# Clustering of Proteomics Imaging Mass Spectrometry Data

Annie Conway

*Thesis submitted for the degree of*

*Master of Philosophy*

*in*

*Statistics*

*at*

*The University of Adelaide*

*Faculty of Mathematical and Computer Sciences*

School of Mathematical Sciences

January 12, 2016

# Contents

# Signed Statement

# Acknowledgements

I give my foremost thanks to my supervisor Inge Koch for her helpful advice and teaching, as well as her unwavering support and encouragement.

I would also like to thank Lyron Winderbaum and the Adelaide Proteomics Centre for providing data, assistance and information. I acknowledge The University of Adelaide for my scholarship and The School of Mathematical Sciences for giving additional support as well as providing me with a place to work.

Finally, I would like to thank my friends and family for their support and encouragement. Many people were a source of insightful conversation which was valuable for both my research and understanding of mathematics.

# Abstract

This thesis presents a toolbox for the exploratory analysis of multivariate data, in particular proteomics imaging mass spectrometry data. Typically such data consist of 15000 - 20000 spectra with a spatial component, and for each spectrum ion intensities are recorded at specific masses. Clustering is a focus of this thesis, with discussion of $k$-means clustering and clustering with principal component analysis (PCA). Theoretical results relating PCA and clustering are given based on Ding and He (2004), and detailed and corrected proofs of the authors' results are presented. The benefits of transformations prior to clustering of the data are explored. Transformations include normalisation, peak intensity correction (PIC), binary and log transformations. A number of techniques for comparing different clustering results are also discussed and these include set based comparisons with the Jaccard distance, an information based criterion (variation of information), point-pair comparisons (Rand index) and a modified version of the prediction strength of Tibshirani and Walther (2005).

These exploratory analyses are applied to imaging mass spectrometry data taken from patients with ovarian cancer. The data are taken from slices of cancerous tissue. The analyses in this thesis are primarily focused on data from one patient, with some techniques demonstrated on other patients for comparison.

# Notation Index

| | |
|---|---|
| $\mathbb{R}$ | the real numbers. |
| $p$ | number of variables. |
| $n$ | number of observations. |
| $x$ | (observation of) a functional random variable. |
| $\mathbf{x}$ | (observation of) a multivariate random variable, $p \times 1$. |
| $\mathbb{X}$ | data matrix, $p \times n$. |
| $\boldsymbol{\mu}$ | the mean of $\mathbf{x}$, $p \times 1$. |
| $\Sigma$ | the covariance matrix of $\mathbf{x}$, $p \times p$. |
| $\bar{\mathbf{x}}$ | the sample mean of $\mathbb{X}$, $p \times 1$. |
| $S$ | the sample covariance matrix of $\mathbb{X}$, $p \times p$. |
| $d(\mathbf{x}_1, \mathbf{x}_2)$ | distance between two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$. |
| $\delta(C_1, C_2)$ | distance between two sets $C_1$ and $C_2$. |
| $k$ | number of clusters. |
| $\mathcal{C} = \{C_1, \ldots, C_k\}$ | a $k$-cluster arrangement. |
| $\mathcal{P}(\mathbb{X})$ | the power set of $\mathbb{X}$, i.e. the set of all subsets of $\mathbb{X}$. |
| $m/z$ | mass-on-charge. |

# Chapter 1

# Introduction

Proteomics mass spectrometry is a powerful tool for the study and identification of proteins in organisms. Proteomics has provided new insight into human diseases, such as biomarker discovery for early detection of cancers and the discrimination of different cancer types.

In this thesis we consider proteomics mass spectrometry data from tissue samples of ovarian cancer. Ovarian cancer is the deadliest of all gynaecological cancers. In early stages, ovarian cancer does not present symptoms, and hence the disease is often not detected until it has reached an advanced stage. Researchers are eager to find new ways to detect the disease at an early stage, as this can significantly increase the likelihood of patient survival.

Many different methodologies have been developed in the last few decades within proteomics mass spectrometry. In this thesis I focus on matrix assisted laser desorption/ionisation (MALDI) for imaging mass spectrometry (IMS), and on data acquired from tissue samples of patients with ovarian cancer using MALDI-IMS. The Adelaide Proteomics Centre has made important advances in MALDI-IMS in recent years. In this thesis I am working with their imaging data.

A mass spectrum is usually presented as a function: intensity versus mass-to-charge. In MALDI-IMS the charge is typically 1, so we are essentially considering intensities of mass, but we use the term mass-to-charge as it is standard for mass spectrometry (MS). We observe a large number of spectra, typically in the tens of thousands.

This thesis is concerned with both exploratory data analysis, as well as the theoretical basis for exploratory methods. I primarily consider *cluster analysis*, which is also referred to in the literature as *segmentation* (particularly in proteomics applications) and *unsupervised learning* or *machine learning*. I consider different ways to cluster data, with most emphasis on $k$-means. I give special consideration to ways of measuring *distance* between observations, and demonstrate how using different distances can give very different results. My goal is to create a toolbox together with discussions and comparisons of the effects of the different tools. These tools may be useful to the bioinformatician, biologist or statistician interested in analysing or clustering proteomics data.

All known material discussed in the literature are referenced in the relevant chapter. Results unattributed are my own, and all data analyses are my own.

## 1.1 Chapter overview

In Chapter 2 I explain both the motivation for this project and how the data are obtained with MALDI-IMS.

In Chapter 3 I consider the functional nature of the data, and put forward a review and discussion of functional data analysis. I discuss the common approaches to functional data analysis with respect to the smoothing and interpolation of data. Since the IMS data tend to be sparse with many isolated peaks, we bin the mass spectra onto a fine grid and then treat the data as vectors.

Preparation of functional data is extended in Chapter 4, where I consider normalisation of IMS data. Normalisation is an important step in preparing the data for analysis as the spectra are badly affected by systematic machine errors. I review a number of methods which are commonly used, but these do not give promising results. In collaboration with Winderbaum (2015), I have put forward a new method of normalisation, known as peak intensity correction (PIC) which corrects the spectra according to internal calibrants. This is related to the work on removal of unwanted variance (RUV) proposed by Gagnon-Bartsch and Speed (2012).

Cluster analysis is explored in Chapter 5, where I use $k$-means clustering on the vectorised data. A rough aim of this analysis is to partition the data into sections that match with the different types of tissue. I have given consideration towards the choice of distance measure in $k$-means, and demonstrate how different distance measures can give vastly different clustering results. I also cluster data after normalisation and transformations such as a binary representation and a log transform.

In Chapter 6 I look closely at the intersection of $k$-means clustering and principal component analysis (PCA). In particular I look at a proposition put forward by Ding and He (2004) that the first principal component gives the solution to a continuous relaxation of the $k$-means objective function when $k = 2$. I give rigorous proofs of their theorems and lemmas, and I have corrected for a number of mistakes in the original arguments. I also discuss limitations and possible practical uses for the theory.

After having produced many clustering results in Chapters 5 and 6, in Chapter 7 I look at quantitative ways to compare cluster arrangements. I have chosen four approaches to this: a set based method, information based method, a pairwise method and a method adapted from Tibshirani and Walther's prediction strength.

With any exploratory analysis, it is important to try different approaches as this in itself may reveal important information about the data. For example, using $k$-means clustering on the raw data with both Euclidean and cosine distances can give very different results, which can indicate some interesting features, such as the effect of error in the data.

# Chapter 2

# Proteomics Imaging Mass Spectrometry

In the past few decades much progress has been made in the field of *genomics*, which provides a basis for understanding the uniqueness and variability in humans as differences in DNA sequences (Mishra, 2011, chapter 1). More recently, a breakthrough has been underway in the field of *proteomics* which extends the work of genomics by considering the roles of protein interactions. If genes are like a code, then proteins are like the machinery that implements the code in a given cell.

The term *proteome* is a combination of the terms *protein* and *genome*, and it is defined as the total proteins encoded by the genome of an organism. Unlike the genome, which remains the same in all cells at all times, the proteome is dynamic, differing in different cell types and even changing for a given cell type at different stages in development.

The term proteomics refers to quantitative analysis of the proteome of a given cell, tissue (e.g. cancer) or biological fluid (e.g. serum) at a given point in time or under the effects of a defined biological stimulus (Gustafsson et al., 2011).

In this chapter I will briefly outline the role of proteomics in disease research (particularly ovarian cancer) and some of the aims that motivate the project.

## 2.1 Proteomics and ovarian cancer

Genetics and genomics have been instrumental in the study of human diseases. Genes, or combinations of genes and environmental factors, can be linked to certain diseases and this has provided substantial understanding into how and why a person contracts that disease. Proteomics has also provided insight into the role of genes and environmental factors by identifying the roles of different proteins and their interactions in metabolic pathways. Proteomics can help us understand the biochemical basis of a disease, and also its diagnosis and treatment. In particular, proteomics helps us explain:

- why certain cancers can be controlled by certain drugs and not by others,
- side effects of particular drugs,
- whether a particular person will respond to a particular treatment,
- and proteomics can also identify proteins that indicate a certain disease (biomarkers).

In this project we deal with tissue samples of ovarian cancer. Of all the gynaecological cancers, ovarian cancer has the highest mortality rate. The asymptomatic nature of the disease means that most cases go undiagnosed until the cancer has reached an advanced stage, by which time most treatments are not successful. However, early detection of the cancer leads to a higher five-year survival rate and a higher probability of cure (Gustafsson et al., 2011). There is currently no effective screening test for ovarian cancer and diagnosis is difficult, so researchers are searching for biomarkers to improve ovarian cancer detection.

So far a number of proteins have been associated with ovarian cancer: Leptin, prolactin, Osteopontin, and insulin-like growth factor II (Mishra, 2011, chapter 6). The search for relevant proteins is still in its infancy, and more research on the identification of proteins associated with biomarkers and diseases is needed, both

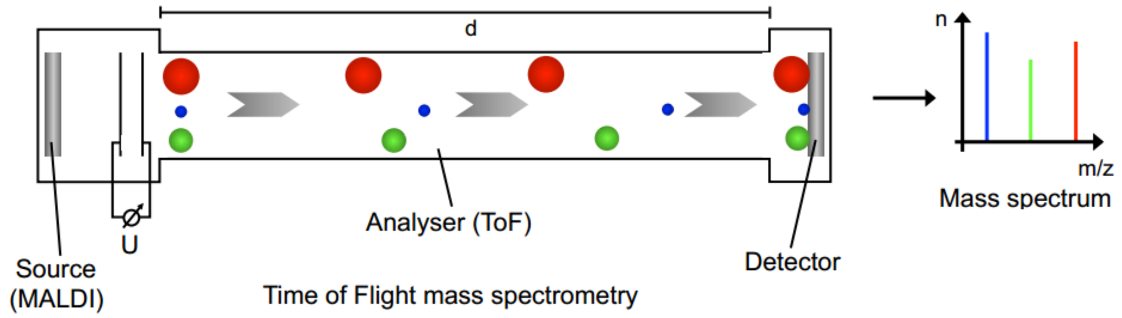


Figure 2.2.1: Diagram illustrating the process of MALDI-TOF mass spectrometry. Source: Hoffmann 2012.

for diagnosis and treatment. Statistical analyses of proteomics data can help in pointing to or identifying important proteins.

## 2.2 Mass spectrometry

The data for this project has been acquired via *mass spectrometry* (MS), and more specifically, imaging mass spectrometry. Mass spectrometry is a powerful tool in biochemistry, by which we can identify the abundance of a molecule or peptide, which points to the presence of a certain protein. From Gross (2010, chapter 1):

> The basic principle of mass spectrometry (MS) is to generate ions from either inorganic or organic compounds by any suitable method, to separate these ions by their mass-to-charge ratio (m/z) and to detect them qualitatively and quantitatively by their respective m/z and abundance.
>
> (...) A mass spectrum is the two-dimensional representation of signal intensity (ordinate) versus m/z (abscissa). The position of a peak, as signals are usually called, reflects the m/z of an ion that has been created from the analyte within the ion source. The intensity of this peak correlates to the abundance of that ion.

Essentially, the sample is ionised - in our case by the process known as MALDI (matrix-assisted laser desorption/ionization) and the ions are separated based on their masses. The method by which this is done in our case is ToF (time of flight). A simple illustration of this process is given in Figure 2.2.1. The results of this process are displayed in a 2-dimensional plot known as a *mass spectrum*, which has *ion intensity* on the vertical axis and $m/z$ on the horizontal axis. A peak at a given $m/z$ value indicates the abundance of an ion with that particular mass-to-charge ratio. In MALDI experiments the charge is typically controlled to be 1, i.e. $z = 1$ so the horizontal axis can be thought to represent mass.

The data in this project will be referred to as *imaging mass spectrometry* (IMS) as each mass spectrum is associated with a point in 2-dimensional space on a tissue sample. I describe IMS in more detail in Section 2.3.

## 2.3 A brief overview of the acquisition of data for this project

For details, see Gustafsson (2011, chapter 7).

We consider data from surgically excised ovarian cancers. Samples are available from three patients (Patient 44, Patient 173 and Patient 540). For the majority of analyses in this project, I will be looking at Patient 44. Tissue structure is visualised by a pathologist using histological stains such as haematoxin and eosin (H&E), such as in the example in Figure 2.3.1. To give an idea of the size, the black line in this image represents 5mm.

Figure 2.3.2 details the steps in the preparation of the tissue samples. A sample of tissue is divided into thin (2-10 $\mu$m) slices and mounted onto conductive microscopy slides. These tissue samples include cancerous tissue as well as healthy tissue (adipose and stroma).

An overview of the workflow, illustrated in Figure 2.3.2:

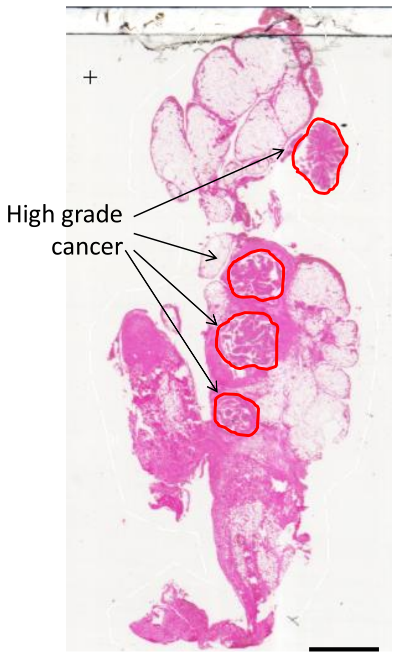


Figure 2.3.1: H&E stain for the sample from patient 44, slide 4. The high grade cancer regions have been highlighted by the pathologist. The black line in this image represents 5mm.

