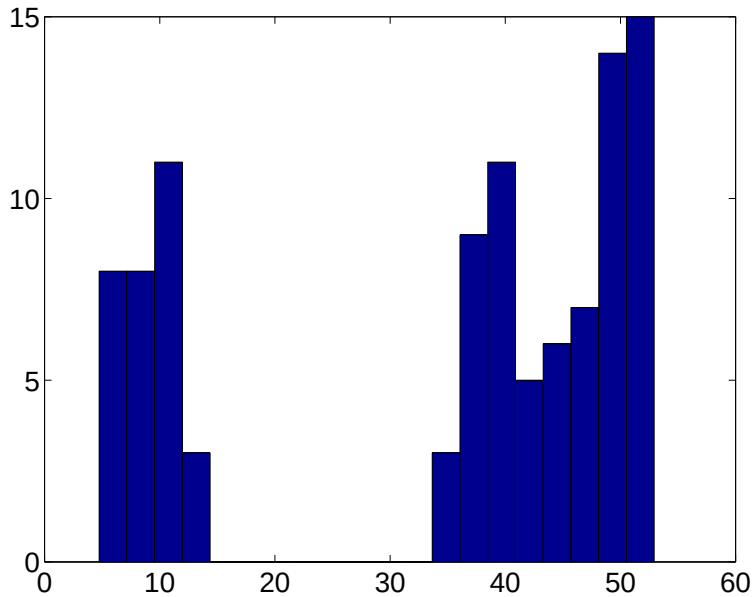


Figure 12.1: A variable with 3 modes. This can be accurately modelled with a 3-component Gaussian Mixture Model.

- E-Step: Soft-partitioning.
- M-Step: Parameter updating.

Application of a 3-component GMM to our example data gives for cluster (i) $\pi_1 = 0.3$, $\mu_1 = 10$, $\sigma_1^2 = 10$, (ii) $\pi_1 = 0.35$, $\mu_1 = 40$, $\sigma_1^2 = 10$, and (iii) $\pi_3 = 0.35$, $\mu_1 = 50$, $\sigma_1^2 = 5$.

GMMs are readily extended to multivariate data by replacing each univariate Gaussian in the mixture with a multivariate Gaussian. See eg. chapter 3 in [3].

12.2 General Approach

If V are visible variables, H are hidden variables and θ are parameters then

1. E-Step: Get $p(H|V,\theta)$
2. M-Step, change θ so as to maximise

$$Q = < \log p(V,H|\theta) > \tag{12.6}$$

where expectation is wrt $p(H|V,\theta)$.

Why does it work ? Maximising Q maximises the likelihood $p(V|\theta)$. This can be proved as follows. Firstly

$$p(V \mid \theta) = \frac{p(H,V \mid \theta)}{p(H \mid V,\theta)} \tag{12.7}$$

This means that the log-likelihood, $L(\theta) \equiv \log p(V \mid \theta)$, can be written

$$L(\theta) = \log p(H,V \mid \theta) - \log p(H \mid V,\theta) \tag{12.8}$$

If we now take expectations with respect to a distribution $p'(H)$ then we get

$$L(\theta) = \int p'(H) \log p(H,V \mid \theta) dH - \int p'(H) \log p(H \mid V,\theta) dH \tag{12.9}$$

The second term is minimised by setting $p'(H) = p(H|V,\theta)$ (we can prove this from Jensen's inequality or the positivity of the KL divergence; see [12] or lecture 4). This takes place in the E-Step. After the E-step the auxiliary function Q is then equal to the log-likelihood. Therefore, when we maximise Q in the M-step we are maximising the likelihood.

12.3 Probabilistic Principal Component Analysis

In an earlier lecture, Principal Component Analysis (PCA) was viewed as a linear transform

$$\boldsymbol{y} = \boldsymbol{Q}^T \boldsymbol{x} \tag{12.10}$$

where the jth column of the matrix $\boldsymbol{Q}$ is the jth eigenvector, $\boldsymbol{q}_j$, of the covariance matrix of the original d-dimensional data $\boldsymbol{x}$. The jth projection

$$y_j = \boldsymbol{q}_j^T \boldsymbol{x} \tag{12.11}$$

has a variance given by the jth eigenvalue λ_j. If the projections are ranked according to variance (ie. eigenvalue) then the M variables that reconstruct the original data with minimum error (and are also linear functions of $\boldsymbol{x}$) are given by $y_1, y_2, ..., y_M$. The remaining variables $y_{M+1}, .., y_d$ can be be discarded with minimal loss of information (in the sense of least squares error). The reconstructed data is given by

$$\hat{\boldsymbol{x}} = \boldsymbol{Q}_{1:M} \boldsymbol{y}_{1:M} \tag{12.12}$$

where $\boldsymbol{Q}_{1:M}$ is a matrix formed from the first M columns of $\boldsymbol{Q}$. Similarly, $\boldsymbol{y}_{1:M} = [y_1, y_2, ..., y_M]^T$.

In probabilistic PCA (pPCA) [60] the PCA transform is converted into a statistical model by explaining the 'discarded' variance as observation noise

$$\boldsymbol{x} = \boldsymbol{W}\boldsymbol{y} + \boldsymbol{e} \tag{12.13}$$

where the noise is drawn from a zero mean Gaussian distribution with isotropic covariance $\sigma^2\boldsymbol{I}$. The 'observations' $\boldsymbol{x}$ are generated by transforming the 'sources' $\boldsymbol{y}$ with the 'mixing matrix' $\boldsymbol{W}$ and then adding 'observation noise'. The pPCA model has M sources where $M < d$. For a given M, we have $\boldsymbol{W} = \boldsymbol{Q}_{1:M}$ and

$$\sigma^2 = \frac{1}{M-d}\sum_{j=M+1}^{d}\lambda_j \tag{12.14}$$

which is the average variance of the discarded projections.

There also exists an EM algorithm for finding the mixing matrix which is more efficient than SVD for high dimensional data. This is because it only needs to invert an M-by-M matrix rather than a d-by-d matrix.

If we define $\boldsymbol{S}$ as the sample covariance matrix and

$$\boldsymbol{C} = \boldsymbol{W}\boldsymbol{W}^T + \sigma^2\boldsymbol{I} \tag{12.15}$$

then the log-likelihood of the data under a pPCA model is given by [60]

$$\log p(\boldsymbol{X}) = -\frac{Nd}{2}\log 2\pi - \frac{N}{2}\log|\boldsymbol{C}| - \frac{N}{2}Tr(\boldsymbol{C}^{-1}\boldsymbol{S}) \tag{12.16}$$

where N is the number of data points.

We are now in the position to apply the MDL model order selection criterion. We have

$$MDL(M) = -\log p(\boldsymbol{X}) + \frac{Md}{2}\log N \tag{12.17}$$

This gives us a procedure for choosing the optimal number of sources.

Because pPCA is a probabilistic model (whereas PCA is a transform) it is readily incorporated in larger models. A useful model, for example, is the Mixtures of pPCA model. This is identical to the Gaussian Mixture model except that each Gaussian is decomposed using pPCA (rather than keeping it as a full covariance Gaussian). This can greatly reduce the number of parameters in the model [61].

12.4 Linear Dynamical Systems

A Linear Dynamical System is given by the following 'state-space' equations

$$\begin{aligned} x_{t+1} &= Ax_t + w_t \\ y_t &= Cx_t + v_t \end{aligned} \tag{12.18}$$

where the state noise and observation noise are zero mean Gaussian variables with covariances Q and R. Given A,C,Q and R the state can be updated using the Kalman filter.

For real-time applications we can infer the states using a Kalman filter. For retrospective/offline data analysis the state at time t can be determined using data before t *and* after t. This is known as Kalman smoothing. See eg. [21].

Moreover, we can also infer other parameters; eg. state noise covariance Q , state transformation matrix C, etc. See eg. [22]. To do this, the state is regarded as a 'hidden variable' (we do not observe it) and we apply the EM algorithm [15].

For an LDS

$$\begin{aligned} x_{t+1} &= Ax_t + w_t \\ y_t &= Cx_t + v_t \end{aligned} \tag{12.19}$$

the hidden variables are the states x_t and the observed variable is the time series y_t. If $x_1^t = [x_1, x_2, .., x_t]$ are the states and $y_1^T = [y_1, y_2, ..., y_t]$ are the observations then the EM algorithm is as follows.

M-Step

In the M-Step we maximise

$$Q =< \log p(y_1^T, x_1^T|\theta) > \tag{12.20}$$

Because of the Markov Property of an LDS (the current state only depends on the last one, and not on ones before that) we have

$$p(y_1^T, x_1^T|\theta) = p(x_1) \prod_{t=2}^{T} p(x_t|x_{t-1}) \prod_{t=1}^{T} p(y_t|x_t) \tag{12.21}$$

and when we we take logs we get

$$\log p(y_1^T, x_1^T|\theta) = \log p(x_1) + \sum_{t=2}^{T} \log p(x_t|x_{t-1}) + \sum_{t=1}^{T} \log p(y_t|x_t) \tag{12.22}$$

where each PDF is a multivariate Gaussian. We now need to take expectations wrt. the distribution over hidden variables

$$\gamma_t \equiv p(x_t|y_1^T) \tag{12.23}$$

This gives

$$< \log p(y_1^T, x_1^T|\theta) >= \gamma_1 \log p(x_1) + \sum_{t=2}^{T} \gamma_t \log p(x_t|x_{t-1}) + \sum_{t=1}^{T} \gamma_t \log p(y_t|x_t) \tag{12.24}$$

By taking derivates wrt each of the parameters and setting them to zero we get update equations for A, Q,C and R. See [22] for details. The distribution over hidden variables is calculated in the E-Step.

E-Step

The E-Step consists of two parts. In the forward-pass the joint probability

$$\alpha_t \equiv p(x_t, y_1^t) = \int \alpha_{t-1} p(x_t|x_{t-1}) p(y_t|x_t) dx_{t-1} \tag{12.25}$$

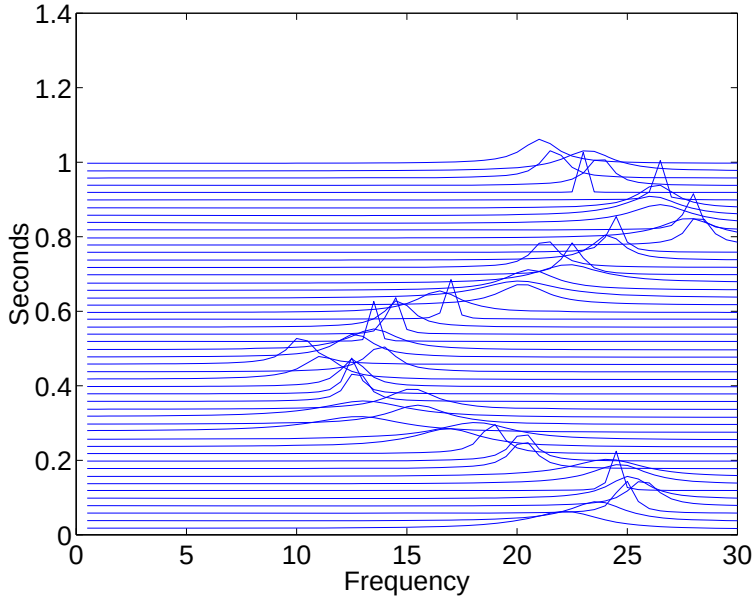
is recursively evaluated using a Kalman filter. In the backward pass we estimate the conditional probability

$$\beta_t \equiv p(y_t^T|x_t) = \int \beta_{t+1} p(x_{t+1}|x_t) p(y_{t+1}|x_{t+1}) dx_{t+1} \tag{12.26}$$

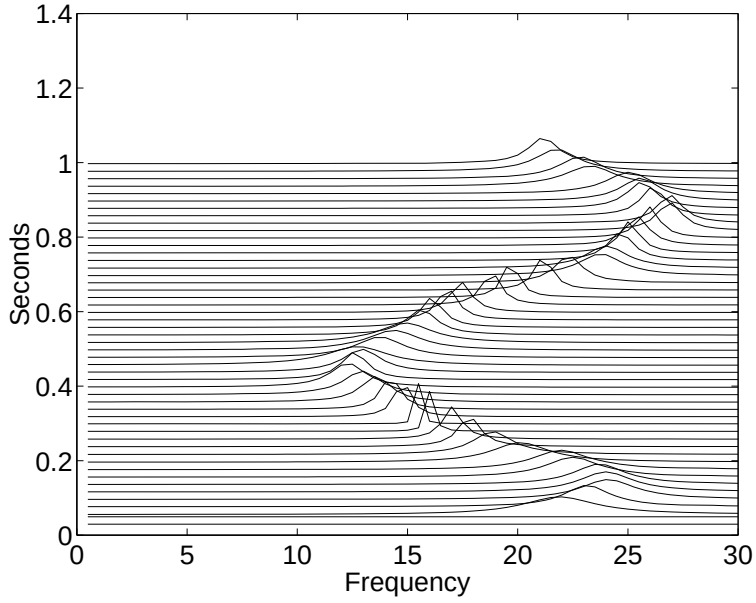
The two are then combined to produce a smoothed estimate

$$p(x_t|y_1^T) \propto \alpha_t \beta_t \tag{12.27}$$

This E-Step constitutes a Kalman smoother.



(a)



(b)

Figure 12.2: (a) Kalman filtering and (b) Kalman smoothing.

Appendix A

Series and Complex Numbers

A.1 Power series

A function of a variable x can often be written in terms of a series of powers of x. For the sin function, for example, we have

$$\sin x = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + ... \tag{A.1}$$

We can find out what the appropriate coefficients are as follows. If we substitite $x = 0$ into the above equation we get $a_0 = 0$ since $sin0 = 0$ and all the other terms disappear. If we now *differentiate* both sides of the equation and substitute $x = 0$ we get $a_1 = 1$ (because $\cos 0 = 1 = a_1$). Differentiating twice and setting $x = 0$ gives $a_2 = 0$. Continuing this process gives

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + ... \tag{A.2}$$

Similarly, the series representations for $cosx$ and e^x can be found as

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + ... \tag{A.3}$$

and

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + ... \tag{A.4}$$

More generally, for a function $f(x)$ we get the general result

$$f(x) = f(0) + xf'(0) + \frac{x^2}{2!}f''(0) + \frac{x^3}{3!}f'''(0) + ... \tag{A.5}$$

where $f'(0)$, $f''(0)$ and $f'''(0)$ are the first, second and third derivatives of $f(x)$ evaluated at $x = 0$. This expansion is called a *Maclaurin series.*

So far, to calculate the coefficients in the series we have differentiated and substituted $x = 0$. If, instead, we substitute $x = a$ we get

$$f(x) = f(a) + (x-a)f'(a) + \frac{(x-a)^2}{2!}f''(a) + \frac{(x-a)^3}{3!}f'''(a) + ... \tag{A.6}$$

which is called a *Taylor series.*

For a d-dimensional vector of parameters $\boldsymbol{x}$ the equivalent Taylor series is

$$f(\boldsymbol{x}) = f(\boldsymbol{a}) + (\boldsymbol{x} - \boldsymbol{a})^T \boldsymbol{g} + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{a})^T \boldsymbol{H}(\boldsymbol{x} - \boldsymbol{a}) + ... \tag{A.7}$$

where

$$\boldsymbol{g} = [\partial f/\partial a_1, \partial f/\partial a_2, ..., \partial f/\partial a_d]^T \tag{A.8}$$

is the gradient vector and

$$\boldsymbol{H} = \begin{bmatrix} \frac{\partial f^2}{\partial a_1^2} & \frac{\partial f^2}{\partial a_1 \partial a_2} & .. & \frac{\partial f^2}{\partial a_1 \partial a_d} \\ \frac{\partial f^2}{\partial a_2 \partial a_1} & \frac{\partial f^2}{\partial a_2^2} & .. & \frac{\partial f^2}{\partial a_2 \partial a_d} \\ .. & .. & .. & .. \\ \frac{\partial f^2}{\partial a_d \partial a_1} & \frac{\partial f^2}{\partial a_d \partial a_2} & .. & \frac{\partial f^2}{\partial a_d^2} \end{bmatrix} \tag{A.9}$$

is the Hessian.

A.2 Complex numbers

Very often, when we try to find the roots of an equation [1], we may end up with our solution being the square root of a negative number. For example, the quadratic equation

$$ax^2 + bx + c = 0 \tag{A.10}$$

has solutions which may be found as follows. If we divide by a and *complete the square* [2] we get

$$\left(x + \frac{b}{2a}\right)^2 - \frac{b^2}{4a^2} = \frac{-c}{a} \tag{A.11}$$

Re-arranging gives the general solution

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{A.12}$$

Now, if $b^2 - 4ac < 0$ we are in trouble. What is the square root of a negative number ? To handle this problem, mathematicians have defined the number

$$i = \sqrt{-1} \tag{A.13}$$

allowing all square roots of negative numbers to be defined in terms of i, eg $\sqrt{-9} = \sqrt{9}\sqrt{-1} = 3i$. These numbers are called *imaginary numbers* to differentiate them from *real numbers.*

[1] We may wish to do this in a signal processing context in, for example, an autoregressive model, where, given a set of AR coefficients we wish to see what signals (ie. x) correspond to the AR model. See later in this chapter.

[2] This means re-arranging a term of the form $x^2 + kx$ into the form $(x + \frac{k}{2})^2 - \left(\frac{k}{2}\right)^2$ which is often convenient because x appears only once.

Finding the roots of equations, eg. the quadratic equation above, requires us to combine imaginary numbers and real numbers. These combinations are called *complex numbers*. For example, the equation

$$x^2 - 2x + 2 = 0 \tag{A.14}$$

has the solutions $x = 1 + i$ and $x = 1 - i$ which are complex numbers.