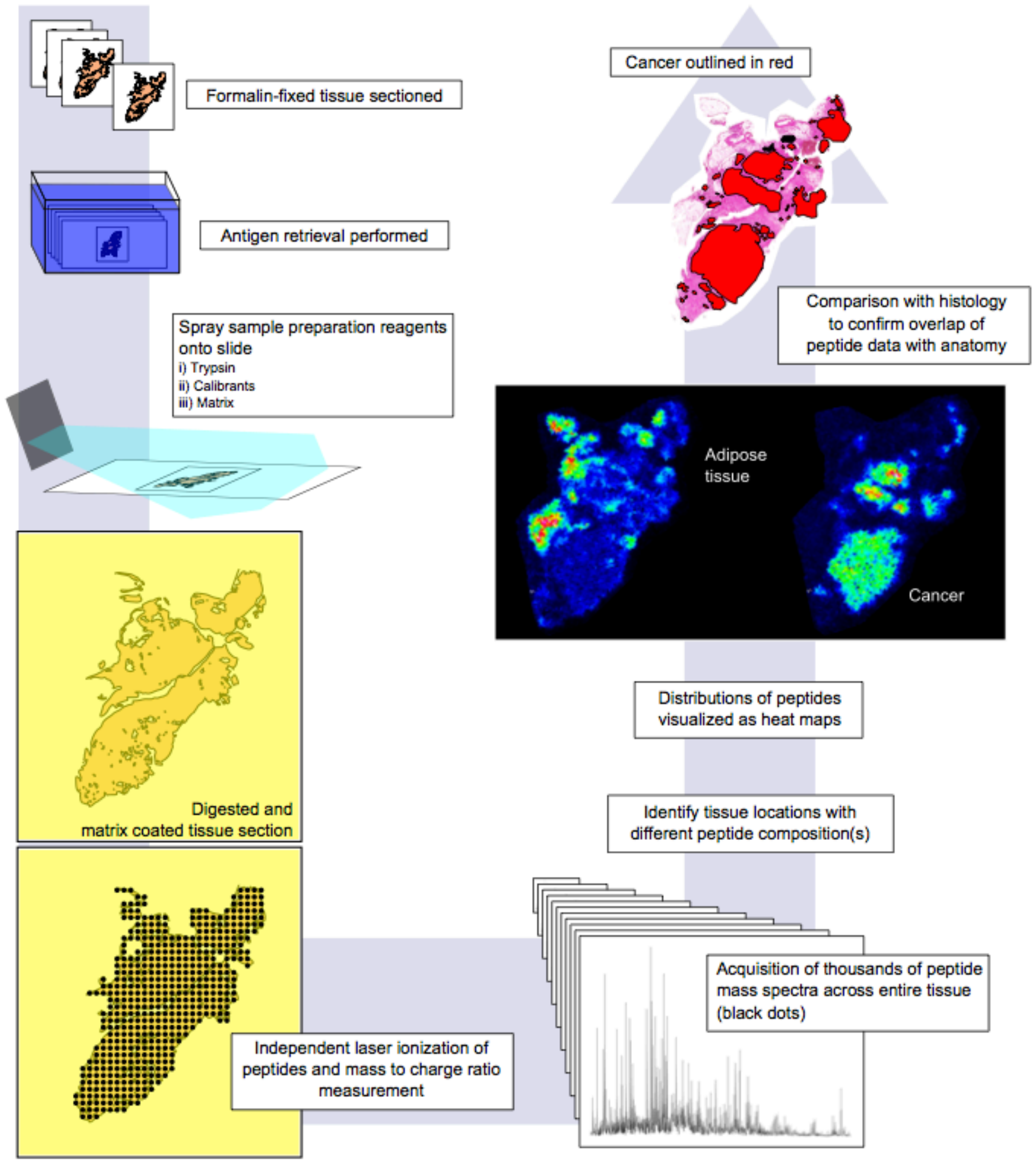

Formalin-fixed tissue sectioned
Antigen retrieval performed
Spray sample preparation reagents onto slide
i) Trypsin
ii) Calibrants
iii) Matrix
Digested and matrix coated tissue section
Independent laser ionization of peptides and mass to charge ratio measurement
Acquisition of thousands of peptide mass spectra across entire tissue (black dots)
Identify tissue locations with different peptide composition(s)
Distributions of peptides visualized as heat maps
Adipose tissue
Cancer
Comparison with histology to confirm overlap of peptide data with anatomy
Cancer outlined in red

Figure 2.3.2: Workflow diagram for the preparation and analysis of tissue samples. Source: Gustafsson 2011.

1. A sample of tissue is divided into thin (2-10 $\mu$m) slices and mounted onto conductive microscopy slides (see first step in Figure 2.3.2).

2. The tissue sections are prepared with antigen retrieval, trypsin digestion and matrix deposition (see second, third steps in Figure 2.3.2).

3. A 2-dimensional grid is superimposed onto the sample. The grid spacing used is 100 $\mu$m (see yellow image in Figure 2.3.2).

4. MALDI-IMS data are collected using a Bruker Ultraflex III and a $m/z$ range of 1000-4500. A mass spectrum is acquired for every point on the grid. The number of mass spectra in total ranges from 5000 to 100000. A single mass spectrum corresponding to one grid point is shown bottom right of Figure 2.3.2.

5. The distribution of certain peptides can be displayed in heat maps. The image also gives an indication of the spatial distribution of the peptide, and this can be compared with the histology (see final steps in Figure 2.3.2).

One of the advantages of MALDI-IMS is that we can undertake a direct analysis of the tissue and preserve the spatial information. For instance, we can see how the abundance of certain ions can relate to tissue types by comparing the heat map images in Figure 2.3.3 with the histology in Figure 2.3.1. Through statistical analysis, these comparisons can be made objectively. With clustering we can determine regions where the spectra are most similar. We can also improve the data collection (through normalisation) and with classification and prediction we can make decisions and and identify biomarkers related to disease.

## 2.4 Conclusion

In this chapter I have briefly introduced the motivation for analysis of proteomics IMS data. I have also indicated the nature of the data, and I will give a more mathematical explanation in the next chapter.

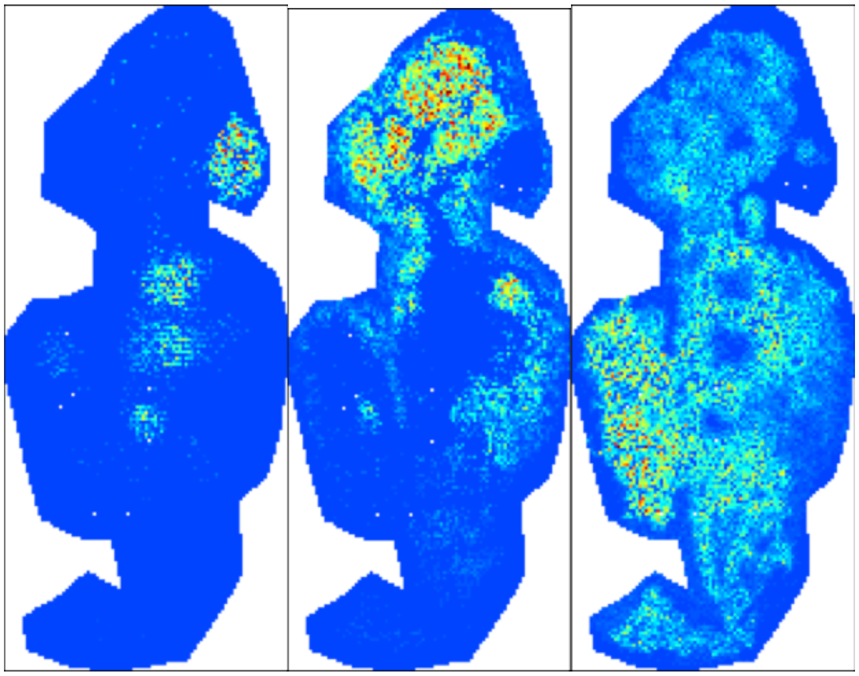


Figure 2.3.3: Comparison of $m/z$ heat maps for patient 44, slide 4. From left to right, intensities of particular peptides that indicate cancer, adipose and stroma respectively.

We have seen that the data are in the form of 1000s of mass spectra and these spectra can be thought of as ion intensities as a function of $m/z$. This means that we are considering a special type of data known as *functional data*. This will be the topic of the next chapter.

# Chapter 3

# Functional Data Analysis

The proteomics data which are of primary interest in this project may be treated as *functional data.* The purpose of this chapter is to establish the basic ideas of functional data, to illustrate how functional data extend and relate to multivariate data, and how functional data ideas apply to IMS data.

## 3.1 Introduction

This chapter is largely based on two texts: *Functional Data Analysis* by Ramsay and Silverman (1997, chapters 1 & 3), and *Nonparametric Functional Data Analysis* by Ferraty and Vieu (2006, chapter 1).

Consider an observation of some (single) continuous phenomenon for which we take discrete measurements. A typical example would be a process observed over an interval of time, the variables being the time steps $t_1, t_2, \ldots, t_k$. In the imaging mass spectrometry (IMS) data, the variables correspond to the mass-to-charge $m/z$ values. Mass spectra are regarded as continuous.

Even though these variables will be recorded discretely, they will typically be close, so it makes intuitive sense to consider the data as an observation $x$ of the continuous family:

$$X = \{x : T \to \mathbb{R} \,|\, T = [t_1, t_k] \subset \mathbb{R}\}. \tag{3.1.1}$$

It is important to note the distinction here between the model or conceptual level and the practical level. Since data are always observed discretely, the classification of data as *functional* is essentially theoretical. However, modelling the data as functions may be appropriate for the given situation. This is typically the case when we are considering time steps, since time is continuous, regardless of whether we observe it discretely. The same is true for mass values, wavelengths and so on.

The goals of functional data analysis are similar to any kind of statistical analysis: to represent the data in a way that best aids analysis, to detect patterns, study the relationships between independent and dependent variables and to compare different sets of data. Treating data as functions makes analysis with derivatives more intuitive, which is useful in applications where rate of change and other continous transformations of the data are of interest.

The analysis here will be assumed to be *nonparametric*. There are varying definitions of nonparametric in the literature, but here we take it to mean the approach is nonparametric if we do not require or refer to any underlying distribution in the data. This is a natural idea, since it is not particularly helpful to think of distributions such as normal or exponential in the functional case. Note that Ferraty and Vieu (2006, p.7), make the distinction between fitting a model to the data (e.g. a linear model) for which parameters should be estimated, and fitting some general and flexible continuous function to the data and call these parametric and nonparametric respectively.

### 3.1.1 The extension of multivariate to functional

To gain better insight into functional data, I will begin by comparing it with multivariate data. In this thesis, multivariate data are data in which each observation is associated with $p > 1$ discrete variables. In the same way that the integral is the

Table 3.1.1: Summary of Data Types.

| | Number of Variables $p$ |
|---|---|
| Univariate | $p = 1$ |
| Multivariate | $p > 1$ |
| High-dimensional | $p \gg 1$ (In a sample of $n$ observations, might have $p > n$). |
| Functional | $p = \infty$, that is, $p$ is continuous. |

continuous extension of the sum, I would like to consider functional data analysis as a continuous extension of multivariate data analysis.

Table 1 compares the different data types by their number of variables. A distinction is made between multivariate and high-dimensional data, since in practice these cases may need different treatment. As an aside, it is important to note that multivariate data allows for different variable types (e.g. height, weight etc) whereas functional data only allows for data of the same type (e.g. the variables represent different times, or different wavelengths.)

In multivariate analysis an observation will consist of measurements with a discrete and finite number of variables. Consider the population case $\mathbf{x} \in \mathbb{R}^p$ where $p$ is the number of variables. The analogous functional example of this is $x \in L^2$, i.e. we take the observation to be some square-integrable function $\mathbb{R} \to \mathbb{R}$. Conveniently $L^2$ is a Hilbert space, which leads to useful results to be discussed shortly.

The multivariate random vector, $\mathbf{x}$ has a mean, $\boldsymbol{\mu} \in \mathbb{R}^p$ and likewise the functional random variable $x$ has a mean $\mu \in L^2$.

In multivariate analysis we are typically interested in the covariance matrix $\Sigma^{(p)}$. Since

$$\Sigma^{(p)}\boldsymbol{\eta} = \boldsymbol{\eta}^*$$

where $\boldsymbol{\eta}, \boldsymbol{\eta}^* \in \mathbb{R}^p$, we may treat $\Sigma^{(p)}$ as a linear operator: $\Sigma^{(p)} : \mathbb{R}^p \to \mathbb{R}^p$.

In the functional case, we can no longer think of the covariance as a matrix, but its interpretation as a bounded linear operator still applies. Hence we may write:

$$\Sigma : L^2 \to L^2. \tag{3.1.2}$$

Covariance matrices are always symmetric, and a similar notion can be noted for this operator. Typically we require that the operator is *self-adjoint*.

We may be more specific in talking about the operators here. In the matrix case, the operator has finite rank (since all matrices will have finite dimension when dealing with multivariate data). In the functional case, we can no longer assume the operator is a finite rank operator but we may regard them as *compact* operators, as this idea naturally extends that of finite rank (Wang et al., 2015). A definition of compact operators is given by Garrett (2012):

**Definition 3.1.1.** *(Garrett, 2012) A linear operator $T : X \to Y$ from a Hilbert space $X$ to a Hilbert space $Y$ is* compact *if and only if it maps bounded sequences in $X$ to sequences in $Y$ with convergent subsequences.*

For a self-adjoint compact operator $\Sigma$ we may define a corresponding $\lambda$-eigenspace,

$$X_\lambda = \{x \in X \mid \Sigma x = \lambda x\} \tag{3.1.3}$$

which naturally extends the notion of eigenvectors from the matrix case. We observe the following results for this eigenspace given by Garrett (2012):

**Theorem 3.1.1.** *For a compact operator $\Sigma$, defined on $L^2$ with corresponding eigenspace $X_\lambda$ the following hold:*

1. *The completion of the direct sum of $X_\lambda$ for all $\lambda$ is the original Hilbert space $L^2$. That is, there is an orthonormal basis consisting of eigenvectors.*

2. *The eigenspaces corresponding to the discrete eigenvalues are finite dimensional.*

3. *All the eigenvalues are real.*

Deriving summary statistics for functional data is similar to the multivariate case. However, statistics such as sample mean and variance are typically evaluated at a given $t$.

$$\text{Mean:} \quad \bar{x}(t) = \frac{1}{n}\sum_{i=1}^{n} x^{(i)}(t) \tag{3.1.4}$$

$$\text{Variance:} \quad \text{var}_x(t) = \frac{1}{n-1}\sum_{i=1}^{n}[x^{(i)}(t) - \bar{x}(t)]^2 \tag{3.1.5}$$

$$\text{Covariance:} \quad \text{cov}_x(t_1, t_2) = \frac{1}{n-1}\sum_{i=1}^{n}[x^{(i)}(t_1) - \bar{x}(t_1)][x^{(i)}(t_2) - \bar{x}(t_2)], \tag{3.1.6}$$

where $x^{(i)}$ is the $i$th function in an observed sample of $n$.

## 3.2 Smoothing of functional data

An observation of functional data (a functional datum) will be observed as a set of discrete values $x = \{x(t_1), \ldots, x(t_n)\}$ (note that these may not necessarily be equally spaced on the variable axis). If the underlying process is a continuous process, and the measurements are made at discrete and possibly unequal steps and corrupted by noise, it is often advantageous to first convert to a function $x(t)$. This may be done with *interpolation*, however if we expect the data to have noise present then we require *smoothing*. After this, we may discretise the data again with the freedom to choose spacing on the variable axis.

In the next part I will discuss two main approaches to smoothing: the basis function approach, whereby a function is expressed in terms of some chosen basis, and also a local weighting approach. This section serves as a brief introduction to common methodologies in functional analysis, and although not used directly in this project, they are of interest and could be applied to the proteomics data instead of the peak picking and binning approach which I use in this thesis and which are described in Section 3.4.

### 3.2.1 The basis function approach

A common approach to smoothing is to express the function $x : \mathbb{R} \to \mathbb{R}$ as a linear combination of $K$ basis functions $\phi_k : \mathbb{R} \to \mathbb{R}$, $k = 1, \ldots, K$:

$$x(t) = \sum_{k=1}^{K} c_k \phi_k(t). \tag{3.2.1}$$

If the datum is received as $n$ points on the function, then the basis function will be evaluated at each variable point $t_j$, $j = 1, \ldots, n$. A typical way to evaluate the coefficients $c_k$ for each basis would be to use a least squares fit. That would be the minimiser of the least squares criterion:

$$\text{SMSSE}(x|c) = \sum_{j=1}^{n} \left[ x_{t_j} - \sum_{k=1}^{K} c_k \phi_k(t_j) \right]^2, \tag{3.2.2}$$

where $n$ is the number of discretised points of the function and $K$ is the total number of basis functions and coefficients to be estimated. This can also be expressed in matrix notation:

$$\text{SMSSE}(x|c) = (\mathbf{x} - \Phi\mathbf{c})^T(\mathbf{x} - \Phi\mathbf{c}) = \|\mathbf{x} - \phi\mathbf{c}\|^2$$

where $\Phi = \{\phi_k\}$ is the $(n \times K)$ matrix of basis functions evaluated at $t_j$ and $\mathbf{x}$ and $\mathbf{c}$ are $n$ and $K$ dimensional vectors, respectively. The solution to the least squares criterion is the smoothing matrix $SM = \Phi(\Phi^T\Phi)^{-1}\Phi^T$. This is a projection matrix, hence we may think of least squares smoothing as a projection onto the space spanned by the basis functions.

The choice of basis functions is an important issue. It is natural to choose a basis function that shares properties with our model of what the observed functions represent. It is also important to keep in mind what sort of analysis we wish to do with the data, for example whether we intend to consider derivatives, since this will impact on the choice of function.

### Fourier series

A common choice of basis function is a Fourier basis. Fourier series are used widely in mathematics for approximating functions. The Fourier series is of the following form:

$$\hat{x}(t) = c_0 + c_1 \sin \omega t + c_2 \cos \omega t + c_3 \sin 2\omega t + c_4 \cos 2\omega t + \dots \tag{3.2.3}$$

which is defined by the basis: $\phi_0(t) = 1$, $\phi_{2r-1}(t) = \sin r\omega t$ and $\phi_{2r}(t) = \cos r\omega t$.

Although the Fourier basis is a popular choice, it is not without disadvantages. This approach to smoothing works best when the underlying function is without discontinuities and strong local features, and curvature tends to be of the same order everywhere. For IMS data, which are full of peaks, such smooth models are not appropriate.

### Polynomial bases

Polynomial bases are also a popular approach. This means choosing bases of the form $\phi_k(t) = (t - \omega)^k$, $k = 0, \dots, K$, $\omega \in \mathbb{R}$. As with Fourier bases, polynomial bases work well when the function does not have too many strong local features (large amounts of local variation would require a large $K$). Polynomials tend to fit well in the centre, but may exhibit strange behaviour in the tails.

### Regression spline bases

The regression spline approach to smoothing addresses the problem that Fourier and polynomial bases have with ignoring local features. It essentially creates a piecewise polynomial function (where the pieces of polynomials in question are typically cubic) by joining them together smoothly at certain values $\tau_k$, called *knots*. The placement of knots is often arbitrary, although knot placement should be dense around areas with high variability, so as to capture that variability.

**Wavelet bases**

Wavelet bases have been used in the literature for proteomics data, notably by Morris (2012). Typically a function $\Psi$ is chosen as the *mother wavelet* and the bases are formed from contractions and translations of that base function:

$$\Psi_{jk}(t) = 2^{j/2}\Psi(2^j t - k) \tag{3.2.4}$$

for integers $j$ and $k$. These functions are constructed such that the resulting basis is orthogonal. Wavelet bases tend to work well when dealing with functions with sharp local features.

### 3.2.2 Kernel smoothing

Smoothing with local weighting means that we smooth based on local observations without the use of basis functions. This is usually done with a kernel function. A commonly used kernel function is the Gaussian kernel:

$$\texttt{kern}(u) = (2\pi)^{-1/2}\exp(-u^2/2). \tag{3.2.5}$$

Weights functions are defined subsequently as:

$$w_j(t) = \texttt{kern}\left(\frac{t_j - t}{h}\right). \tag{3.2.6}$$

This weight is concentrated for values of $t_j$ centred at the point of interest $t$. The parameter $h$, the *bandwidth*, controls the degree of concentration or smoothness. A small $h$ means that only values very close to $t$ are given weight, whereas a large $h$ means a larger range of values are averaged.

From these weights we derive the *kernel estimator*:

$$\hat{x}(t) = \sum_{j=1}^{N} W_j(t) y_j. \tag{3.2.7}$$

where $W_j$ is some suitably designed weight function based on the $w_j$.

## 3.3 Preprocessing and interpolation of IMS data

The aim of this section is to build a bridge between functional data and the form of the data that are to be used in the following chapters.

In this project, we consider data from imaging mass spectrometry (IMS). At each pixel in the image a mass spectrum is observed. We model, $x_j$, the $j$th spectrum, evaluated at the $m/z$ value $t$ like so:

$$x_j^{[raw]}(t) = B_j(t) + N_j S_j(t) + \epsilon_j(t). \tag{3.3.1}$$

The key piece of information in this equation is $S_j(t)$, the underlying signal at $t$, and all other quantities refer to measurement errors or artefacts. We aim to remove most of these errors with smoothing, or more generally, data preprocessing.

Morris (2012) outlines the following steps involved in preprocessing:

- Alignment. This involves matching up peaks from $m/z$ values known to be present, typically internal calibrants added for this purpose.
- Denoising and smoothing. This removes the white noise $\epsilon_j(t)$.
- Baseline correction. This removes the smooth background artefacts $B_j(t)$, caused by systematic ion artefacts.
- Peak picking. Multiple peaks in a certain neighbourhood are combined to represent a single peak and peaks below a threshold value are discarded as noise and the value is set to 0.
- Normalisation. This aims to remove the $N_j$ coefficient, which is a systematic error on each spectrum.

The first three steps are taken before I receive the data, and are carried out within the machine (in this case Bruker Ultraflex III) with some parameter choices the user can apply. The final step, *normalisation*, I will discuss in detail in the next chapter.

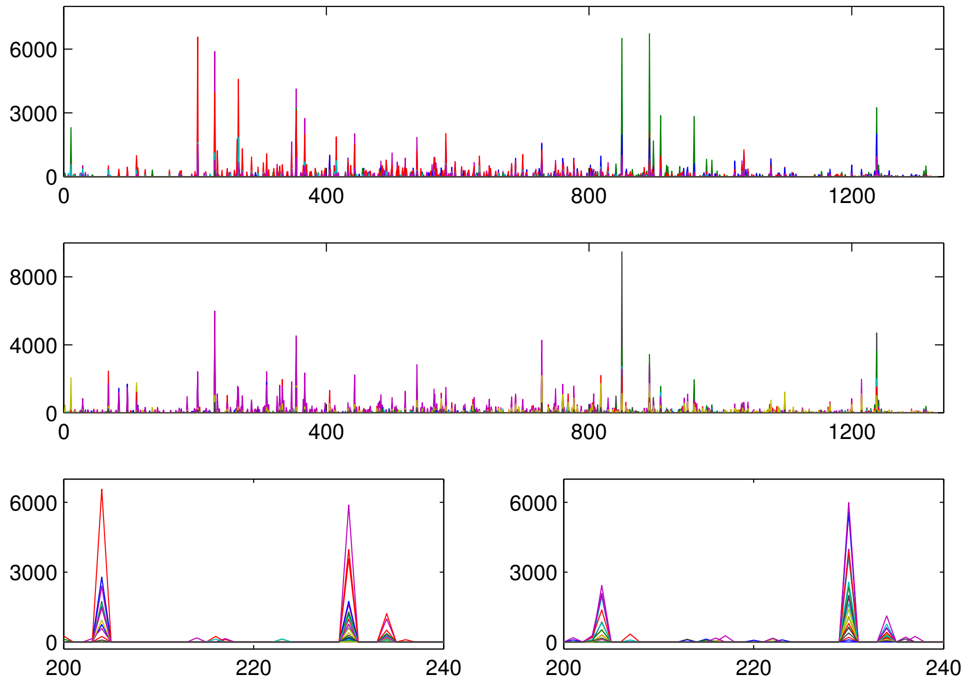


Figure 3.3.1: Mass spectrometry profiles from the ovarian cancer tissue sample patient 44, slide 4; peak intensities on the y-axis against m/z-values on the x-axis for profiles 400-420 in the top panel and with zoom-ins in the bottom left panel, and profiles 1000-1020 in the middle panel with their zoom-ins in the bottom right panel. Each coloured line refers to one spectrum.

We will refer to the following expression as the equation for the peak list data, evaluated at t:

$$x_j^{[peak]}(t) = N_j S_j(t) \epsilon_j(t). \tag{3.3.2}$$

We assume there is still some amount of random error present in the sample, included in the $\epsilon_j(t)$ term, as well as the systematic, spectrum specific error $N_j$.

## 3.4 Interpolation

I receive data as peak lists $x^{(j)} = \{x(t_1), \ldots, x(t_{m_j})\}$, as in Equation (3.3.2). Only the $m/z$ values where intensities are recorded are given, and naturally these will be different for each spectrum. The number of peaks in each spectrum will vary. In order to make the spectra comparable, they will need to be interpolated onto a common grid. This can be done via binning.

I previously described the process of interpolation, smoothing and discretisation of the smoothed function. If the peaks are very dense, the three steps would need to be applied, but due to the relative sparsity of peaks on the $m/z$ axis, binning is sufficient. As a side note, it would also be possible to interpolate the peaks with kernel density smoothing; each peak would then be replaced by a Gaussian curve, and from there the function could be discretised again. However, for this project, I only consider binning.

Ideally we would like to choose the bin width small enough so that at most one peak would fall into any given interval. It is expected that differences in peaks are typically greater than 1 Dalton on the $m/z$ axis, although sometimes the differences are approximately 0.5 Dalton, due to experimentation error. Consequently, a bin width of 0.25 works sufficiently well for ensuring at most only one peak is present in each bin. An additional note about binning: the position of the bin might have some effect on the result, similar to what one observes when producing histogram plots. However, we found that for clustering this shift of bins did not noticeably affect the analysis, and we therefore do not consider it further.

By interpolating spectra into $m/z$ bins, we are essentially converting our functional data (spectra) into multivariate data (vectors). After interpolation many $m/z$ values have 0 peaks across all observations. Since we are no longer concerned with the whole spectra, but rather points of comparison between the spectra, these points can be eliminated. In the sample Patient 44, slide 4, we reduce the number of $m/z$ bins from 13,000 to 5,000 by deleting bins that have 0 intensity across all pixels after binning. The number of bins will henceforth be referred to as $p$, and this is equivalent to the number of variables.

We now consider a data matrix $\mathbb{X} = [\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_n]$. The size of the matrix is $p \times n$, that is, the rows correspond to variables and the columns to observations. This orientation of the data matrix varies in the statistical literature, but this is the convention used by Koch (2013).

## 3.5 Conclusion

Although the data obtained from IMS are functional by nature, we have the ability to treat them as multivariate data. Moreover, much of the analysis of multivariate data (such as principal components) can be extended to functional data.

From this point forward, data is usually seen in the form of the peak list $x^{(j)} = \{x(t_1), \ldots, x_(t_k)\}$ or the multivariate data matrix $\mathbb{X}$. In the next chapter I will look closer at the problem of normalisation, and later investigate the data with cluster analysis.

# Chapter 4

# Normalisation

In the previous chapter I explained the four preprocessing steps necessary in preparing IMS data for analysis: smoothing, baseline correction, peak picking and normalisation. I will now discuss *normalisation* in more detail. Since the data that are the focus of this project have been preprocessed in all ways except normalisation, it is necessary to spend some time choosing and comparing methods to do this.

## 4.1 Motivation for normalisation

In each mass spectrum we assume there is some distortion or suppression of the intensities and this is constant for a given spectrum. We denote this constant for the $j$th spectrum by the non-zero multiplicative factor $N_j$. Normalisation, defined by Deininger et al. (2011) is the process of multiplying the $j$th spectrum by $1/\hat{N}_j$ where $\hat{N}_j$ is an estimator for $N_j$.

Normalisation is a vital step because we need our data to be comparable and informative. Recall our assumed model for the mass spectra:

$$x_j^{[peak]}(t) = N_j S_j(t) \epsilon_j(t). \tag{4.1.1}$$

We wish to get as close as we can to working with the underlying signals $S_j$, or at the very least, removing enough artefacts to make the spectra comparable with each other.

This is especially important in the following chapters when we will be using clustering methods to distinguish different tissue types in the sample. As we will see, clustering data without normalising first can lead to results that are uninformative, or even misleading.

Our choice of normalisation method - that is, how we estimate $N_j$ - is also important. A poor choice of $\hat{N}_j$ could make the data even more misleading and uninformative.

## 4.2 Some simple normalisation techniques

From this point onwards we consider a mass spectrum as a vector of observed intensities, or a vector of peaks. At this stage it is not necessary to have interpolated the peaks onto a grid of equal length; in all these cases I will perform normalisation before interpolation, as once interpolation has been done the vector is full of placeholder 0s and these should have no influence on the normalisation.

We will use vector notation to denote the $k$th spectrum:

$$\mathbf{x}_j = [x_1, x_2, \dots, x_n]^T. \tag{4.2.1}$$

Hence the resulting normalised vector will be:

$$\mathbf{x}_j^{[norm]} = \frac{1}{N_k}\mathbf{x}_k. \tag{4.2.2}$$

In their paper, Deininger et al. (2011) have given a comprehensive review of common normalisation methods currently being used in practice in proteomics MS:

- $\ell_1$ norm, also called total ion count (TIC),
- $\ell_2$ norm, also called mean squared error (MSE),
- $l_\infty$ norm, i.e. the maximum intensity,

- the median intensity.

I implement each of these normalisations for comparison. In the second half of this chapter I propose a new normalisation method which makes explicit use of the calibrant quantities in the sample.

### 4.2.1 Normalisation factors

For $P > 0$ we define the $\ell_P$-norm, $\hat{N}^{(P)}$, like so:

$$\hat{N}^{(P)} = \left( \sum_i |x_i|^P \right)^{1/P}. \tag{4.2.3}$$

We consider three cases of the $\ell_P$-norm. For $P = 1$ we are essentially taking the normalisation factor to be the sum of all intensities in the spectrum. We define the $\ell_1$ normalisation factor $\hat{N}^{(1)}$ as follows:

$$\hat{N}^{(1)} = \sum_i |x_i|. \tag{4.2.4}$$

Deininger et al. refer to this normalisation as Total Ion Count (TIC). It should be noted, however, that this interpretation of the $\ell_1$-norm might not be consistent for with the kind of spectra we are using here. In our case, a number of major pre-processing steps have already been taken (peak picking, for example) and this might change the interpretation of ion count. It is not explicitly clear what preprocessing steps Deininger et al. have taken prior to doing their normalisation. Having said that, I will be using this normalisation and henceforth referring to it as $\ell_1$-norm.

For $P = 2$, we consider the $\ell_2$-norm, which is a popular choice of normalisation factor. It is commonly known as the Euclidean norm, and is used often in statistics, for example in calculating mean squared error. We define the $\ell_2$-normalisation factor $\hat{N}^{(2)}$ as follows:

$$\hat{N}^{(2)} = \left( \sum_i x_i^2 \right)^{1/2}. \tag{4.2.5}$$

As we increase $P$, higher intensity signals will have more impact on the normalisation. We may take $P \to \infty$, and this will give us only the highest intensity signal as the normalisation factor. We call this the max norm or the $\ell_\infty$-norm. This means we would normalise so that the highest intensity in each spectrum is at the same level. The $\ell_\infty$ normalisation factor $\hat{N}^{(max)}$ is defined as follows:

$$\hat{N}^{(max)} = \max_i(x_i). \tag{4.2.6}$$

Deininger et al. (2011) also suggest normalising with the median of the intensities as a normalisation factor. We call this normalisation factor $\hat{N}^{(med)}$ and it is given as follows:

$$\hat{N}^{(med)} = \text{median}(x_i). \tag{4.2.7}$$

The median as a statistic has the property of being robust to the effects of outliers. Using the median as a normalisation factor should be robust to the effects of preprocessing.

## 4.3 Comparison of normalisation techniques

In this section I compare the four normalisation factors detailed above. A number of calibrants were applied to the sample before mass spectra were measured. Calibrants are a form of control, as we know that each calibrant is applied evenly over the sample. After a successful normalisation, we would expect to see roughly the same intensity for a calibrant's mass value in each spectrum. Hence we can judge the effectiveness of a normalisation by these criteria:

- reduces the variance of the intensities for a given calibrant,
- gives a more homogenous distribution of intensities for a given calibrant.

The four calibrants present in each of the IMS samples are listed in Table 4.3.1. For practical purposes, I included any $m/z$ values within a 0.2 Dalton radius of

Table 4.3.1: Internal Calibrants

| | Name | m/z |
|---|---|---|
| C1 | AngI | 1296.685 |
| C2 | Glu-fib | 1570.677 |
| C3 | DynA | 2147.199 |
| C4 | ACTH | 2932.588 |

these values.

### 4.3.1 Distribution of the calibrants before and after normalisation

Table 4.3.2 shows the mean and standard deviation of the ion intensities for each calibrant (listed 1 to 4) both before (raw data) and after normalisation with the four normalisation factors listed previously. Normalisation reduces the size of the intensities, scaling everything between 0 and 1 in the case of the $\ell_P$ norms. Consequently, the standard deviations are also reduced but they remain of the same order as the means. This suggests that our goal to reduce the variance of the calibrant intensities remains largely unfulfilled.

An illustration of how the distribution of ion intensities are affected can be seen in Figures 4.3.1 to 4.3.5. These images show the distribution of ion intensities separately for each calibrant in heat maps.

In the raw case, the first calibrant appears to have extremely high intensities near the top of the tissue, which quickly taper off. The normalisations have raised the peaks at other parts of the image, but none have produced a result that might be considered an even distribution of intensity.

The $\ell_\infty$ or maximum intensity normalisation appears to be particularly prone to problems, as there are many pixel positions (particularly on the edges where the background material is located) where the calibrant has the highest intensity in the

Table 4.3.2: The means and standard deviations of ion intensities for the four calibrants.