A complex number $z = a + bi$ has two components; a real part and an imaginary part which may be written

$$\begin{aligned} a &= Re\{z\} \\ b &= Im\{z\} \end{aligned} \tag{A.15}$$

The *absolute value* of a complex number is

$$R = Abs\{z\} = \sqrt{a^2 + b^2} \tag{A.16}$$

and the *argument* is

$$\theta = Arg\{z\} = \tan^{-1}\left(\frac{b}{a}\right) \tag{A.17}$$

The two numbers $z = a + bi$ and $z^* = a - bi$ are known as *complex conjugates*; one is the complex conjugate of the other. When multiplied together they form a real number. The roots of equations often come in complex conjugate pairs.

A.3 Complex exponentials

If we take the exponential function of an imaginary number and write it out as a series expansion, we get

$$e^{i\theta} = 1 + \frac{i\theta}{1!} + \frac{i^2\theta^2}{2!} + \frac{i^3\theta^3}{3!} + ... \tag{A.18}$$

By noting that $i^2 = -1$ and $i^3 = i^2 i = -i$ and similarly for higher powers of i we get

$$e^{i\theta} = \left[1 - \frac{\theta^2}{2!} + ...\right] + i\left[\frac{\theta}{1!} - \frac{\theta^3}{3!} + ...\right] \tag{A.19}$$

Comparing to the earlier expansions of $\cos\theta$ and $\sin\theta$ we can see that

$$e^{i\theta} = \cos\theta + i\sin\theta \tag{A.20}$$

which is known as *Euler's formula*. Similar expansions for $e^{-i\theta}$ give the identity

$$e^{-i\theta} = \cos\theta - i\sin\theta \tag{A.21}$$

We can now express the sine and cosine functions in terms of complex exponentials

$$\begin{aligned} \cos\theta &= \frac{e^{i\theta} + e^{-i\theta}}{2} \\ \sin\theta &= \frac{e^{i\theta} - e^{-i\theta}}{2i} \end{aligned} \tag{A.22}$$

A.4 DeMoivre's Theorem

By using the fact that

$$e^{i\theta}e^{i\theta} = e^{i\theta+i\theta} \tag{A.23}$$

(a property of the exponential function and exponents in general eg. $5^3 5^3 = 5^6$) or more generally

$$(e^{i\theta})^k = e^{ik\theta} \tag{A.24}$$

we can write

$$(cos\theta + i\sin\theta)^k = cosk\theta + isink\theta \tag{A.25}$$

which is known as *DeMoivre's theorem.*

A.5 Argand Diagrams

Any complex number can be represented as a complex exponential

$$a + bi = Re^{i\theta} = R(cos\theta + i\sin\theta) \tag{A.26}$$

and drawn on an *Argand diagram.*

Multiplication of complex numbers is equivalent to rotation in the complex plane (due to DeMoivre's Theorem).

$$(a + bi)^2 = R^2 e^{i2\theta} = R^2(cos2\theta + i\sin 2\theta) \tag{A.27}$$

Appendix B

Linear Regression

B.1 Univariate Linear Regression

We can find the slope a and offset b by minising the cost function

$$E = \sum_{i=1}^{N} (y_i - ax_i - b)^2 \tag{B.1}$$

Differentiating with respect to a gives

$$\frac{\partial E}{\partial a} = -2 \sum_{i=1}^{N} x_i (y_i - ax_i - b) \tag{B.2}$$

Differentiating with respect to b gives

$$\frac{\partial E}{\partial b} = -2 \sum_{i=1}^{N} (y_i - ax_i - b) \tag{B.3}$$

By setting the above derivatives to zero we obtain the *normal equations* of the regression. Re-arranging the normal equations gives

$$a \sum_{i=1}^{N} x_i^2 + b \sum_{i=1}^{N} x_i = \sum_{i=1}^{N} x_i y_i \tag{B.4}$$

and

$$a \sum_{i=1}^{N} x_i + bN = \sum_{i=1}^{N} y_i \tag{B.5}$$

By substituting the mean observed values μ_x and μ_y into the last equation we get

$$b = \mu_y - a\mu_x \tag{B.6}$$

Now let

$$S_{xx} = \sum_{i=1}^{N} (x_i - \mu_x)^2 \tag{B.7}$$

$$= \sum_{i=1}^{N} x_i^2 - N\mu_x^2 \tag{B.8}$$

and

$$S_{xy} = \sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y) \tag{B.9}$$

$$= \sum_{i=1}^{N} x_i y_i - N\mu_x\mu_y \tag{B.10}$$

Substiting for b into the first normal equation gives

$$a\sum_{i=1}^{N} x_i^2 + (\mu_y - a\mu_x)\sum_{i=1}^{N} x_i = \sum_{i=1}^{N} x_i y_i \tag{B.11}$$

Re-arranging gives

$$\begin{aligned} a &= \frac{\sum_{i=1}^{N} x_i y_i - \mu_y \sum_{i=1}^{N} x_i}{\sum_{i=1}^{N} x_i^2 + \mu_x \sum_{i=1}^{N} x_i} \\ &= \frac{\sum_{i=1}^{N} x_i y_i - N\mu_x\mu_y}{\sum_{i=1}^{N} x_i^2 + N\mu_x^2} \\ &= \frac{\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y)}{\sum_{i=1}^{N}(x_i - \mu_x)^2} \\ &= \frac{\sigma_{xy}}{\sigma_x^2} \end{aligned} \tag{B.12}$$

B.1.1 Variance of slope

The data points may be written as

$$\begin{aligned} y_i &= \hat{y}_i + e_i \\ &= ax_i + b + e_i \end{aligned} \tag{B.13}$$

where the noise, e_i has mean zero and variance σ_e^2. The mean and variance of each data point are

$$E(y_i) = ax_i + b \tag{B.14}$$

and

$$Var(y_i) = Var(e_i) = \sigma_e^2 \tag{B.15}$$

We now calculate the variance of the estimate a. From earlier we see that

$$a = \frac{\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y)}{\sum_{i=1}^{N}(x_i - \mu_x)^2} \tag{B.16}$$

Let

$$c_i = \frac{(x_i - \mu_x)}{\sum_{i=1}^{N}(x_i - \mu_x)^2} \tag{B.17}$$

We also note that $\sum_{i=1}^{N} c_i = 0$ and $\sum_{i=1}^{N} c_i x_i = 1$. Hence,

$$a = \sum_{i=1}^{N} c_i(y_i - \mu_y) \tag{B.18}$$

$$= \sum_{i=1}^{N} c_i y_i - \mu_y \sum_{i=1}^{N} c_i \tag{B.19}$$

The mean estimate is therefore

$$E(a) = \sum_{i=1}^{N} c_i E(y_i) - \mu_y \sum_{i=1}^{N} c_i \tag{B.20}$$

$$= a \sum_{i=1}^{N} c_i x_i + b \sum_{i=1}^{N} c_i - \mu_y \sum_{i=1}^{N} c_i$$

$$= a \tag{B.21}$$

The variance is

$$Var(a) = Var(\sum_{i=1}^{N} c_i y_i - \mu_y \sum_{i=1}^{N} c_i) \tag{B.22}$$

The second term contains two fixed quantities so acts like a constant. From the later Appendix on Probability Distributions we see that

$$Var(a) = Var(\sum_{i=1}^{N} c_i y_i) \tag{B.23}$$

$$= \sum_{i=1}^{N} c_i^2 Var(y_i)$$

$$= \sigma_e^2 \sum_{i=1}^{N} c_i^2$$

$$= \frac{\sigma_e^2}{\sum_{i=1}^{N}(x_i - \mu_x)^2}$$

$$= \frac{\sigma_e^2}{(N-1)\sigma_x^2}$$

B.2 Multivariate Linear Regression

B.2.1 Estimating the weight covariance matrix

Different instantiations of target noise will generate different estimated weight vectors according to the last equation. The corresponding weight covariance matrix is given by

$$Var(\hat{\boldsymbol{w}}) = Var((\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}) \tag{B.24}$$

Substituting $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{w} + \boldsymbol{e}$ gives

$$Var(\hat{\boldsymbol{w}}) = Var((\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} + (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{e}) \tag{B.25}$$

This is in the form of equation B.28 in Appendix A with $\boldsymbol{d}$ being given by the first term which is constant, $\boldsymbol{C}$ being given by $(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$ and $\boldsymbol{z}$ being given by $\boldsymbol{e}$. Hence,

$$\begin{aligned} Var(\hat{\boldsymbol{w}}) &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T[Var(\boldsymbol{e})][(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T]^T \\ &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\sigma^2\boldsymbol{I})[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T]^T \\ &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\sigma^2\boldsymbol{I})\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1} \end{aligned} \tag{B.26}$$

Re-arranging further gives

$$Var(\hat{\boldsymbol{w}}) = \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1} \tag{B.27}$$

B.3 Functions of random vectors

For a vector of random variables, $\boldsymbol{z}$, and a matrix of constants, $\boldsymbol{C}$, and a vector of constants, $\boldsymbol{d}$, we have

$$Var(\boldsymbol{C}\boldsymbol{z} + \boldsymbol{d}) = \boldsymbol{C}[Var(\boldsymbol{z})]\boldsymbol{C}^T \tag{B.28}$$

where, here, Var() denotes a covariance matrix. This is a generalisation of the result for scalar random variables $Var(cz) = c^2Var(z)$.