(a) Mean

| | C1 | C2 | C3 | C4 |
|---|---|---|---|---|
| Raw | 1578.224 | 1813.577 | 1536.269 | 827.172 |
| $\ell_1$ | 0.0264 | 0.0474 | 0.0527 | 0.0312 |
| $\ell_2$ | 0.1420 | 0.2412 | 0.2560 | 0.1549 |
| Max | 0.2699 | 0.4601 | 0.4866 | 0.2999 |
| Median | 4.9111 | 7.4821 | 7.6118 | 4.4343 |

(b) Standard Deviation

| | C1 | C2 | C3 | C4 |
|---|---|---|---|---|
| Raw | 2255.073 | 1706.340 | 1401.447 | 776.513 |
| $\ell_1$ | 0.0267 | 0.0383 | 0.0517 | 0.0364 |
| $\ell_2$ | 0.1149 | 0.1345 | 0.1778 | 0.1320 |
| Max | 0.2186 | 0.2534 | 0.3171 | 0.2489 |
| Median | 6.6613 | 6.6447 | 7.6404 | 5.0622 |

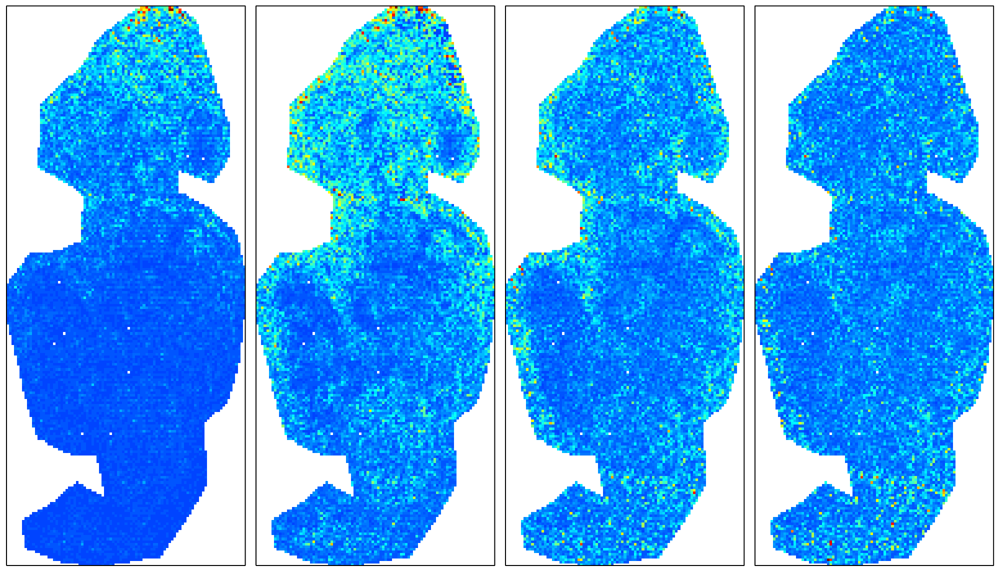

(a) C1 (b) C2 (c) C3 (d) C4

Figure 4.3.1: Distribution of the four calibrants on the raw sample from Patient 44.

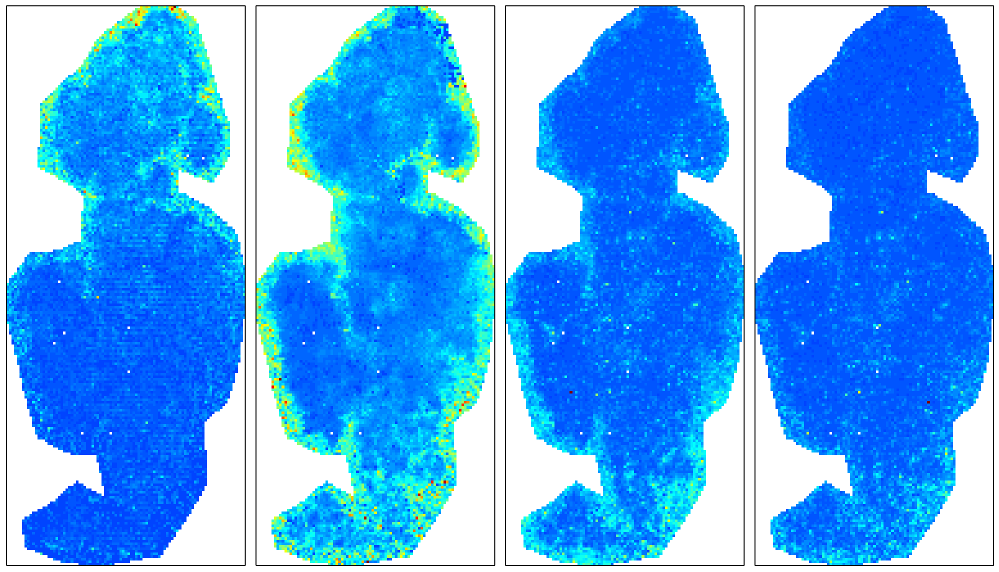

(a) C1 (b) C2 (c) C3 (d) C4

Figure 4.3.2: Distribution of the four calibrants after $\ell_1$ normalisation on sample from Patient 44.

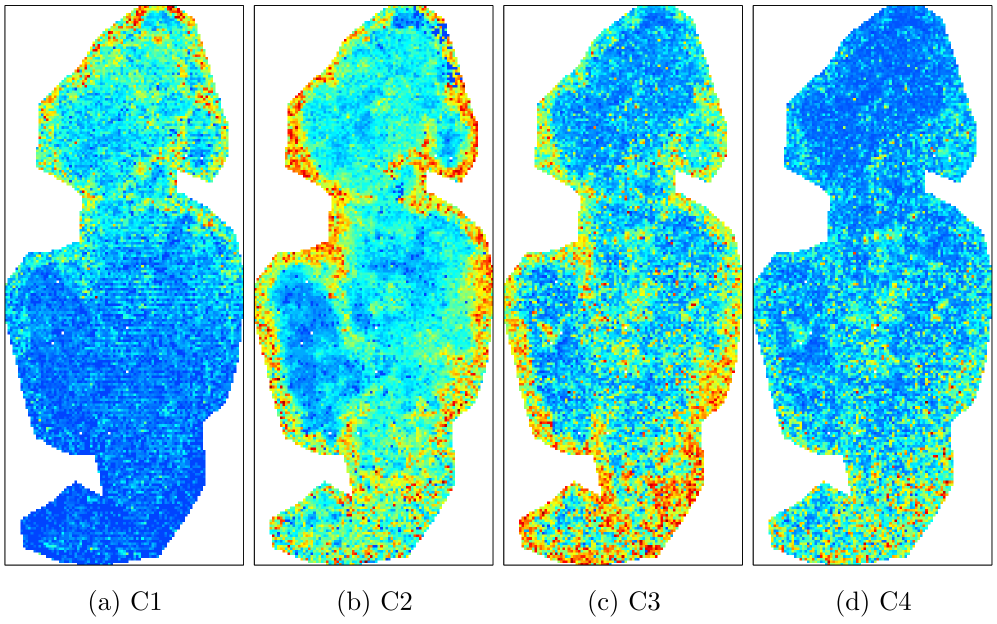
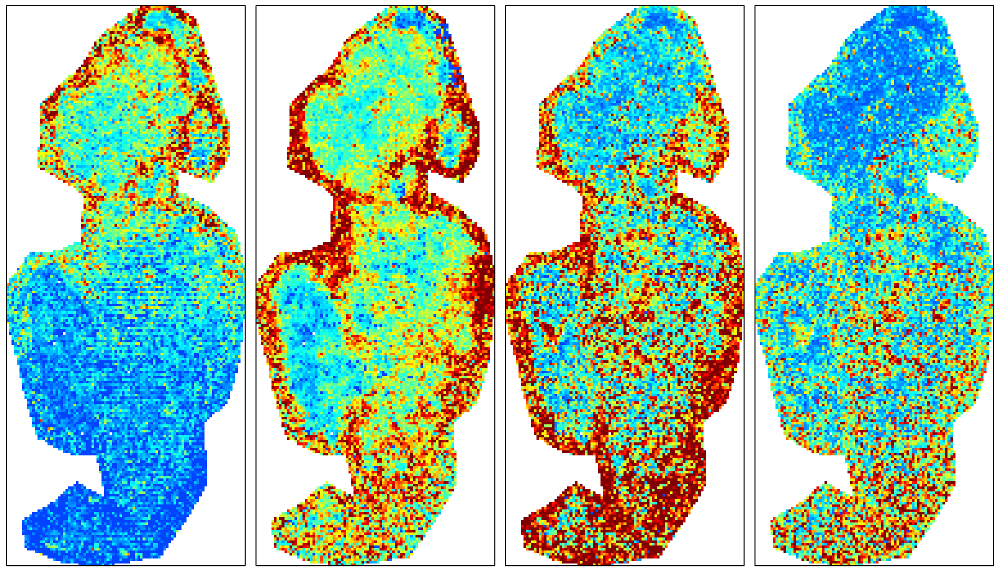


Figure 4.3.3: Distribution of the four calibrants after $\ell_2$ normalisation on sample from Patient 44.

spectrum and is hence normalised to 1. Hence we see high intensity regions clustered around the edges.

The $\ell_2$ normalisation also displays some of this problem of overweighting calibrants. At the edges (on the background material) it is likely that the calibrants are the only peaks present in the spectra (since they were added there artificially). As a result, these areas are a little skewed towards overweighting the calibrant intensities. Having said that, the $\ell_2$ normalisation may be the most successful of the methods I have used so far, as the heat map in Figure 4.3.3 shows less extreme values than the max norm in Figure 4.3.4. The heat maps for the $\ell_1$ (Figure 4.3.2) and median (Figure 4.3.5) norms are not significantly different from the raw case, and hence they can be regarded as ineffective methods of normalisation.

Since none of these normalisations have provided satisfactory results, we look to a different method of normalisation. Since we assume that the calibrants are applied



(a) C1 (b) C2 (c) C3 (d) C4

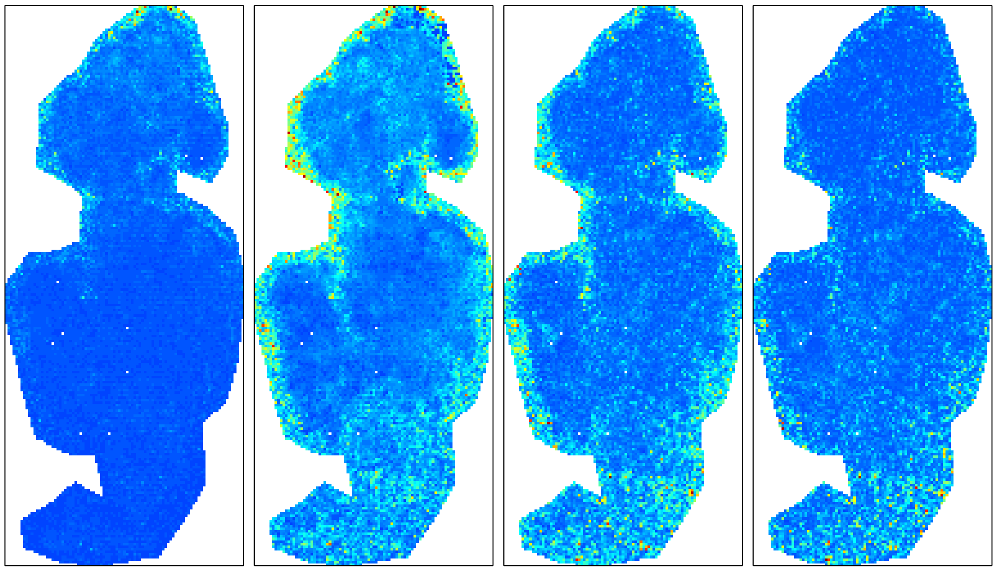
Figure 4.3.4: Distribution of the four calibrants after $\ell_\infty$ (maximum ion intensity) normalisation on sample from Patient 44.

with constant intensity across the sample, we can use them explicitly to determine an appropriate normalisation constant $N_j$ for each spectrum. We call this approach peak intensity correction (PIC).

## 4.4 Peak intensity correction

Gagnon-Bartsch and Speed (2012), in their discussion of the removal of unwanted variance (RUV), discuss the function of *negative controls*, which they define as variables that are expected not to change. Hence any apparent variability in these variables must be attributed to the unwanted variance that we are trying to mitigate. The internal calibrants listed in Table 4.3.1 are precisely the negative controls.

RUV also relies on positive controls, but it is not clear what the positive controls could be in our case. As RUV has been designed primarily for genes, it may not be directly applicable to work with proteins. This approach may be viewed as a special



(a) C1 (b) C2 (c) C3 (d) C4

Figure 4.3.5: Distribution of the four calibrants after median normalisation on sample from Patient 44.

case of RUV.

Recall our model for the mass spectra, after peak picking and baseline correction:

$$x_j^{[peak]}(t) = N_j S_j(t) \epsilon_j(t). \tag{4.4.1}$$

Taking the log of (4.4.1) gives us a simple linear equation:

$$\log x_j^{[peak]}(t) = \log N_j + \log S_j(t) + \epsilon_j(t). \tag{4.4.2}$$

As an aside: when taking logs it is important to be wary of zero values. If one were to normalise the interpolated/binned data then adding 1 to the entire spectrum is recommended. According to my definition, $x_k$ is a peak list and so no zero quantities are included.

Now take $t_m$ to be the $m/z$ value of the $m$th calibrant. If we consider only the $m/z$ values for the calibrants (that is, fixing $t = t_m$), the signal $S_j(t_m) = S_m$ only depends on $m$. Recall that calibrants should have constant intensity, so there is no dependence on the spectrum $j$. We now have:

$$\log x_{jm}^{[peak]} = \log N_j + \log S_m + \epsilon_{jm}. \tag{4.4.3}$$

This is now a regression problem where $\log N_j$ and $\log S_m$ are coefficients we can estimate from the data. The linear regression problem, 4.4.3, is of the form:

$$\mathbf{y} = \mathrm{X}\vec{\beta} + \vec{\epsilon}. \tag{4.4.4}$$

For the $j$th calibrant and $k$th spectrum, put

$$\begin{aligned} y_{jm} &= \log x_{jm}^{[peak]} \qquad \text{so} \\ y_{jm} &= \mu_m + \nu_j + \epsilon_{jm} \end{aligned} \tag{4.4.5}$$

where $\mu_j = \log S_j$ (calibrant dependent), $\nu_j = \log N_j$ (spectrum dependent) and $\epsilon_{jm}$ is random error. For $\vec{\beta} = [\mu_1, \dots, \mu_\rho, \nu_1, \dots, \nu_n]^T$, the design matrix X consists of $n+1$ blocks $B_\ell$, each $\rho \times (\rho + n)$, where $\rho$ is the number of calibrants:

$$\mathrm{X} = \begin{bmatrix} B_1 \\ \vdots \\ B_{n+1} \end{bmatrix} \quad \& \quad B_\ell = \begin{bmatrix} 1 & 0\dots & 0 & |\, 0\dots0 \,|\, 1 \,|\, 0 & 0\dots0 \\ 0 & 1\dots & 0 & |\, 0\dots0 \,|\, 1 \,|\, 0 & 0\dots0 \\ & \ddots & & |\, 0\dots0 \,|\, 1 \,|\, 0 & 0\dots0 \\ 0 & 0\dots & 1 & |\, 0\dots0 \,|\, 1 \,|\, 0 & 0\dots0 \end{bmatrix}.$$

The least squares estimator for $\vec{\beta}$ is given by:

$$\hat{\beta} = (X^T X)^{-1} X^T y_{jm}. \tag{4.4.6}$$

Hence finding $\hat{\beta}$ requires the calculation of $(X^T X)^{-1}$. This calculation is computationally too expensive to use routinely, since this matrix is very large and sparse. However, note that $X^T X$ has a rather simple form:

$$X^T X = \begin{bmatrix} (n+1) I_{\rho\times\rho} & \mathbf{1}_{\rho\times n} \\ \mathbf{1}_{n\times\rho} & \rho I_{n\times n} \end{bmatrix}. \tag{4.4.7}$$

Due to this specific form, an inverse can be found analytically by considering each section of the matrix separately and deriving the matrix adjoint. Consequently, we can determine an analytic solution for $\vec{\beta}$ (see Winderbaum (2015) for details). The solutions are as follows:

$$\log \hat{S}_m = \bar{x_m} = \frac{1}{n}\sum_{j=1}^{n} x_{jm}, \tag{4.4.8}$$

$$\log \hat{N}_j = \frac{1}{\rho}\sum_{m=1}^{\rho}(x_{jm} - \bar{x_m}). \tag{4.4.9}$$

We take $\exp(\log \hat{N}_j)$ as an estimate for $N_j$, and this becomes the normalisation factor by which we divide the $j$th spectrum. Explicitly:

$$\hat{N}^{(PIC)} = \exp(\log \hat{N}_j). \tag{4.4.10}$$

Then we normalise the $j$th spectrum like so:

$$x_j^{[norm]} = \frac{1}{\exp(\log \hat{N}_j)} x_j. \tag{4.4.11}$$

Alternatively, if we wish to keep the data on a log scale, we can subtract the term $\log \hat{N}_j$ from Equation (4.4.2).

$$\log(x_j)^{[norm]} = \log(x_j) - \log(\hat{N}_j). \tag{4.4.12}$$

The transformation in Equation 4.4.11 is henceforth referred to as PIC and the transformation in Equation 4.4.12 is referred to as log-PIC.

### 4.4.1 Distribution of the calibrants after PIC

Since the PIC normalisation is based on the calibrants, the calibrants act as a training set. Hence also using the calibrants as a testing set (as we have done up until now) would show biased results. In order to amend this, I have tested

Table 4.4.1: The mean and standard deviation of ion intensities for the four calibrants after PIC and PIC with log transformation.