The covariance between a pair of random vectors is given by

$$Var(\boldsymbol{C}_1\boldsymbol{z}, \boldsymbol{C}_2\boldsymbol{z}) = \boldsymbol{C}_1[Var(\boldsymbol{z})]\boldsymbol{C}_2^T \tag{B.29}$$

B.3.1 Estimating the weight covariance matrix

Different instantiations of target noise will generate different estimated weight vectors according to the equation 3.7. The corresponding weight covariance matrix is given by

$$\boldsymbol{\Sigma} = Var((\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}) \tag{B.30}$$

Substituting $\boldsymbol{y} = \boldsymbol{X}\hat{\boldsymbol{w}} + \boldsymbol{e}$ gives

$$\boldsymbol{\Sigma} = Var((\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} + (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{e}) \tag{B.31}$$

This is in the form of $Var(\boldsymbol{C}\boldsymbol{z} + \boldsymbol{d})$ (see earlier) with $\boldsymbol{d}$ being given by the first term which is constant, $\boldsymbol{C}$ being given by $(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$ and $\boldsymbol{z}$ being given by $\boldsymbol{e}$. Hence,

$$\begin{aligned} \boldsymbol{\Sigma} &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T[Var(\boldsymbol{e})][(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T]^T \\ &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\sigma_e^2\boldsymbol{I})[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T]^T \\ &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\sigma_e^2\boldsymbol{I})\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1} \end{aligned} \tag{B.32}$$

Re-arranging further gives

$$\boldsymbol{\Sigma} = \sigma_e^2(\boldsymbol{X}^T\boldsymbol{X})^{-1} \tag{B.33}$$

B.3.2 Equivalence of t-test and F-test for feature selection

When adding a new variable x_p to a regression model we can test to see if the increase in the proportion of variance explained is *significant* by computing

$$F = \frac{(N-1)\sigma_y^2\left[r^2(y,\hat{y}_p) - r^2(y,\hat{y}_{p-1})\right]}{\sigma_e^2(p)} \tag{B.34}$$

where $r^2(y,\hat{y}_p)$ is the square of the correlation between y and the regression model with all p variables (ie. including x_p) and $r^2(y,\hat{y}_{p-1})$ is the square of the correlation between y and the regression model without x_p. The denominator is the noise variance from the model including x_p. This statistic is distributed according to the F-distribution with $v_1 = 1$ and $v_2 = N - p - 2$ degrees of freedom.

This test is identical to the double sided t-test on the t-statistic computed from the regression coefficient a_p, described in this lecture (see also page 128 of [32]). This test is also equivalent to seeing if the partial correlation between x_p and y is significantly non-zero (see page 149 of [32]).

Appendix C

Matrix Identities

C.1 Multiplication

Matrix multiplication is associative

$$(\boldsymbol{AB})\boldsymbol{C} = \boldsymbol{A}(\boldsymbol{BC}) \tag{C.1}$$

distributive

$$\boldsymbol{A}(\boldsymbol{B} + \boldsymbol{C}) = \boldsymbol{AB} + \boldsymbol{AC} \tag{C.2}$$

but not commutative

$$\boldsymbol{AB} \neq \boldsymbol{BA} \tag{C.3}$$

C.2 Transposes

Given two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ we have

$$(\boldsymbol{AB})^T = \boldsymbol{B}^T\boldsymbol{A}^T \tag{C.4}$$

C.3 Inverses

Given two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ we have

$$(\boldsymbol{AB})^{-1} = \boldsymbol{B}^{-1}\boldsymbol{A}^{-1} \tag{C.5}$$

The Matrix Inversion Lemma is

$$(\boldsymbol{XBX}^T + \boldsymbol{A})^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{X}(\boldsymbol{B}^{-1} + \boldsymbol{X}^T\boldsymbol{A}^{-1}\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{A}^{-1} \tag{C.6}$$

The Sherman-Morrison-Woodury formula or Woodbury's identity is

$$(\boldsymbol{UV}^T + \boldsymbol{A})^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{U}(\boldsymbol{I} + \boldsymbol{V}^T\boldsymbol{A}^{-1}\boldsymbol{U})^{-1}\boldsymbol{V}^T\boldsymbol{A}^{-1} \tag{C.7}$$

C.4 Eigendecomposition

$$\boldsymbol{Q}^T \boldsymbol{A} \boldsymbol{Q} = \boldsymbol{\Lambda} \tag{C.8}$$

Pre-multiplying by $\boldsymbol{Q}$ and post-multiplying by $\boldsymbol{Q}^T$ gives

$$\boldsymbol{A} = \boldsymbol{Q} \boldsymbol{\Lambda} \boldsymbol{Q}^T \tag{C.9}$$

which is known as the *spectral theorem*. Any real, symmetric matrix can be represented as above where the columns of $\boldsymbol{Q}$ contain the eigenvectors and $\boldsymbol{\Lambda}$ is a diagonal matrix containing the eigenvalues, λ_i. Equivalently,

$$\boldsymbol{A} = \sum_{k=1}^{d} \lambda_k \boldsymbol{q}_k \boldsymbol{q}_k^T \tag{C.10}$$

C.5 Determinants

If $\det(\boldsymbol{A}) = 0$ the matrix $\boldsymbol{A}$ is not invertible; it is *singular*. Conversely, if $\det(\boldsymbol{A}) \neq 0$ then $\boldsymbol{A}$ *is* invertible. Other properties of the determinant are

$$\begin{aligned} \det(\boldsymbol{A}^T) &= \det(\boldsymbol{A}) \\ \det(\boldsymbol{A}\boldsymbol{B}) &= \det(\boldsymbol{A}) \det(\boldsymbol{B}) \\ \det(\boldsymbol{A}^{-1}) &= 1/\det(\boldsymbol{A}) \\ \det(\boldsymbol{A}) &= \prod_k a_{kk} \det(\boldsymbol{A}) = \prod_k \lambda_k \end{aligned} \tag{C.11}$$

C.6 Traces

The *Trace* is the sum of the diagonal elements

$$Tr(\boldsymbol{A}) = \sum_k a_{kk} \tag{C.12}$$

and is also equal to the sum of the eigenvalues

$$Tr(\boldsymbol{A}) = \sum_k \lambda_k \tag{C.13}$$

Also

$$Tr(\boldsymbol{A} + \boldsymbol{B}) = Tr(\boldsymbol{A}) + Tr(\boldsymbol{B}) \tag{C.14}$$

C.7 Matrix Calculus

From [37] we know that the derivative of $\boldsymbol{c}^T \boldsymbol{B} \boldsymbol{c}$ with respect to $\boldsymbol{c}$ is $(\boldsymbol{B}^T + \boldsymbol{B})\boldsymbol{c}$.

Appendix D

Probability Distributions

This appendix archives a number of useful results from texts by Papoulis [44], Lee [33] and Cover [12]. Table 16.1 in Cover (page 486) gives entropies of many distributions not listed here.

D.1 Transforming PDFs

Because probabilities are defined as areas under PDFs when we transform a variable

$$y = f(x) \tag{D.1}$$

we transform the PDF by preserving the areas

$$p(y)|dy| = p(x)|dx| \tag{D.2}$$

where the absolute value is taken because the changes in x or y (dx and dy) may be negative and areas must be positive. Hence

$$p(y) = \frac{p(x)}{|\frac{dy}{dx}|} \tag{D.3}$$

where the derivative is evaluated at $x = f^{-1}(y)$. This means that the function $f(x)$ must be one-to-one and invertible.

If the function is many-to-one then it's inverse will have multiple solutions $x_1, x_2, ..., x_n$ and the PDF is transformed at each of these points (Papoulis' Fundamental Theorem [44], page 93)

$$p(y) = \frac{p(x_1)}{|\frac{dy}{dx_1}|} + \frac{p(x_2)}{|\frac{dy}{dx_2}|} + ... + \frac{p(x_n)}{|\frac{dy}{dx_n}|} \tag{D.4}$$

D.1.1 Mean and Variance

For more on the mean and variance of functions of random variables see Weisberg [64].

Expectation is a *linear operator*. That is

$$E[(a_1 x + a_2 x)] = a_1 E[x] + a_2 E[x] \tag{D.5}$$

Therefore, given the function

$$y = ax \tag{D.6}$$

we can calculate the mean and variance of y as functions of the mean and variance of x.

$$\begin{aligned} E[y] &= aE[x] \\ Var(y) &= a^2 Var(x) \end{aligned} \tag{D.7}$$

If y is a function of many *uncorrelated* variables

$$y = \sum_i a_i x_i \tag{D.8}$$

we can use the results

$$E[y] = \sum_i a_i E[x_i] \tag{D.9}$$

$$Var[y] = \sum_i a_i^2 Var[x_i] \tag{D.10}$$

But if the variables are correlated then

$$Var[y] = \sum_i a_i^2 Var[x_i] + 2 \sum_i \sum_j a_i a_j Var(x_i, x_j) \tag{D.11}$$

where $Var(x_i, x_j)$ denotes the covariance of the random variables x_i and x_j.