(a) Mean

| | C1 | C2 | C3 | C4 |
|---|---|---|---|---|
| Raw | 1578.224 | 1813.577 | 1536.269 | 827.172 |
| PIC | 996.118 | 1270.828 | 1064.314 | 621.523 |
| log PIC | 6.3814 | 7.0018 | 6.9252 | 6.3268 |

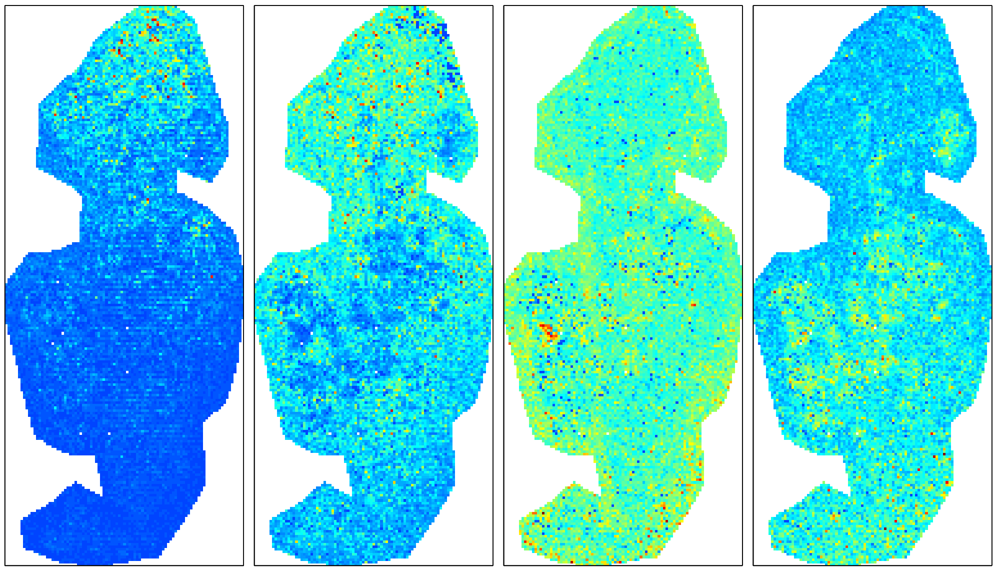
(b) Standard Deviation

| | C1 | C2 | C3 | C4 |
|---|---|---|---|---|
| Raw | 2255.073 | 1706.340 | 1401.447 | 776.513 |
| PIC | 1104.701 | 687.643 | 313.046 | 296.654 |
| log PIC | 1.0714 | 0.5606 | 0.3171 | 0.4640 |

one calibrant at a time, with PIC normalisation trained on the remaining three calibrants. This can be considered a leave-one-out cross validation.

Once again, the means and standard deviations before and after normalisation are listed in Table 4.4.1. PIC normalisation does not greatly reduce the mean intensities, but we do notice a greater decrease in the standard deviations. Taking the log completely changes the scale of the intensities, but we note that the standard deviations reduce significantly in proportion to the means of intensities. In other words, we have reduced the coefficient of variation. Reducing the coefficient of variation is a key aim, as a smaller standard deviation relative to the mean is a move toward the model that assumes constant intensities for calibrants.

The heat maps in Figures 4.4.1, 4.4.2 and 4.4.3 show the distribution of ion intensities after PIC normalisation. Note that the colours themselves are not illustrative; it is the differences in colour on a given heat map that show the evenness of distribution. The heat maps on the log data even out over a higher intensity range



(a) C1 (b) C2 (c) C3 (d) C4

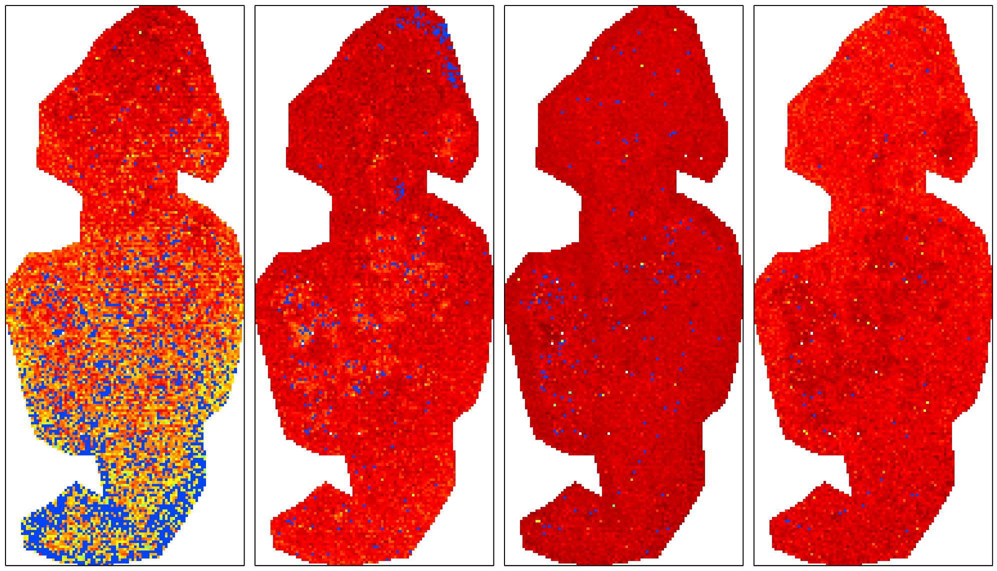
Figure 4.4.1: Distribution of the four calibrants after PIC on sample from Patient 44.

and this is why they appear more red.

The distribution of calibrant ion intensities has become more evenly spread after each transformation. Out of the four calibrants, the first calibrant appears the worst in each case - which could suggest an instrumentation error and it may be the case that this particular calibrant does not appear evenly across the tissue. From the images it would seem that the ion intensities are systematically higher at the top and fade out towards the bottom. This is not surprising, as it is in line with how the data are collected: they are rasterised left to right, top to bottom in rows and there is a time delay in which the calibrant may be affected.

## 4.5 Conclusion

In this chapter I have demonstrated why normalisation is a relevant problem. Most of the common, naive approaches to normalisation seem to have little effect, and

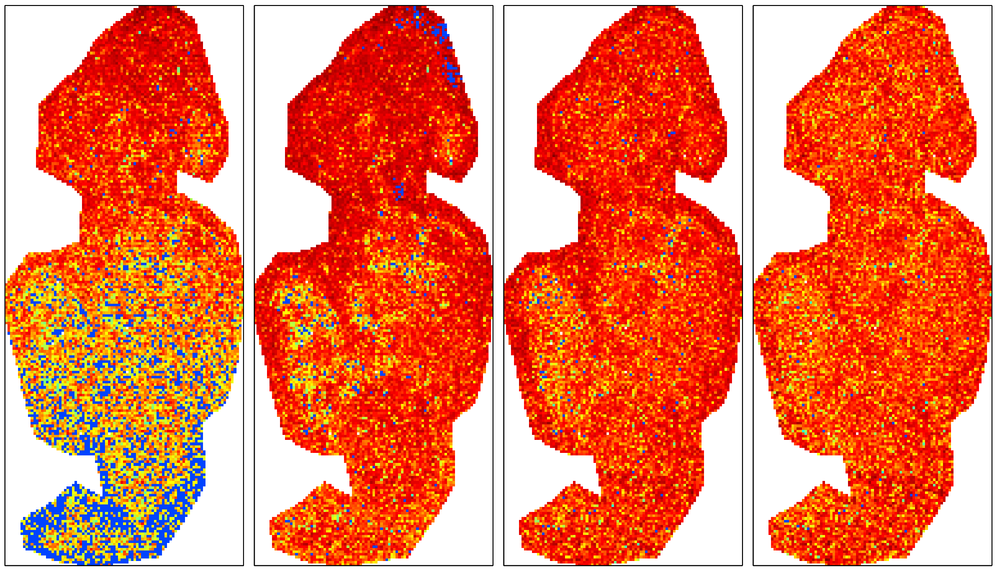


(a) C1 (b) C2 (c) C3 (d) C4

Figure 4.4.2: Distribution of the four calibrants after PIC in the log scale on sample from Patient 44.

some may even have a negative effect on the interpretability of the data. PIC has been introduced as a more intelligent approach that learns from information in the data. The tables and the images suggest that PIC is a more successful approach to normalisation, as it lowers the standard deviation of calibrant ion intensities relative to the mean, and hence creates a more constant spread. However, there are a number of assumptions inherent in PIC: including the assumption that the calibrant quantities are indeed homogeneously present within the tissue. The anomalous first calibrant suggests that this might not always be an accurate assumption, as it has likely been affected by systematic machine error.

We also note the effect of applying a log transformation to the ion intensities. A log transformation completely changes the scale, and extreme values become less extreme. The log transformed data show much less variance in the distribution of the calibrants, and therefore appplying a log transformation also has a similar effect



(a) C1 (b) C2 (c) C3 (d) C4

Figure 4.4.3: Distribution of the four calibrants after log transformation only on sample from Patient 44.

to normalisation.

# Chapter 5

# Clustering

This chapter is primarily concerned with clustering approaches for the proteomics IMS data described in previous chapters.

*Cluster analysis* is refers to an exploratory approach in its own right which partitions observations into groups according to which observations are *close* or *similar*. In clustering, information about class labels is not used, and typically does not exist.

The goal of cluster analysis is to develop a "picture" of the data. Clustering data may lead to further insight into the effect of errors, the presence of abnormalities and outliers, and may reveal interesting features. When we have data with many observations, we would like to be able to isolate or identify the most important and interesting parts, as this can simplify later analysis.

Cluster analysis could also be a first step in a more comprehensive analysis and carried out prior to classification. Classification (Koch, 2013, chapter 6) refers to supervised learning whereby class labels are used to derive a rule for determining which observations belong to each class. Classification is related to regression, in the sense that the regression response is used in the construction of the estimator and for prediction. For the data used in this thesis, labels do not exist, so classification is not a appropriate. This project focuses on clustering.

For the IMS data our preference would be to cluster the observations into groups that correspond to different tissue types, the most interesting of these being cancerous tissue. Such a cluster arrangement would be ideal, however we cannot guarantee that any given clustering method will do this. Clustering is only exploratory; it does not make use of explicit information about tissue type, but tries to find some pattern in the data.

Different clustering methods may produce different results, and they will vary depending on what parameters and controls they are given (such as number of clusters, choice of distance measure). Hence in order to obtain useful and robust clustering results, it will be worth spending some time investigating different possible configurations. We will make use of the imaging aspect of the IMS data to assess clustering results via visualisation, providing a rough, qualitative guide as to whether the clusters are reflecting tissue type. In this chapter I will firstly explore some issues with the choice of clustering method. I will cluster both the raw peak list data, and a binary transformation of the peak list data for comparison. Later I will apply a variety of other possible transformations to the data in an attempt to improve results. This includes the normalisation methods discussed in the previous chapter.

## 5.1 Clustering methods

There are a variety of methods that may be used for clustering. Two popular methods are *hierarchical clustering* and *k-means* clustering (Koch, 2013, chapter 6).

### 5.1.1 Hierarchical clustering

Hierarchical clustering has not been used extensively in this project, so my outline will be brief.

There are typically two approaches to hierarchical clustering:

- *Agglomerative clustering* begins by treating every observation as its own cluster, and at each step joins clusters together so that larger clusters are formed.

- *Divisive clustering* begins with the data in a single cluster and subsequently divides it into smaller clusters.

In each case, the method is stopped whenever the user decides that an appropriate number of clusters has been reached. This means that the user does not necessarily need to know the number of clusters beforehand, but will still make a decision at some point as to how many clusters there should be. The decision could be based on a predetermined number, $k$, or the user could impose a bound on the size of a cluster (Koch, 2013, section 6.2).

The hierarchical clustering methods depends on two notions of closeness: the first being the distance between observations, and the second being the proximity of sets of observations, called the *linkage*.

A disadvantage of hierarchical clustering is the computation time. The algorithm works with the distance matrix between all observations in the sample which can be very large. For example in our proteomics data we usually have more than 10,000 observations in a given sample.

One advantage of hierarchical clustering is that we do not need to know the number of clusters beforehand, but this may not be an issue if we already have an intuitive idea as to how many clusters we would like to see. For our application, we would like our clusters to correspond to the four classes we expect to see (three tissue types and background) so there is some justification for choosing four clusters from the outset. Hierarchical clustering is sometimes used in proteomics when the user wants to manually check the clusters (see Deininger et al., 2008; Bruand et al., 2011; Bonnel et al., 2011; Trede et al., 2012).

### 5.1.2 $k$-means clustering

Another common approach to clustering is $k$-means clustering, which is a partition style approach. The number of clusters, called $k$, is decided from the outset, and from there it becomes a matter of deciding their *optimal* formation. Optimality in this case means minimising the $k$-means objective function, called $J_k$.

Let $\mathcal{W} = \{C_1, C_2, \dots, C_k\}$ be a cluster arrangement. We call $\mathcal{W}^*$ the optimal cluster arrangement if:

$$\mathcal{W}^* = \underset{\mathcal{W}}{\operatorname{argmin}}\, J_k = \underset{\mathcal{W}}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} d(\mathbf{x}, \bar{\mathbf{x}}_i)^2, \tag{5.1.1}$$

where $k$ is the number of clusters, chosen beforehand, $C_i$ is the set of observations in cluster $i$, $\bar{\mathbf{x}}_i$ is the centre of cluster $i$ and $d$ is a distance measure.

Computationally, determining $\mathcal{W}^*$ has been shown to be a NP-hard problem, even for the simplest case $k = 2$ (Drineas et al., 2004). Consequently, algorithms for finding $\mathcal{W}^*$ are heuristic and often converge to a local, non-global, minimum.

The standard $k$-means algorithm, sometimes referred to as Lloyd's algorithm, consists of an assignment step and an update step:

- **Initialisation:** Randomly or otherwise, assign initial cluster centroids.
- **Assignment Step:** Assign observations to the nearest centroid.
- **Update Step:** Make the means of the current clusters to be the new cluster centroids.

The assignment and update steps are repeated until the solution stabilises.

A bad choice of initial cluster centroids can lead to the algorithm converging on a local minimum. There has been some investigation into choosing good initial centroids, also known as *seeds*, notably by Arthur and Vassilvitskii (2007), who propose `k-means++`, an algorithm that improves the accuracy of $k$-means with careful seeding. Another approach for seeding, using principal component analysis (PCA) will be discussed in more detail in Chapter 6.

For the remainder of this chapter I will be using $k$-means to cluster the proteomics data. Before doing so, I will make some further comments about the limitations of $k$-means and how to choose the parameters for the algorithm.

## 5.2 Comments on the parameters of $k$-means

In order to cluster according to $k$-means, the user must specify three parameters (Jain, 2010); the number of clusters $k$, the cluster initialisation, and a *distance* measure.

Choosing the number of clusters $k$ can be a difficult problem, since the ideal number of clusters may be an ambiguous concept, especially when little information about the data is known beforehand. On the other hand, there are instances in which the user may have some intuitive understanding of how many clusters they expect to see. In the case of the proteomics data we expect to see four clusters corresponding to three tissue types and background space, so it seems reasonable to choose $k = 4$. It might also be of interest to try $k = 3$ or $k = 5$, but by clustering with $k$ more than 5 we begin to lose the desired interpretability.

Choosing good initialisation is important, since as I mentioned earlier a bad initial input could lead to a non-optimal solution. Although the `k-means++` algorithm helps to decide good starting points, simply replicating the $k$-means algorithm (up to 10 times) and choosing the result with the smallest $J_k$ can help ensure an optimal solution.

The choice of distance measure is perhaps the most important of the parameters, since it is the distance measure that defines precisely what we mean by two observations being 'close' or 'similar'. Different distance measures can give very different clustering results.

Firstly, it is necessary to be precise about what I mean by distance. The definition of distance given by Koch (2013, section 5.3) is as follows:

**Definition 5.2.1.** *For $i = 1, 2, \ldots,$ let $\mathbf{x}_i$ be $p$-dimensional vectors. A* distance *$d$ is a map defined on pairs of random vectors $\mathbf{x}_i, \mathbf{x}_j$ such that $d(\mathbf{x}_i, \mathbf{x}_j)$ is a positive random variable which satisfies:*

1. $d(\mathbf{x}_i, \mathbf{x}_j) = d(\mathbf{x}_j, \mathbf{x}_i) \geq 0$ *for all $i, j$,*

*2.* $d(\mathbf{x}_i, \mathbf{x}_j) = 0$ *if* $i = j$*, and*

*3.* $d(\mathbf{x}_i, \mathbf{x}_j) \leq d(\mathbf{x}_i, \mathbf{x}_k) + d(\mathbf{x}_k, \mathbf{x}_j)$ *for all* $i, j$ *and* $k$*.*

*Let* $\mathbb{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ *be p-dimensional data. We call* $d$ *a distance for* $\mathbb{X}$ *if* $d$ *is defined for all pairs of random vectors belonging to* $\mathbb{X}$*.*

*A distance* $d$ *is called a* metric *if 2 is replaced by* $d(\mathbf{x}_i, \mathbf{x}_j) = 0$ *if and only if* $i = j$*.*

We can think of distance as a weak form of metric. Often the distances used are also metrics, for example the Euclidean distance, but this definition also allows us to use measures of similarity that are not metrics, for example the cosine distance.