Standard Error

As an example, the mean

$$m = \frac{1}{N} \sum_i x_i \tag{D.12}$$

of uncorrelated variables x_i has a variance

$$\begin{aligned} \sigma_m^2 \equiv Var(m) &= \sum_i \frac{1}{N} Var(x_i) \\ &= \frac{\sigma_x^2}{N} \end{aligned} \tag{D.13}$$

where we have used the substitution $a_i = 1/N$ in equation D.10. Hence

$$\sigma_m = \frac{\sigma_x}{\sqrt{N}} \tag{D.14}$$

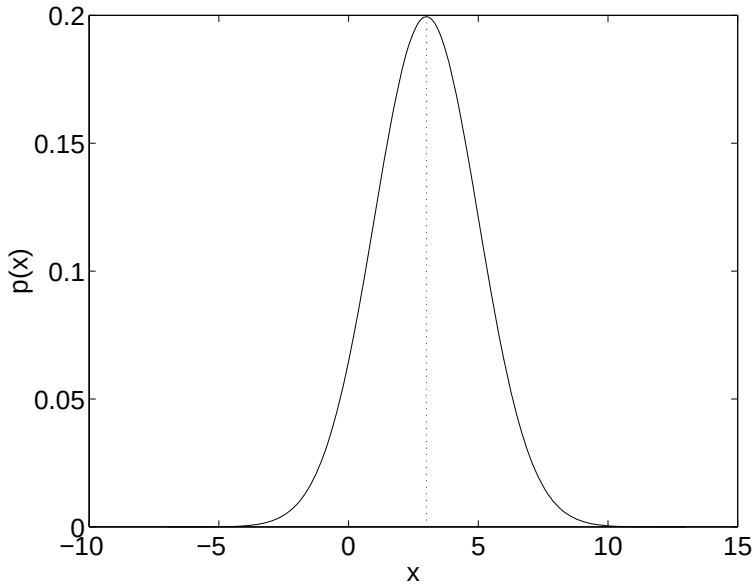


Figure D.1: *The Gaussian Probability Density Function with* $\mu = 3$ *and* $\sigma = 2$.

D.2 Uniform Distribution

The uniform PDF is given by

$$U(x; a, b) = \frac{1}{b - a} \tag{D.15}$$

for $a \leq x \leq b$ and zero otherwise. The mean is $0.5(a + b)$ and variance is $(b - a)^2/12$.

The entropy of a uniform distribution is

$$H(x) = \log(b - a) \tag{D.16}$$

D.3 Gaussian Distribution

The *Normal* or *Gaussian* probability density function, for the case of a single variable, is

$$N(x; \mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right) \tag{D.17}$$

where μ and σ^2 are the mean and variance.

D.3.1 Entropy

The entropy of a Gaussian variable is

$$H(x) = \frac{1}{2} \log \sigma^2 + \frac{1}{2} \log 2\pi + \frac{1}{2} \tag{D.18}$$

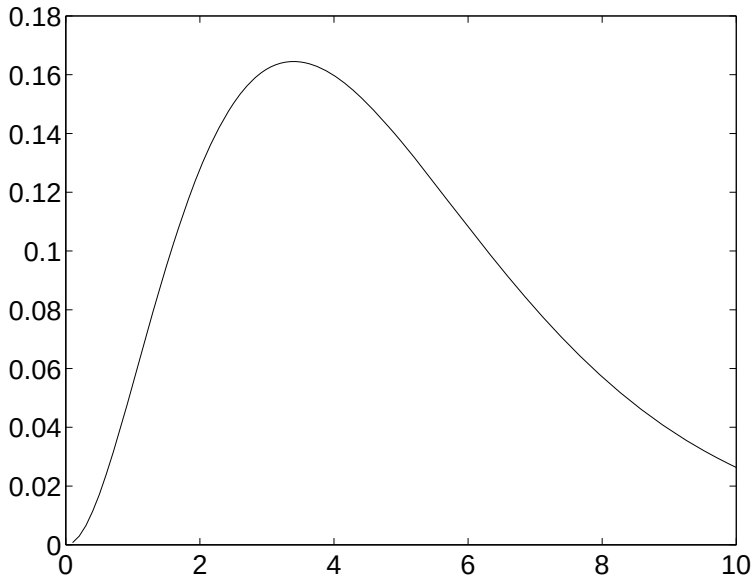


Figure D.2: *The Gamma Density for* $b = 1.6$ *and* $c = 3.125$.

For a given variance, the Gaussian distribution has the highest entropy. For a proof of this see Bishop ([3], page 240).

D.3.2 Relative Entropy

For Normal densities $q(x) = N(x; \mu_q, \sigma_q^2)$ and $p(x) = N(x; \mu_p, \sigma_p^2)$ the KL-divergence is

$$D[q||p] = \frac{1}{2} \log \frac{\sigma_p^2}{\sigma_q^2} + \frac{\mu_q^2 + \mu_p^2 + \sigma_q^2 - 2\mu_q\mu_p}{2\sigma_p^2} - \frac{1}{2} \tag{D.19}$$

D.4 The Gamma distribution

The Gamma density is defined as

$$\Gamma(x; b, c) = \frac{1}{\Gamma(c)} \frac{x^{c-1}}{b^c} \exp\left(\frac{-x}{b}\right) \tag{D.20}$$

where $\Gamma()$ is the *gamma function* [49]. The mean of a Gamma density is given by bc and the variance by b^2c. Logs of gamma densities can be written as

$$\log \Gamma(x; b, c) = \frac{-x}{b} + (c-1) \log x + K \tag{D.21}$$

where K is a quantity which does not depend on x; the log of a gamma density comprises a term in x and a term in $\log x$. The Gamma distribution is only defined for positive variables.

D.4.1 Entropy

Using the result for Gamma densities

$$\int p(x) \log x = \Psi(c) + \log b \tag{D.22}$$

where $\Psi()$ is the digamma function [49] the entropy can be derived as

$$H(x) = log\Gamma(c) + c \log b - (c-1)(\Psi(c) + \log b) + c \tag{D.23}$$

D.4.2 Relative Entropy

For Gamma densities $q(x) = \Gamma(\boldsymbol{x}; b_q, c_q)$ and $p(x) = \Gamma(\boldsymbol{x}; b_p, c_p)$ the KL-divergence is

$$\begin{aligned} D[q||p] &= (c_q - 1)\Psi(c_q) - \log b_q - c_q - \log\Gamma(c_q) \\ &+ \log\Gamma(c_p) + c_p \log b_p - (c_p - 1)(\Psi(c_q) + \log b_q) + \frac{b_q c_q}{b_p} \end{aligned} \tag{D.24}$$

D.5 The χ^2-distribution

If $z_1, z_2, ..., z_N$ are independent normally distributed random variables with zero-mean and unit variance then

$$x = \sum_{i=1}^{N} z_i^2 \tag{D.25}$$

has a χ^2-distribution with N degrees of freedom ([33], page 276). This distribution is a special case of the Gamma distribution with $b = 2$ and $c = N/2$. This gives

$$\chi^2(x; N) = \frac{1}{\Gamma(N/2)} \frac{x^{N/2-1}}{2^{N/2}} \exp\left(\frac{-x}{2}\right) \tag{D.26}$$

The mean and variance are N and $2N$. The entropy and relative entropy can be found by substituting the the values $b = 2$ and $c = N/2$ into equations D.23 and D.24. The χ^2 distribution is only defined for positive variables.

If x is a χ^2 variable with N degrees of freedom and

$$y = \sqrt{x} \tag{D.27}$$

then y has a χ-density with N degrees of freedom. For $N = 3$ we have a *Maxwell* density and for $N = 2$ a *Rayleigh* density ([44], page 96).