The most commonly used distance is the Euclidean distance. It has the nice properties of being a metric and a norm and it is also an intuitive way of thinking about distance, particularly in low dimensions.

**Definition 5.2.2.** *The* Euclidean distance $d_E$ *is defined for random vectors* $\mathbf{x}_1$ *and* $\mathbf{x}_2$ *by:*

$$d_E(\mathbf{x}_1, \mathbf{x}_2) = [(\mathbf{x}_1 - \mathbf{x}_2)^T (\mathbf{x}_1 - \mathbf{x}_2)]^{1/2}. \tag{5.2.1}$$

The Euclidean distance lends itself towards finding clusters that are spherical in shape. When clusters take other shapes, the Euclidean distance may become unreliable. Take for example the simulated data in Figure (5.2.1). The natural clusters appear to be the two rectangular/linear clusters, but $k$-means with the Euclidean distance is not able to detect these shapes.

The Euclidean distance may also become too large too quickly in very high dimensions. Observations in high dimensions become isolated in the Euclidean sense, and this is clear from the mathematics: the higher the dimension of the vector, the more components to be summed and so distance becomes increasingly large. This can mean that errors could be exacerbated in high dimensions.

Despite the Euclidean distance being ubiquitous in most instances of $k$-means, it is wise to consider other ways of thinking about proximity of vectors in high

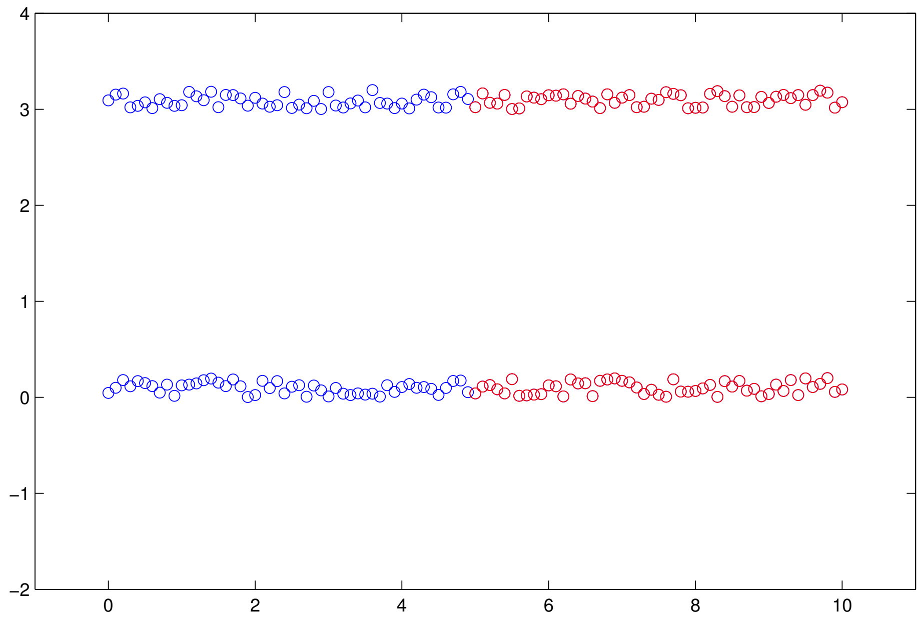


Figure 5.2.1: Observations are clustered into a red cluster and a blue cluster. This cluster arrangement, optimal according to $k$-means with Euclidean distance, does not detect the obvious cluster structure.

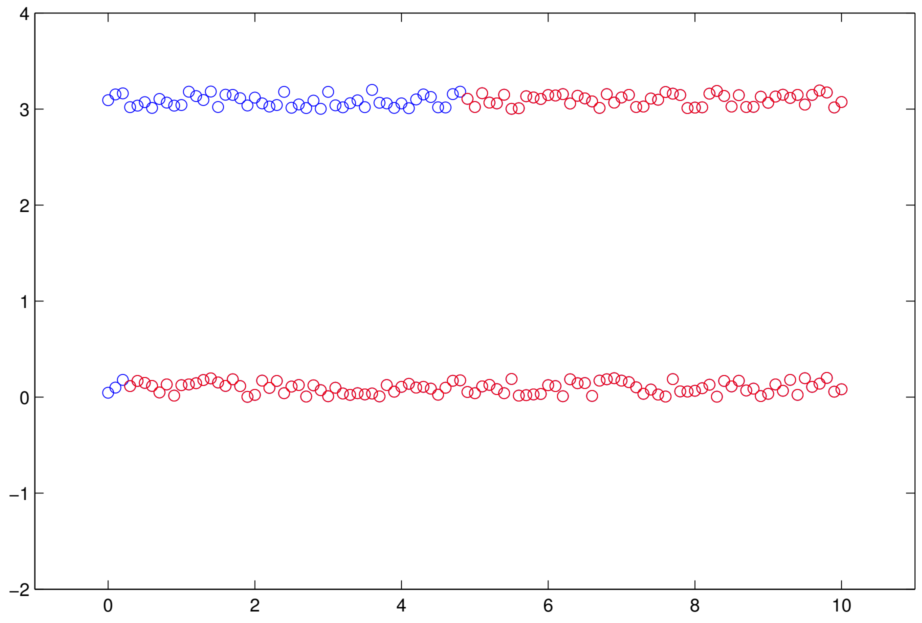


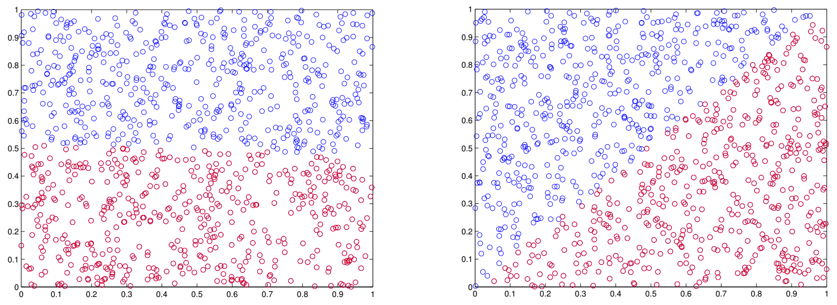
Figure 5.2.2: $k$-means with cosine distance also fails to detect the obvious structure.

dimensions. In particular, we consider the cosine distance, which measures the closeness of vectors by the angle between them.

**Definition 5.2.3.** *The* cosine distance $d_C$ *is defined for random vectors* $\mathbf{x}_1$ *and* $\mathbf{x}_2$ *by:*

$$d_C(\mathbf{x}_1, \mathbf{x}_2) = 1 - \cos(\mathbf{x}_1, \mathbf{x}_2) = 1 - \frac{\mathbf{x}_1^T \mathbf{x}_2}{\sqrt{(\mathbf{x}_1^T \mathbf{x}_1)(\mathbf{x}_2^T \mathbf{x}_2)}}. \tag{5.2.2}$$

Clustering with the cosine distance lends itself to clusters which are more conical in shape. When we apply $k$-means with the cosine distance to our simulated data, as in Figure 5.2.2 we see a slightly different cluster arrangement. Figure 5.2.3 shows clustering on the uniform random square and this also helps to illustrate the difference in shapes of clusters determined by the method with different distances. Although there are no obvious clusters, the Euclidean distance finds clusters that are spherical, resulting in two rectangular shapes, and the cosine distance finds clusters that are conical, resulting in two triangular shapes.



(a) $k$-means with Euclidean distance. (b) $k$-means with cosine distance.

Figure 5.2.3: Clustering results for uniform random square.

The best choice of distance will likely depend on context. The cosine distance will take no account of the differences in lengths of vectors, since it only measures angles. In some cases information about a vector's length may be important.

## 5.3 Clustering raw peak-list data

In this section I am going to be using $k$-means to cluster the proteomics IMS data, using both the Euclidean and cosine distances. We consider data from three tissue samples, belonging to three distinct patients. For the purposes of the study, they are called Patient 44, Patient 173 and Patient 540. I will be mainly considering Patient 44, with the other two shown for comparison.

The data are vectors, having been interpolated into bins (as described in Chapter 3). To begin with I will cluster the *raw peak-list data* which is data that have not been normalised or transformed. I will be using both Euclidean and cosine distance.

The resulting clusters for Patient 44 are shown in Figure 5.3.2. Each pixel in the image corresponds to an observation and each pixel has been coloured in accordance with the cluster membership of the observation at that pixel. There are four clusters in each (blue, orange, pink and purple). On the left clustering with Euclidean distance is shown and on the right clustering with cosine distance. These

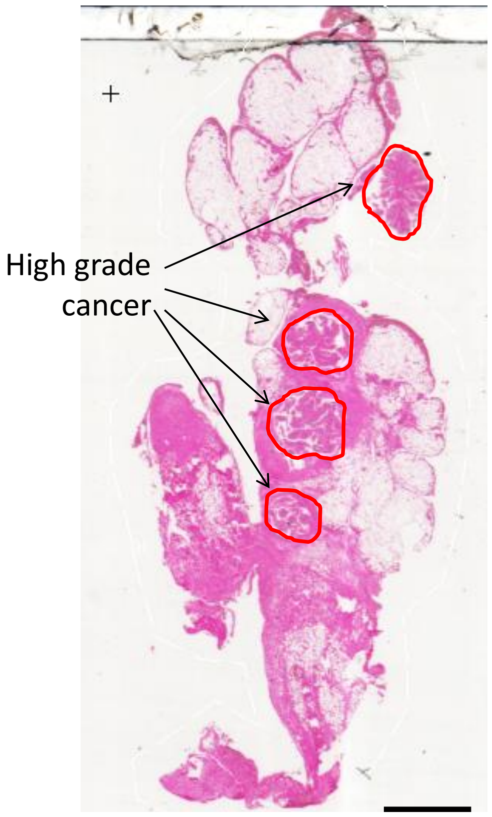


Figure 5.3.1: Dyed image of the tissue sample from Patient 44, indicating the cancer regions.

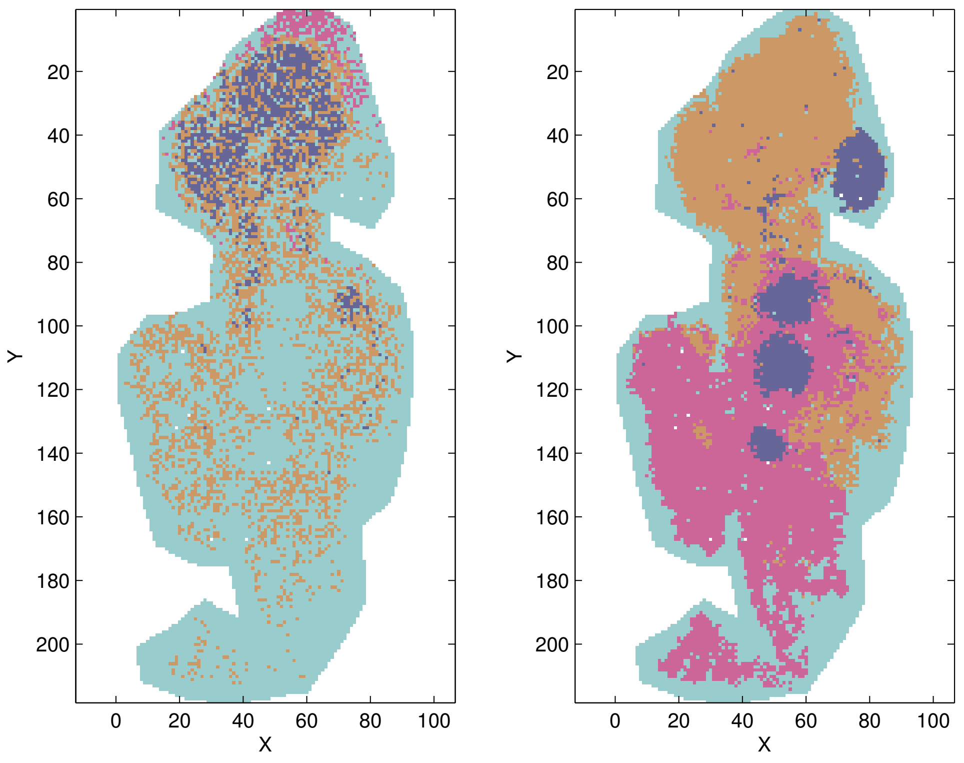


Figure 5.3.2: $k$-means clustering results for Patient 44 with Euclidean distance *(left)* and cosine distance *(right)*.

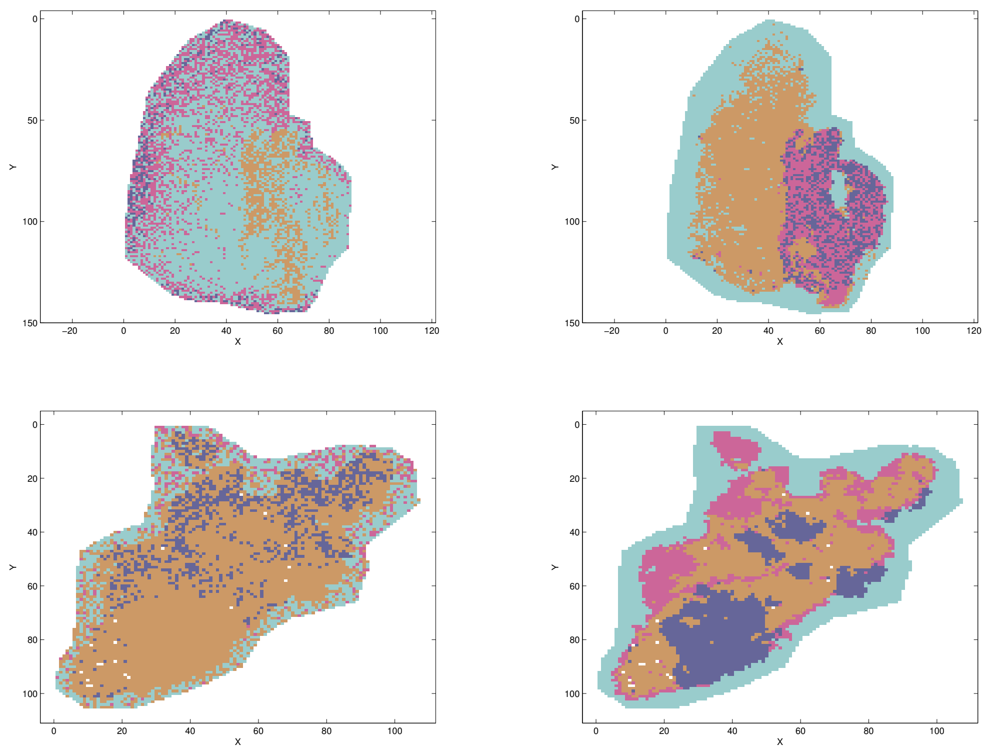


Figure 5.3.3: $k$-means clustering results with $K = 4$ for Patient 173 *(top)* and Patient 540 *(bottom)* with Euclidean distance *(left)* and cosine distance *(right)*.

images may be visually compared to the dyed tissue image in Figure 5.3.1. On this image the pathologist has indicated the high grade cancer regions, circled in red.

On inspection of Figure 5.3.2 one notices that the purple cluster in the cosine image seems to match up with the cancer regions. Furthermore, the orange cluster matches approximately with the fatty tissue (adipose) and the pink cluster with the connective tissue (stroma). The blue cluster matches with the background (no tissue). From this inspection we might decide that the $k$-means clustering with cosine distance is successful in the sense that it approximately identifies the tissue types.

The cluster allocation given by $k$-means with Euclidean distance (on the left hand side) is completely different to the cosine image and does not appear to match the different tissue types. This is not surprising since, as I have previously illustrated in Chapter 4, the raw peak list data are affected by artefacts which make the intensity peaks between different spectra difficult to compare. The Euclidean distance tends to be sensitive to these errors. Conversely, the errors have no effect on the cosine distance, since the cosine distance implicitly normalises the vectors.

What this seems to suggest is that $k$-means with Euclidean distance may be improved if the data is normalised or transformed beforehand. Since the Euclidean distance is more popular and commonplace than the cosine distance, I aim to produce a result that is similar for both distance measures.

The raw peak list data for patients 173 and 540 were also clustered with both Euclidean and cosine distance. Figure 5.3.3 shows that we see a similar outcome for these datasets also.

## 5.4 Clustering after a binary transformation

Leaving normalisation aside for the moment, a straightforward way of removing artefacts in the data is to apply a binary transformation. This leaves empty bins with a value of 0 and all non-empty bins are assigned a value of 1. This essentially

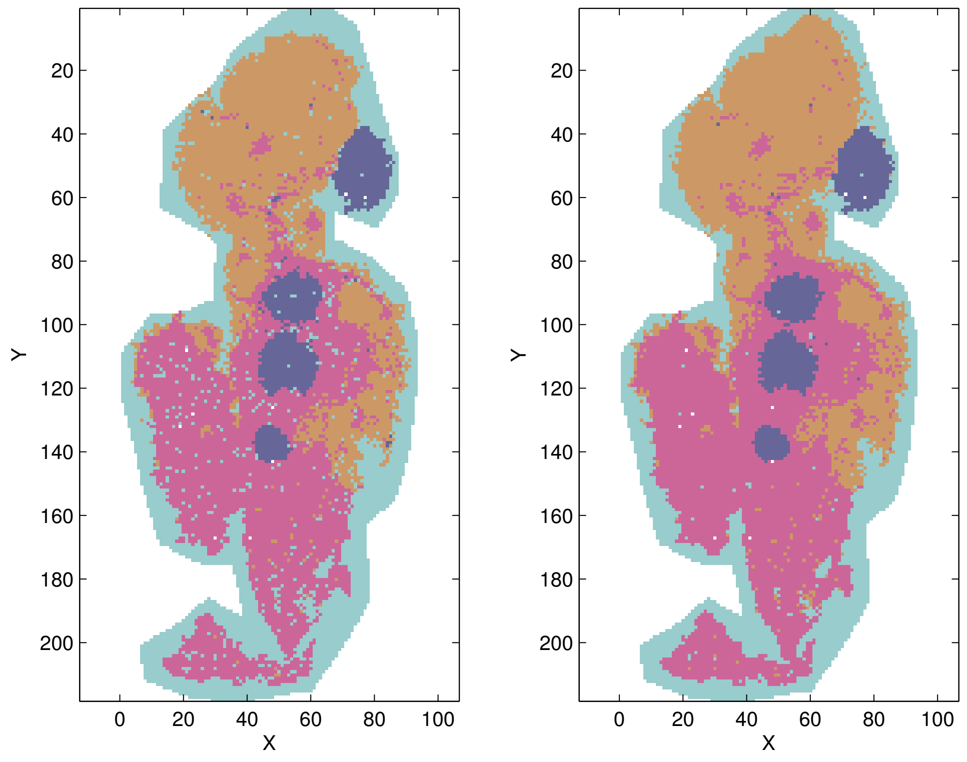


Figure 5.4.1: *k*-means clustering results for binary data with Euclidean distance *(left)* and cosine distance *(right)*.

strips the data of all information about intensity - we are left only with information as to whether or not a peak is present.

The clustering results in Figure 5.4.1 show greater parity between the arrangements for the two distance measures. This is understandable, since in using cosine distance we are disregarding information about the height of peaks, so by using a binary transformation with Euclidean distance we expect a similar result.

By inspection of the graphs, transforming the peak data to binary data and clustering the binary data appears to be a successful approach to clustering for both distance measures. However, the downside of a binary transformation is that it is not invertible, and hence we are losing information about intensity peaks. This is perhaps not a problem for clustering, but for other applications such as classification we may wish to keep all this information.

Hence a question we might ask is, is it possible to recreate similar clustering results without throwing away information about intensity? In the next part I will return to looking at some of the normalisation methods discussed in the previous chapter.

## 5.5 Clustering after normalisation

As in the previous chapter, the usual way to deal with artefacts in the data is to apply a normalisation. In this section I will apply two of the more successful normalisations: $\ell_2$ normalisation and peak intensity correction (PIC). We start with the raw peak data and normalise these data with using the $\ell_2$ and PIC normalisations, and then cluster the normalised data both with the Euclidean and cosine distance. The results can be seen in Figure 5.5.1 and 5.5.2.

As expected, these normalisations do not affect the clustering with cosine distance, as cosine distance is invariant to scaling of the observations. The clustering of $\ell_2$ normalised data with Euclidean distance shows some improvement from the raw data case, but still differs considerably from the image resulting from the cosine distance and from the H&E stained image. Similarly for the clustering of PIC normalised data.

## 5.6 Cluster after log transformation

Prior to doing PIC, we took the log of the peak list data. This was so we could turn our multiplicative model into a linear one and then do linear regression. After performing log-PIC, the data were transformed back and normalised in the usual way (by dividing each spectrum by a constant).

We now consider the possibility of taking the log of the data as a transformation

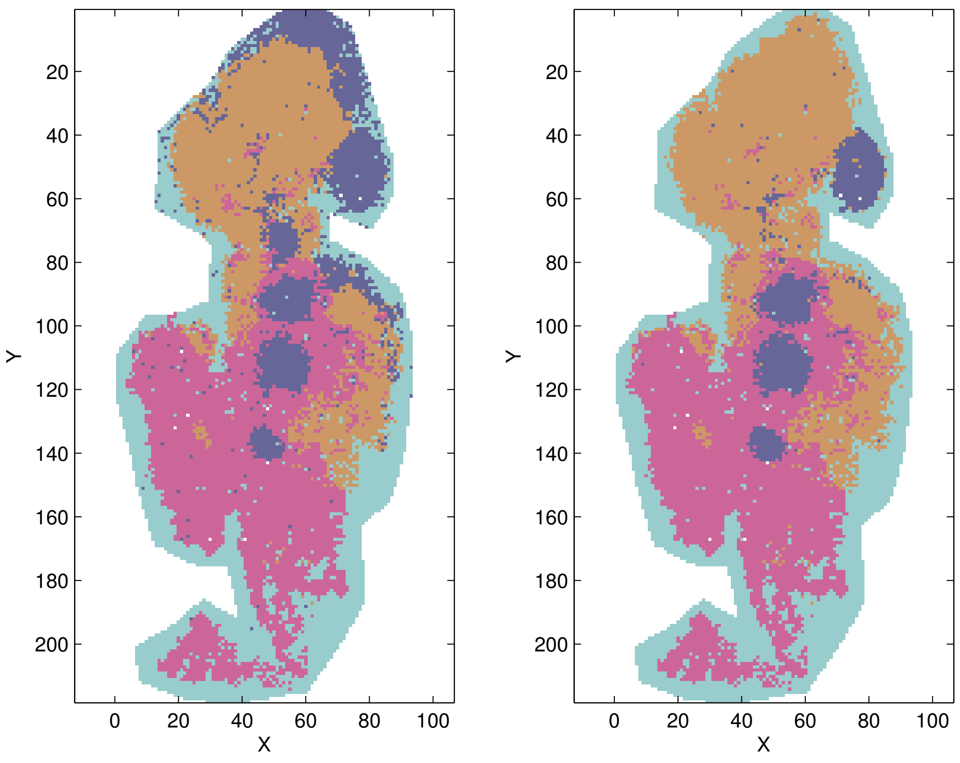


Figure 5.5.1: $k$-means clustering results for raw data after $\ell_2$ normalisation with Euclidean distance *(left)* and cosine distance *(right)*.

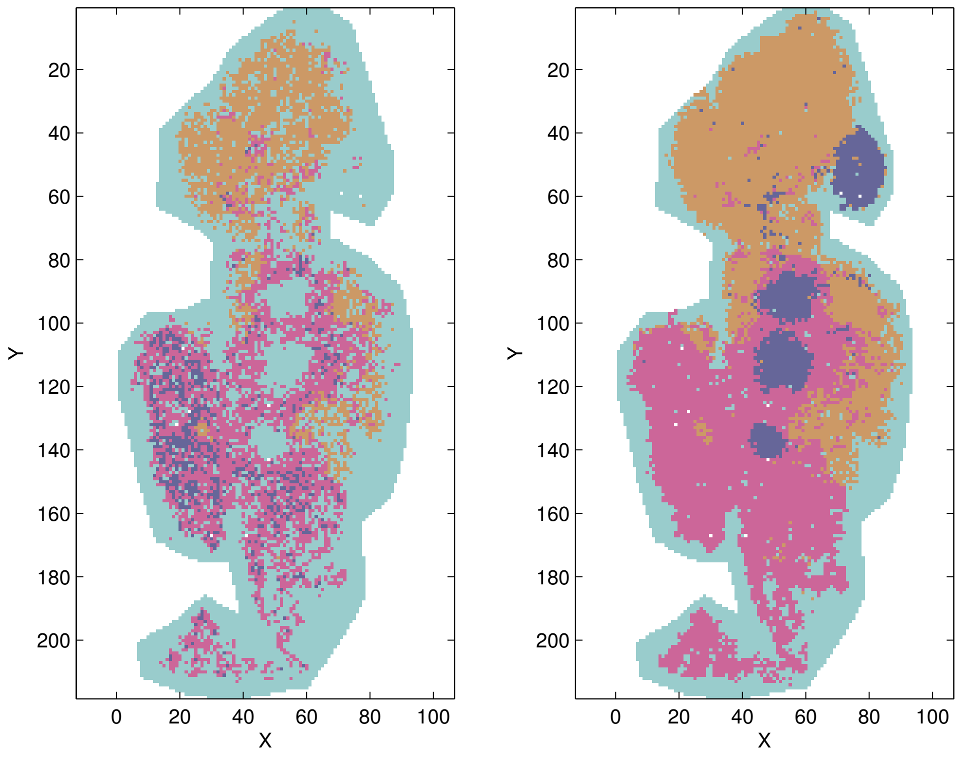


Figure 5.5.2: *k*-means clustering results for raw data after PIC normalisation with Euclidean distance *(left)* and cosine distance *(right)*.

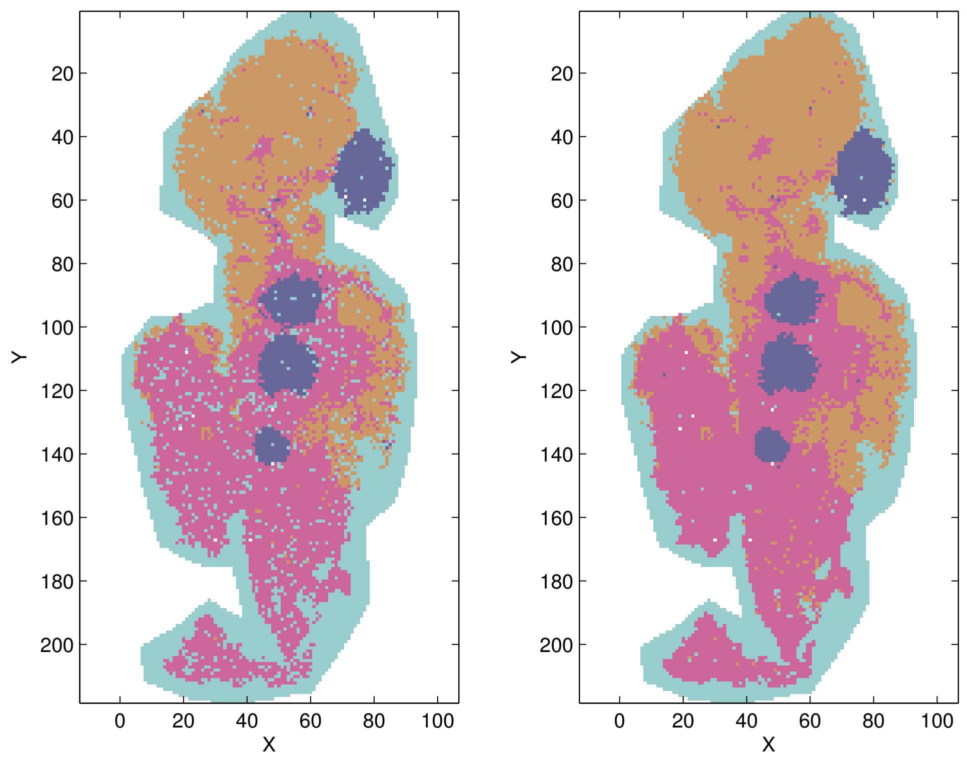


Figure 5.6.1: $k$-means clustering results for data after log transformation with Euclidean distance *(left)* and cosine distance *(right)*.

in itself. Taking logs can help to iron out discrepancies in intensities which are caused by artefacts, simply by virtue of the fact the log function makes the intensities smaller. Unlike the binary transformation, the log transformation is invertible and does not throw away any information about intensities.

The results for clustering log data, shown in Figure 5.6.1 are promising. The cluster arrangements for clustering with Euclidean and cosine distance are now similar, albeit the Euclidean image has more holes and less smooth edges.

I have also clustered the log data with the PIC included. The result, shown in Figure 5.6.2, does not differ greatly from just using the log transformation.

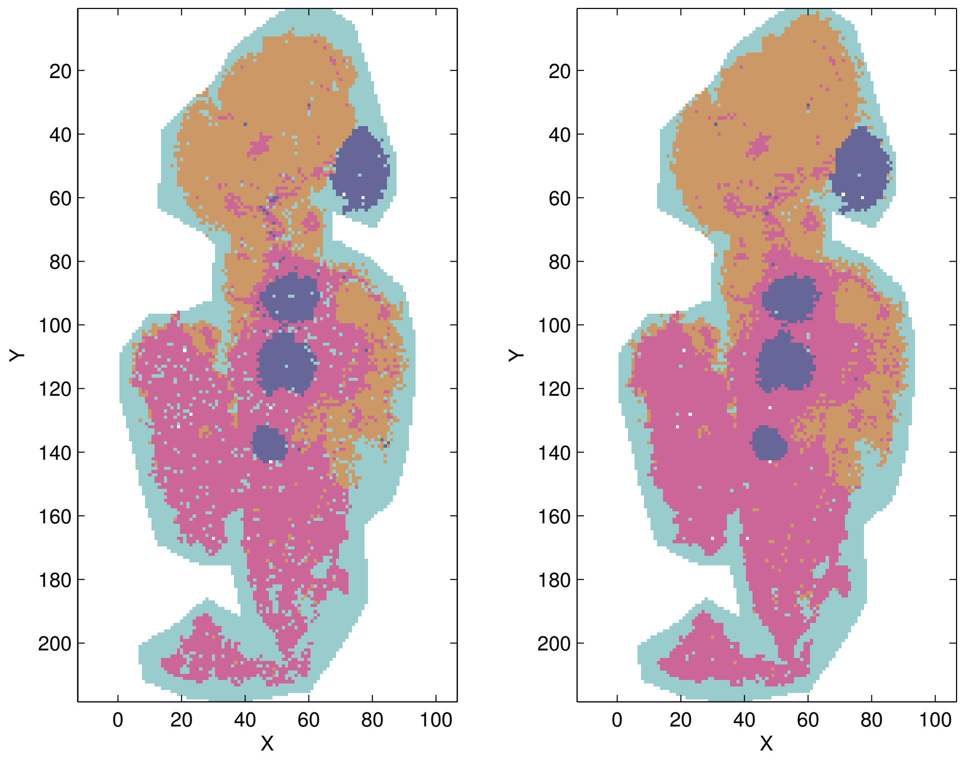


Figure 5.6.2: $k$-means clustering results for data after log transformation and PIC with Euclidean distance *(left)* and cosine distance *(right)*.

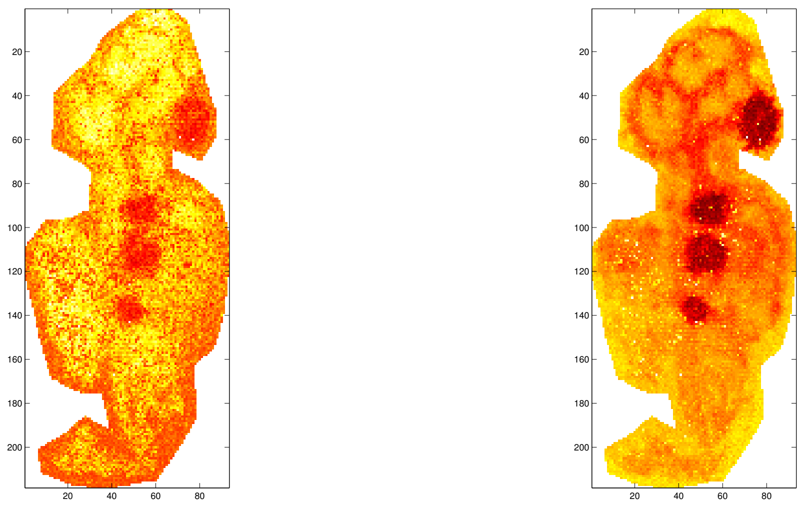
## 5.7 Visualisation of closeness to the cancer cluster

There is a binary aspect to cluster allocation: an observation can either be in a cluster or not. Sometimes if an observation lies half way between two cluster centroids its allocation is not as meaningful as the allocation of observations which are very close to a particular centroid.