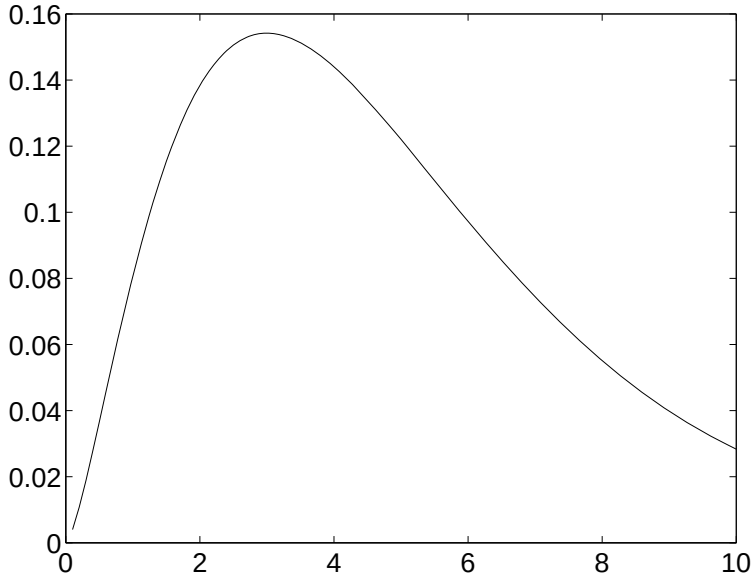


Figure D.3: *The χ^2 Density for $N = 5$ degrees of freedom.*

D.6 The t-distribution

If $z_1, z_2, ..., z_N$ are independent Normally distributed random variables with mean μ and variance σ^2 and m is the sample mean and s is the sample standard deviation then

$$x = \frac{m - \mu}{s/\sqrt{N}} \tag{D.28}$$

has a t-distribution with $N - 1$ degrees of freedom. It is written

$$t(x; D) = \frac{1}{B(D/2, 1/2)} \left(1 + \frac{x^2}{D}\right)^{-(D+1)/2} \tag{D.29}$$

where D is the number of 'degrees of freedom' and

$$B(a, b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a + b)} \tag{D.30}$$

is the *beta function*. For $D = 1$ the t-distribution reduces to the standard Cauchy distribution ([33], page 281).

D.7 Generalised Exponential Densities

The 'exponential power' or 'generalised exponential' probability density is defined as

$$p(a) = G(a; R, \beta) = \frac{R\beta^{1/R}}{2\Gamma(1/R)} \exp(-\beta|a|^R) \tag{D.31}$$

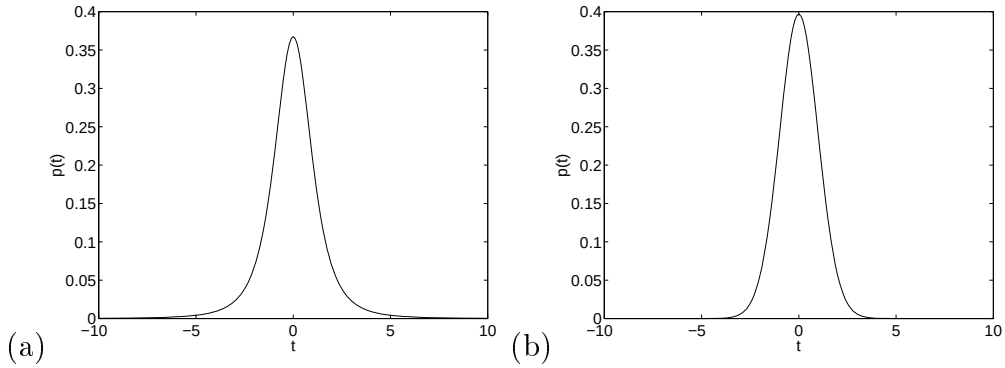


Figure D.4: *The t-distribution with (a) $N = 3$ and (b) $N = 49$ degrees of freedom.*

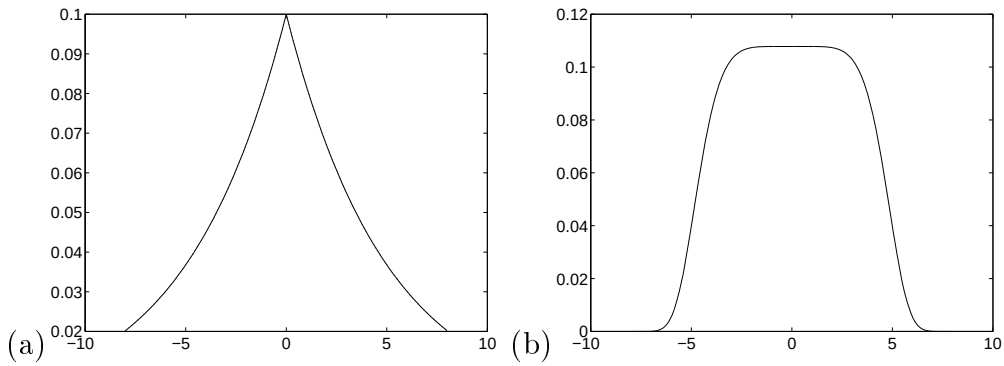


Figure D.5: *The generalised exponential distribution with (a) $R = 1$,$w = 5$ and (b) $R = 6$,$w = 5$. The parameter R fixes the weight of the tails and w fixes the width of the distribution. For (a) we have a Laplacian which has positive kurtosis ($k = 3$); heavy tails. For (b) we have a light-tailed distribution with negative kurtosis ($k = -1$).*

where $\Gamma()$ is the gamma function [49], the mean of the distribution is zero [1], the width of the distribution is determined by $1/\beta$ and the weight of its tails is set by R. This gives rise to a Gaussian distribution for $R = 2$, a Laplacian for $R = 1$ and a uniform distribution in the limit $R \to \infty$. The density is equivalently parameterised by a variable w, which defines the width of the distribution, where $w = \beta^{-1/R}$ giving

$$p(a) = \frac{R}{2w\Gamma(1/R)} \exp(-|a/w|^R) \tag{D.32}$$

The variance is

$$V = w^2 \frac{\Gamma(3/R)}{\Gamma(1/R)} \tag{D.33}$$

which for $R = 2$ gives $V = 0.5w^2$. The kurtosis is given by [7]

$$K = \frac{\Gamma(5/R)\Gamma(1/R)}{\Gamma(3/R)^2} - 3 \tag{D.34}$$

where we have subtracted 3 so that a Gaussian has zero kurtosis. Samples may be generated from the density using a rejection method [59].

[1] For non zero mean we simply replace a with $a - \mu$ where μ is the mean.

D.8 PDFs for Time Series

Given a signal $a = f(t)$ which is sampled uniformly over a time period T, its PDF, $p(a)$ can be calculated as follows. Because the signal is uniformly sampled we have $p(t) = 1/T$. The function $f(t)$ acts to transform this density from one over t to to one over a. Hence, using the method for transforming PDFs, we get

$$p(a) = \frac{p(t)}{|\frac{da}{dt}|} \tag{D.35}$$

where $||$ denotes the absolute value and the derivative is evaluated at $t = f^{-1}(x)$.

D.8.1 Sampling

When we convert an analogue signal into a digital one the sampling process can have a crucial effect on the resulting density. If, for example, we attempt to sample uniformly but the sampling frequency is a multiple of the signal frequency we are, in effect, sampling non-uniformly. For true uniform sampling it is necessary that the ratio of the sampling and signal frequencies be irrational.

D.8.2 Sine Wave

For a sine wave, $a = \sin(t)$, we get

$$p(a) = \frac{1}{|cos(t)|} \tag{D.36}$$

where $cos(t)$ is evaluated at $t = \sin^{-1}(a)$. The inverse sine is only defined for $-\pi/2 \leq t \leq \pi/2$ and $p(t)$ is uniform within this. Hence, $p(t) = 1/\pi$. Therefore

$$p(a) = \frac{1}{\pi\sqrt{1-a^2}} \tag{D.37}$$

This density is *multimodal*, having peaks at $+1$ and -1. For a more general sine wave

$$a = R\sin(wt) \tag{D.38}$$

we get $p(t) = w/\pi$

$$p(a) = \frac{1}{\pi\sqrt{1-(a/R)^2}} \tag{D.39}$$

which has peaks at $\pm R$.

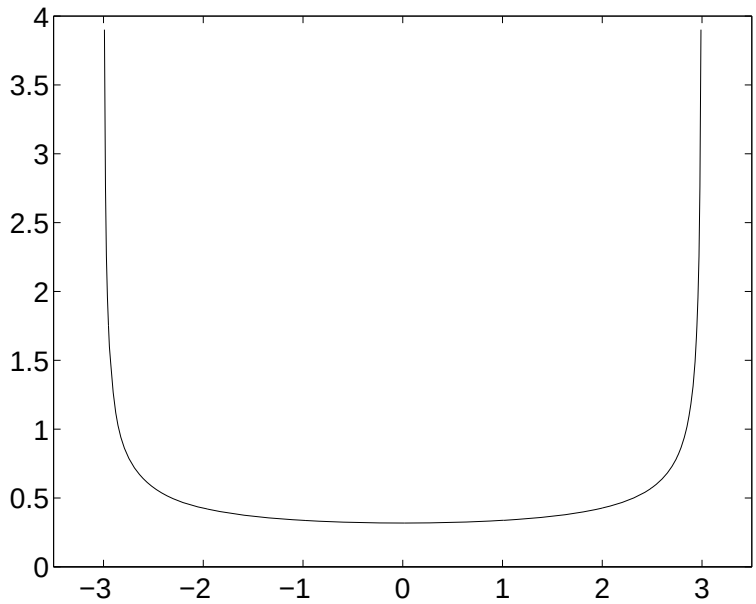


Figure D.6: *The PDF of* $a = R\sin(wt)$ *for* $R = 3$.

Appendix E

Multivariate Probability Distributions

E.1 Transforming PDFs

Just as univariate Probability Density Functions (PDFs) are transformed so as to preserve area so multivariate probability distributions are transformed so as to preserve volume. If

$$\boldsymbol{y} = f(\boldsymbol{x}) \tag{E.1}$$

then this can be achieved from

$$p(\boldsymbol{y}) = \frac{p(\boldsymbol{x})}{abs(|\boldsymbol{J}|)} \tag{E.2}$$

where $abs()$ denotes the absolute value and $||$ the determinant and

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & .. & \frac{\partial y_1}{\partial x_d} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & .. & \frac{\partial y_2}{\partial x_d} \\ .. & .. & .. & .. \\ \frac{\partial y_d}{\partial x_1} & \frac{\partial y_d}{\partial x_2} & .. & \frac{\partial y_d}{\partial x_d} \end{bmatrix} \tag{E.3}$$

is the Jacobian matrix for d-dimensional vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. The partial derivatives are evaluated at $\boldsymbol{x} = f^{-1}(\boldsymbol{y})$. As the determinant of $\boldsymbol{J}$ measures the volume of the transformation, using it as a normalising term therefore preserves the volume under the PDF as desired. See Papoulis [44] for more details.

E.1.1 Mean and Covariance

For a vector of random variables (Gaussian or otherwise), $\boldsymbol{x}$, with mean $\boldsymbol{\mu}_x$ and covariance $\boldsymbol{\Sigma}_x$ a linear transformation

$$\boldsymbol{y} = \boldsymbol{F}\boldsymbol{x} + \boldsymbol{C} \tag{E.4}$$

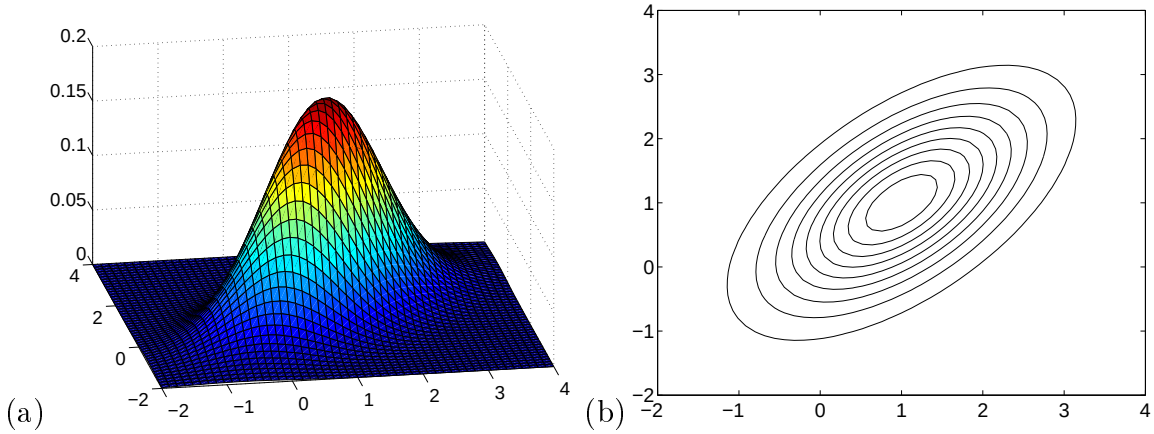


Figure E.1: *(a) 3D-plot and (b) contour plot of Multivariate Gaussian PDF with* $\boldsymbol{\mu} = [1, 1]^T$ *and* $\boldsymbol{\Sigma}_{11} = \boldsymbol{\Sigma}_{22} = 1$ *and* $\boldsymbol{\Sigma}_{12} = \boldsymbol{\Sigma}_{21} = 0.6$ *ie. a positive correlation of* $r = 0.6$.

gives rise to a random vector $\boldsymbol{y}$ with mean

$$\boldsymbol{\mu}_y = \boldsymbol{F}\boldsymbol{\mu}_x + \boldsymbol{C} \tag{E.5}$$

and covariance

$$\boldsymbol{\Sigma}_y = \boldsymbol{F}\boldsymbol{\Sigma}_x\boldsymbol{F}^T \tag{E.6}$$

If we generate another random vector, this time from a *different* linear transformation of $\boldsymbol{x}$

$$\boldsymbol{z} = \boldsymbol{G}\boldsymbol{x} + \boldsymbol{D} \tag{E.7}$$

then the covariance *between* the random vectors $\boldsymbol{y}$ and $\boldsymbol{z}$ is given by

$$\boldsymbol{\Sigma}_{y,z} = \boldsymbol{F}\boldsymbol{\Sigma}_x\boldsymbol{G}^T \tag{E.8}$$

The i,jth entry in this matrix is the covariance between y_i and z_j.

E.2 The Multivariate Gaussian

The multivariate normal PDF for d variables is

$$N(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu})\right) \tag{E.9}$$

where the mean $\boldsymbol{\mu}$ is a d-dimensional vector, $\boldsymbol{\Sigma}$ is a $d \times d$ covariance matrix, and $|\boldsymbol{\Sigma}|$ denotes the determinant of $\boldsymbol{\Sigma}$.

E.2.1 Entropy

The entropy is

$$H(\boldsymbol{x}) = \frac{1}{2}\log|\boldsymbol{\Sigma}| + \frac{d}{2}\log 2\pi + \frac{d}{2} \tag{E.10}$$

E.2.2 Relative Entropy

For Normal densities $q(\boldsymbol{x}) = N(\boldsymbol{x}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$ and $p(\boldsymbol{x}) = N(\boldsymbol{x}; \boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$ the KL-divergence is

$$D[q||p] = 0.5 \log \frac{|\boldsymbol{\Sigma}_p|}{|\boldsymbol{\Sigma}_q|} + 0.5 Trace(\boldsymbol{\Sigma}_p^{-1}\boldsymbol{\Sigma}_q) + 0.5(\boldsymbol{\mu}_q - \boldsymbol{\mu}_p)^T \boldsymbol{\Sigma}_p^{-1}(\boldsymbol{\mu}_q - \boldsymbol{\mu}_p) - \frac{d}{2} \quad \text{(E.11)}$$

where $|\boldsymbol{\Sigma}_p|$ denotes the determinant of the matrix $\boldsymbol{\Sigma}_p$.

E.3 The Multinomial Distribution

If a random variable x can take one of one m discrete values $x_1, x_2, ..x_m$ and

$$p(x = x_s) = \pi_s \quad \text{(E.12)}$$

then x is said to have a multinomial distribution.

E.4 The Dirichlet Distribution

If $\boldsymbol{\pi} = [\pi_1, \pi_2, ...\pi_m]$ are the parameters of a multinomial distribution then

$$q(\boldsymbol{\pi}) = \Gamma(\lambda_{tot}) \prod_{s=1}^{m} \frac{\pi_s^{\lambda_s - 1}}{\Gamma(\lambda_s)} \quad \text{(E.13)}$$

defines a Dirichlet distribution over these parameters where

$$\lambda_{tot} = \sum_s \lambda_s \quad \text{(E.14)}$$

The mean value of π_s is λ_s/λ_{tot}.

E.4.1 Relative Entropy

For Dirichlet densities $q(\boldsymbol{\pi}) = D(\boldsymbol{\pi}; \boldsymbol{\lambda}_q)$ and $p(\boldsymbol{\pi}) = D(\boldsymbol{\pi}; \boldsymbol{\lambda}_p)$ where the number of states is m and $\boldsymbol{\lambda}_q = [\lambda_q(1), \lambda_q(2), .., \lambda_q(m)]$ and $\boldsymbol{\lambda}_p = [\lambda_p(1), \lambda_p(2), .., \lambda_p(m)]$. the KL-divergence is

$$\begin{aligned} D[q||p] &= \Gamma(\log \lambda_{qtot}) + \sum_{s=1}^{m} (\lambda_q(s) - 1)(\Psi(\lambda_q(s)) - \Psi(\lambda_{qtot}) - \log \Gamma(\lambda_q(s)) \quad \text{(E.15)} \\ &- \Gamma(\log \lambda_{ptot}) + \sum_{s=1}^{m} (\lambda_p(s) - 1)(\Psi(\lambda_q(s)) - \Psi(\lambda_{qtot}) - \log \Gamma(\lambda_p(s)) \end{aligned}$$

where

$$\begin{aligned} \lambda_{qtot} &= \sum_{s=1}^{m} \lambda_q(s) \\ \lambda_{ptot} &= \sum_{s=1}^{m} \lambda_p(s) \end{aligned} \tag{E.16}$$

and $\Psi()$ is the digamma function.

Bibliography

[1] B. Abraham and J. Ledolter. *Statistical methods for forecasting.* John Wiley, 1983.

[2] J.O. Berger. *Statistical Decision Theory and Bayesian Analysis.* Springer-Verlag, 1985.

[3] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, Oxford, 1995.

[4] P. Bloomfield. *Fourier Analysis of Time Series. An Introduction.* John Wiley, 2000.

[5] M. Boas. *Mathematical Methods in the Physical Sciences.* Wiley, 1983.

[6] G.E.P. Box and G.C. Tiao. *Bayesian Inference in Statistical Analysis.* John Wiley, 1992.

[7] P. Bratley, B. L. Fox, and E. L. Schrage. *A Guide to Simulation.* Springer, 1983.

[8] G. Clifford Carter. Coherence and time delay estimation. *Proceedings of the IEEE*, 75(2):236–255, 1987.

[9] M.C. Casdagli and A.S. Weigend. Exploring the continuum between deterministic and stochastic modeling. In A.S. Weigend and N.A. Gershenfeld, editors, *Time series prediction: forecasting the future and understanding the past*, pages 347–366. Addison-Wesley, 1994.

[10] C. Chatfield. *An Introduction to Multivariate Analysis.* Chapman and Hall, 1991.

[11] C. Chatfield. *The Analysis of Time Series: An Introduction.* Chapman and Hall, 1996.

[12] T.M. Cover and J.A. Thomas. *Elements of Information Theory.* John Wiley, 1991.

[13] A.G. Darbyshire and D.S. Broomhead. Robust estimation of tangent maps and liapunov spectra. *Physica D*, 89:287, 1996.

[14] J.F.G. DeFreitas, M. Niranjan, and A.H. Gee. Hierarchical Bayesian-Kalman Models for Regularisation and ARD in Sequential Learning. Technical Report CUED/F-INFENG/TR 307, Department of Engineering, Cambridge University, 1998. Also avaiable from http://svr-www.eng.cam.ac.uk/ jfgf/publications.html.

[15] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

[16] J.P. Eckmann and D. Ruelle. Ergodic theory of chaos and strange attractors. *Rev. Mod. Phys.*, 57:617–656, 1985.

[17] B. Efron and R.J. Tibshirani. *An introduction to the bootstrap.* Chapman and Hall, 1993.

[18] J.B. Elsner and A.A. Tsonis. *Singular Spectrum Analysis: A New Tool in Time Series Analysis.* Plenum, 1996.

[19] M. Gell-Mann. What is complexity ? *Complexity*, 1(1), 1995.

[20] S. Geman, E. Bienenstock, and R. Doursat. Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, 4:1–58, 1992.

[21] Z. Ghahramani. Learning Dynamic Bayesian Networks. In C.L. Giles and M. Gori, editors, *Adaptive Processing of Temporal Information.* Springer-Verlag, 1999.

[22] Z. Ghahramani and G.E. Hinton. Parameter Estimation for Linear Dynamical Systems. Technical Report CRG-TR-96-2, Department of Computer Science, University of Toronto, 1996. Also available from http://www.gatsby.ucl.ac.uk/ zoubin/papers.html.

[23] L. Glass and D.T. Kaplan. Complex dynamics in physiology and medicine. In A.S. Weigend and N.A. Gershenfeld, editors, *Time series prediction: forecasting the future and understanding the past*, pages 513–528. Addison-Wesley, 1994.

[24] P. Grassberger. An optimized box-assisted algorithm for fractal dimensions. *Phys. Lett. A*, 113:235, 1990.

[25] G. Grimmett and D. Stirzaker. *Probability and Random Processes.* Oxford, 1982.

[26] G. Grimmett and D. Welsh. *Probability, an Introduction.* Oxford, 1985.

[27] S. Haykin. *Adaptive Filter Theory, 3rd Edition.* Prentice-Hall, 1996.

[28] A.H. Jazwinski. Adaptive Filtering. *Automatica*, 5:475–485, 1969.

[29] R.E. Kalman and R.S. Bucy. New results in linear filtering and prediction theory. *Transactions of ASME (Journal of Basic Engineering)*, 83D:95–108, 1961.

[30] H. Kantz and T. Schreiber. *Nonlinear Time Series Analysis.* Cambridge University Press, 1997.

[31] S.M. Kay and S.L. Marple. Spectrum analysis - a modern perspective. *Proceedings of the IEEE*, 69(11):1380–1419, 1981.

[32] D.G. Kleinbaum, L.L. Kupper, and K.E. Muller. *Applied Regression Analysis and Other Multivariable Methods*. PWS-Kent, Boston, 1988.

[33] P. M. Lee. *Bayesian Statistics: An Introduction*. Arnold, 2 edition, 1997.

[34] H. Lutkepohl. *Introduction to Multiple Time Series Analysis*. Springer Verlag, 1993.

[35] D.J.C. Mackay. Bayesian Interpolation. *Neural computation*, 4(3):415–447, 1992.

[36] D.J.C Mackay. *Information Theory, Inference and Learning Algorithms*. Springer, 2000.

[37] J.R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. John Wiley, 1997.

[38] S. Makeig and M. Inlow. Lapses in alertness: Coherence of fluctuations in performance and eeg spectrum. *Electroencephalography and Clinical Neurophysiology*, 86:23–35, 1993.

[39] S.L. Marple. *Digital spectral analysis with applications*. Prentice-Hall, 1987.

[40] R.J. Meinhold and N.D. Singpurwalla. Understanding the Kalman Filter. *The American Statistician*, 37(2):123–127, May 1983.

[41] T. Mullin. *The Nature of Chaos*. Oxford Science Publications, 1993.

[42] A. Neumaier and T. Schneider. Multivariate autoregressive and ornstein-uhlenbeck processes: Estimates for order, parameters, spectral information and confidence regions. *Unknown*, 1999.

[43] J.J.K. O'Ruaniaidh and W.J. Fitzgerald. *Numerical Bayesian Methods Applied to Signal Processing*. Springer, 1996.

[44] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, 1991.

[45] J. Pardey, S. Roberts, and L. Tarassenko. A Review of Parametric Modelling Techniques for EEG Analysis. *Med. Eng. Phys.*, 18(1):2–11, 1996.

[46] W.D. Penny and S.J. Roberts. Gaussian observation hidden Markov models for EEG analysis. Technical report, Department of Electrical and Electronic Engineering, Imperial College, 1998.

[47] S.M. Pincus. Approximate entropy as a measure of system complexity. *Proc. Natl. Acad. Sci.*, 88:2297–2301, 1991.

[48] F.J. Pineda and J.C. Sommerer. Estimating generalized dimensions and choosing time delays: A fast algorithm. In A.S. Weigend and N.A. Gershenfeld, editors, *Time series prediction: forecasting the future and understanding the past*, pages 367–386. Addison-Wesley, 1994.

[49] W. H. Press, S.A. Teukolsky, W.T. Vetterling, and B.V.P. Flannery. *Numerical Recipes in C, second edition.* Cambridge, 1992.

[50] M.B. Priestley. *Spectral Analysis and Time Series Volume 1: Univariate Series.* Acadamic Press, 1981.

[51] J.G. Proakis and D.G. Manolakis. *Digital Signal Processing: Principles, Algorithms and Applications.* Prentice-Hall, 1996.

[52] I.A. Rezek and S.J. Roberts. Stochastic complexity measures for physiological signal analsyis. *IEEE Transactions on Biomedical Engineering*, 44(9), 1998.

[53] B. Ripley. *Pattern Recognition and Neural Networks.* Cambridge University Press, 1995.

[54] J. Rissannen. Complexity of models. In W.H. Zurek, editor, *Complexity, Entropy and the Physics of Information*, pages 117–125. Addison-Wesley, 1990.

[55] J.C. Shaw and D. Simpson. EEG Coherence: Caution and Cognition. *British Psychological Society Quaterly*, 30/31, 1997.

[56] R.H. Shumway and D.S. Stoffer. An approach to time series smoothing and forecasting using the EM algorithm. *Journal of Time Series Analysis*, 3(4):253–264, 1982.

[57] D. W. Skagen. Estimation of Running Frequency Spectra Using a Kalman Filter Algorithm. *Journal of Biomedical Engineering*, 10:275–279, May 1988.

[58] G. Strang. *Linear algebra and its applications.* Harcourt Brace, 1988.

[59] P. R. Tadikamalla. Random sampling from the exponential power distribution. *Journal of the American Statistical Association*, 75:683–686, 1980.

[60] M. E. Tipping and C. M. Bishop. Probabilistic Principal Component Analysis. Technical report, Neural Computing Research Group, Aston University, 1997.

[61] M. E. Tipping and C. M. Bishop. Mixtures of Probabilistic Principal Component Analyzers. *Neural Computation*, 11(2):443–482, 1999.

[62] M. Wax and T. Kailath. Detection of Signals by Information Theoretic Criteria. *IEEE Trans. Acoustics, Speech and Signal Processing*, ASSP-32:387–392, 1985.

[63] A.S. Weigend and N.A. Gershenfeld. *Time series prediction: forecasting the future and understanding the past.* Addison-Wesley, 1994.

[64] S. Weisberg. *Applied Linear Regression.* John Wiley, 1980.