The different cluster assignments I described in this chapter have been based on the intensity information of the observations only, as is standard in $k$-means clustering. Visualising the cluster membership by colour as we have done here necessarily results in hard boundaries for each cluster and may lead to the wrong impression that spatial neighbours near boundaries of clusters differ considerably, and hence belong to different tissue types. In reality we expect a gradual change from one type of tissue to another.

To gain some insight into such gradual change we consider a post-clustering step which compares spectra at arbitrary grid points to those of cluster centroids. As the cancer cluster is of most interest, we only focus on this cluster, but the approach could be applied to other cluster centroids. In this step I fix the cluster centroid for the cancer cluster (which may not be associated with a pixel or observation) and using the same distance measure as in the clustering step, so here Euclidean or cosine, for each pixel and its corresponding observation I calculate the distance between the observation and the cancer centroid. The distance values obtained for each pixel are then shown in a heat map.

Figure 5.7.1 shows the heat maps of the post-clustering step using the Euclidean distance on the left and the cosine distance on the right with the cancer cluster as the cluster of interest for the log-transformed data. The darker colour represents closeness to the cancer cluster centroid. A comparison with Figure 5.6.1 shows that in each case the cancer regions in Figure 5.6.1 agree well with the darker 'distance-based' regions in Figure 5.6.1. The images also indicate closeness between the cancer tissue and the stroma tissue, which is not surprising, given that cancer typically



(a) Euclidean clustering (b) Cosine clustering

Figure 5.7.1: The distance from the cancer cluster, Euclidean distance *(left)* and cosine distance *(right)* on the Euclidean and cosine cluster results on the log-transformed data. The darker colour represents closeness.

grows out of the stroma.

In conclusion, clustering followed by the distance-based post-clustering step reveals interesting structure in the tissue that is not apparent when the data are clustered into a discrete number of clusters. Other visualisation of the tissue structure is worth exploring, as well as more detailed analyses of the structure that is exhibited in the panels of Figure 5.7.1.

## 5.8 Conclusion

In this section I have clustered the IMS data after a number of transformations and normalisations. My judgement as to whether a particular cluster arrangement is good or not so good is based entirely on whether the cluster arrangement appears to sort observations in accordance with their tissue types, which is judged subjectively based on the dyed tissue image provided by the pathologist.

Since I have used $k$-means with two different distance measures it is of interest to compare the differences and similarities between the results generated by each. Using the cosine distance consistently provides good results. The Euclidean distance is highly sensitive to which normalisation or transformation is applied to the data. An improvement in clustering with Euclidean distance may also give insight into which transformation or normalisation is most effective and useful.

The clustering in this chapter was based entirely on $k$-means. In the next chapter I explore an extension of $k$-means based on principal component analysis (PCA). In the final chapter I will return to the clustering results obtained here and propose a quantitative comparison.

# Chapter 6

# Principal Component Analysis and Clustering

So far I have discussed how clustering data can be beneficial when the data are large and high-dimensional. In some ways we can think of clustering as a form of dimension reduction. A more common dimension reduction method is *principal component analysis* (PCA) (Koch, 2013; Jolliffe, 2002). The goal of PCA is to create linear combinations of variables that maximise variance. We can also think of PCA as projecting data into a lower-dimensional space in such a way that most important information is retained.

Ding and He (2004) draw a relationship between PCA and $k$-means clustering. They claim that the principal components are the continuous solutions to what they refer to as the $k$-means objective function. Furthermore, PCA can be used to assist $k$-means clustering and may alleviate the problem of the $k$-means algorithm failing to converge to the optimal cluster solution.

In this chapter I will be investigating the claims of Ding and He; this includes a corrected and revised proof of theorems given by Ding and He, and a number of studies of the application both on simulated data and the IMS data. Since the bulk of this chapter revolves around the theorems in this particular paper, I have borrowed much of the authors' notation, occasionally making amendments for clarity.

## 6.1 Principal component analysis

The aim of principle component analysis (PCA) is to represent the original data in a lower-dimensional space, by creating new variables as combinations of the original variables. These combinations are chosen such that the maximum variance in the data is retained. Specifically, the principal components are taken from the eigenvectors of the sample covariance matrix, corresponding to the largest eigenvalues.

We refer to the original $(p \times n)$ data matrix as $\mathbb{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$. Now consider the centred data matrix $\mathbb{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$, where $\mathbf{y}_i = \mathbf{x}_i - \bar{\mathbf{x}}$, $\bar{\mathbf{x}} = \sum_i \mathbf{x}_i / n$. The sample covariance matrix is then given by $S = \frac{1}{n-1}\mathbb{Y}\mathbb{Y}^T$. Let $r$ be the rank of $S$, and assume that all eigenvectors are unit vectors.

**Definition 6.1.1.** *The unit vectors $\mathbf{u}_i$, $i = 1, \ldots, r$, such that $\mathbb{Y}\mathbb{Y}^T\mathbf{u}_i = \lambda_i \mathbf{u}_i$ are the* principal directions*.*

Note that $\mathbb{Y}\mathbb{Y}^T$ is the sample covariance matrix without the scale factor of $1/(n-1)$. Since eigenvectors are invariant to scale, unit vectors $\mathbf{u}_i$ are also the eigenvectors of the sample covariance matrix $S$ of the original data matrix $\mathbb{X}$. The principal directions are also commonly referred to as *loadings* or *weight* vectors (Koch, 2013, chapter 2).

**Definition 6.1.2.** *The vectors $\mathbf{v}_i$, $i = 1, \ldots, r$, such that $\mathbb{Y}^T\mathbb{Y}\mathbf{v}_i = \lambda_i \mathbf{v}_i$ are the* unit principal components*.*

Note that Ding and He refer to the $\mathbf{v}_1$ in Definition 6.1.2 as *principal components* but it should be noted that the $\mathbf{v}_i$ are unit vectors. *Principal Component Scores* are obtained when the data are projected in the principal direction. We will see shortly that the principal component *scores* are given by $\lambda^{1/2}\mathbf{v}_i$.

From these definitions we see that the principal directions and unit principal component are closely related to the *singular value decomposition (SVD)* of the centred data matrix $\mathbb{Y}$.

**Definition 6.1.3.** *For a $p \times n$ matrix with rank $r$, $\mathbb{Y}$, we write $\mathbb{Y} = U\Lambda V^T$, where $U$ and $V$ are orthonormal matrices with dimensions $p \times r$ and $n \times r$ respectively, and $\Lambda$ is a diagonal matrix with dimension $r \times r$. This factorisation is called the* Singular Value Decomposition (SVD).

**Result 6.1.1.** *Using the notation of Definition 6.1.3, $U$ is the matrix of eigenvectors of $\mathbb{Y}\mathbb{Y}^T$, $V$ is the matrix of eigenvectors of $\mathbb{Y}^T\mathbb{Y}$ and the diagonal entries of $\Lambda^2$ are the corresponding eigenvalues of $\mathbb{Y}\mathbb{Y}^T$ and $\mathbb{Y}^T\mathbb{Y}$.*

*Proof.* Starting with $\mathbb{Y} = U\Lambda^{1/2}V^T$ and multiplying both sides by $\mathbb{Y}^T$ gives:

$$\begin{aligned}\mathbb{Y}\mathbb{Y}^T &= (U\Lambda^{1/2}V^T)(U\Lambda^{1/2}V^T)^T \\ &= U\Lambda^{1/2}V^TV\Lambda^{1/2}U^T \\ &= U\Lambda^{1/2}\Lambda^{1/2}U^T \quad \text{since } V^TV = I \\ &= U\Lambda U^T.\end{aligned}$$

Similarly:

$$\begin{aligned}\mathbb{Y}^T\mathbb{Y} &= (U\Lambda^{1/2}V^T)^T(U\Lambda^{1/2}V^T) \\ &= V\Lambda^{1/2}U^TU\Lambda^{1/2}V^T \\ &= V\Lambda^{1/2}\Lambda^{1/2}V^T \\ &= V\Lambda V^T.\end{aligned}$$

□

It follows that the principal component scores are the projections of the centred data onto the principal directions, given by:

$$\mathbf{u}_i^T\mathbb{Y} = \mathbf{u}_i^TU\Lambda^{1/2}V^T = \lambda_i^{1/2}\mathbf{v}_i^T. \tag{6.1.1}$$

Equivalently,

$$\mathbf{v}_i = \mathbb{Y}^T \mathbf{u}_i / \lambda^{1/2}, \tag{6.1.2}$$

which is consistent with Equation (1) from Ding and He (2004).

## 6.2 The $k$-means objective function

Recall from Chapter 5:

The goal of the $k$-means algorithm is to choose cluster assignments such that the total within cluster variability, $J_k$, is minimised. We will refer to $J_k$ as the $k$-means objective function. Recall that

$$J_k = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} d(\mathbf{x}, \bar{\mathbf{x}}_i)^2, \tag{6.2.1}$$

where $k$ is the number of clusters, chosen beforehand, $C_i$ is the set of observations in cluster $i$, $\bar{\mathbf{x}}_i$ is the centre of cluster $i$ and $d$ is a distance measure. For this chapter we will take $d$ to be the *Euclidean* distance. Hence we can rewrite $J_k$ as:

$$J_k = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} (\mathbf{x} - \bar{\mathbf{x}}_i)^T (\mathbf{x} - \bar{\mathbf{x}}_i). \tag{6.2.2}$$

Ding and He also define a function for measuring the proximity between two clusters $C_k$ and $C_l$. Define $\delta : \mathcal{P}(\mathbb{X}) \to \mathbb{R}$ such that:

$$\delta(C_k, C_l) = \sum_{i \in C_k} \sum_{j \in C_l} (\mathbf{x}_i - \mathbf{x}_j)^T (\mathbf{x}_i - \mathbf{x}_j). \tag{6.2.3}$$

This is just the sum of squared Euclidean distances between all pairs of points in the two clusters.

## 6.3 Clustering when $k = 2$

For sections 6.3 and 6.4 we will consider the simplest case of $k = 2$, that is, dividing the data into two clusters. Ding and He generalise their results to $k > 2$ but I will primarily consider the case $k = 2$.

**Result 6.3.1.** *When $k = 2$, it can be shown that:*

$$J_2 = n\overline{\mathbf{y}^T\mathbf{y}} - \frac{1}{2}J_D \tag{6.3.1}$$

*where*

$$J_D = \frac{n_1 n_2}{n}\left[2\frac{\delta(C_1, C_2)}{n_1 n_2} - \frac{\delta(C_1, C_1)}{n_1^2} - \frac{\delta(C_2, C_2)}{n_2^2}\right] \tag{6.3.2}$$

*and $n_1$, $n_2$ are the number of elements in $C_1$ and $C_2$ respectively, such that $n_1 + n_2 = n$. $\overline{\mathbf{y}^T\mathbf{y}}$ refers to the mean of $\mathbf{y}_i^T\mathbf{y}_i$ where $\mathbf{y}_i = \mathbf{x}_i - \bar{\mathbf{x}}$.*

In order to show this result, I will also prove the following lemma:

**Lemma 6.3.1.**

$$2\frac{\delta(C_1, C_2)}{n_1 n_2} - \frac{\delta(C_1, C_1)}{n_1^2} - \frac{\delta(C_2, C_2)}{n_2^2} = 2(\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T(\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2). \tag{6.3.3}$$

Note that this expression does not appear in Ding and He (2004). The incorrect expression $\frac{\delta(C_1,C_2)}{n_1 n_2} = \frac{\delta(C_1,C_1)}{n_1^2} + \frac{\delta(C_2,C_2)}{n_2^2} + (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T(\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)$ was given, which I have corrected as Equation (6.3.3).

*Proof.* Consider first the term $\delta(C_1, C_1) = \sum_{i \in C_1}\sum_{j \in C_1}(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)$. Since we are considering elements from the same cluster, zero terms will appear when $i = j$. To illustrate, consider all the terms of the double sum as elements of a matrix:

$$\begin{matrix} \mathbf{x}_1 - \mathbf{x}_1 & \mathbf{x}_1 - \mathbf{x}_2 & \dots & \mathbf{x}_1 - \mathbf{x}_{n_1} \\ \mathbf{x}_2 - \mathbf{x}_1 & \mathbf{x}_2 - \mathbf{x}_2 & \dots & \mathbf{x}_2 - \mathbf{x}_{n_1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_n - \mathbf{x}_1 & \mathbf{x}_n - \mathbf{x}_2 & \dots & \mathbf{x}_{n_1} - \mathbf{x}_{n_1} \end{matrix}. \tag{6.3.4}$$

Note that the diagonal terms are all zero.

Expanding the brackets on the left hand side of the expression for $\delta(C_1, C_1)$ gives:

$$\sum_{i \in C_1}\sum_{j \in C_1}(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j) = \sum_{i \in C_1}\sum_{j \in C_1}\mathbf{x}_i^T\mathbf{x}_i - 2\mathbf{x}_i^T\mathbf{x}_j + \mathbf{x}_j^T\mathbf{x}_j. \tag{6.3.5}$$

When expanding the brackets it is important to remember that the zero terms must be excluded. When we exclude the zero term, each $\mathbf{x}_i^T\mathbf{x}_i$ term appears $n_1 - 1$ times.

Since $\mathbf{x}_i^T\mathbf{x}_i$ and $\mathbf{x}_j^T\mathbf{x}_j$ are the exactly the same, in total the $\mathbf{x}_i^T\mathbf{x}_i$ term appears $2(n_1-1)$ times.

Now let us consider the cross term. First consider only the $i$th row from the matrix in (6.3.4):

$$\begin{aligned}\sum_{j\neq i}\mathbf{x}_i^T\mathbf{x}_j &= \mathbf{x}_i^T\mathbf{x}_1+\ldots\mathbf{x}_i^T\mathbf{x}_{i-1}+\mathbf{x}_i^T\mathbf{x}_{i+1}\ldots\mathbf{x}_i^T\mathbf{x}_{n_1}\\ &= \mathbf{x}_i^T\left(\sum_{j\neq i}\mathbf{x}_j+\mathbf{x}_i-\mathbf{x}_i\right)\\ &= \mathbf{x}_i^T(n_1\bar{\mathbf{x}}_1-\mathbf{x}_i)\quad\{\text{where }\bar{\mathbf{x}}_1\text{ is the cluster mean.}\}\\ &= n_1\mathbf{x}_i^T\bar{\mathbf{x}}_1-\mathbf{x}_i^T\mathbf{x}_i.\end{aligned}$$

Now consider all rows $i=1,\ldots,n_1$:

$$\begin{aligned}\sum_i\sum_{j\neq i}\mathbf{x}_i^T\mathbf{x}_j &= \sum_i^{n_1}(n_1\mathbf{x}_i^T\bar{\mathbf{x}}_1-\mathbf{x}_i^T\mathbf{x}_i)\\ &= n_1^2\bar{\mathbf{x}}_1^T\bar{\mathbf{x}}_1-\sum_i^{n_1}\mathbf{x}_i^T\mathbf{x}_i.\end{aligned}$$

Since we have simplified both the square term and the cross term, we can substitute them back into Equation (6.3.5):

$$\begin{aligned}\delta(C_1,C_1) &= 2(n_1-1)\sum_i^{n_1}\mathbf{x}_i^T\mathbf{x}_i-2n_1^2\bar{\mathbf{x}}_1^T\bar{\mathbf{x}}_1+2\sum_i^{n_1}\mathbf{x}_i^T\mathbf{x}_i\\ &= 2n_1\sum_i^{n_1}\mathbf{x}_i^T\mathbf{x}_i-2\sum_i^{n_1}\mathbf{x}_i^T\mathbf{x}_i-2n_1^2\bar{\mathbf{x}}_1^T\bar{\mathbf{x}}_1+2\sum_i^{n_1}\mathbf{x}_i^T\mathbf{x}_i\\ &= 2n_1\sum_i^{n_1}\mathbf{x}_i^T\mathbf{x}_i-2n_1^2\bar{\mathbf{x}}_1^T\bar{\mathbf{x}}_1.\end{aligned}$$

The case for $\delta(C_2,C_2)$ is the same. We get:

$$\frac{\delta(C_1,C_1)}{n_1^2}=\frac{2}{n_1}\sum_i^{n_1}\mathbf{x}_i^T\mathbf{x}_i-2\bar{\mathbf{x}}_1^T\bar{\mathbf{x}}_1, \tag{6.3.6}$$

$$\frac{\delta(C_2,C_2)}{n_2^2}=\frac{2}{n_2}\sum_j^{n_2}\mathbf{x}_j^T\mathbf{x}_j-2\bar{\mathbf{x}}_2^T\bar{\mathbf{x}}_2. \tag{6.3.7}$$

Next we consider the term $\delta(C_1, C_2) = \sum_{i \in C_1} \sum_{j \in C_2} (\mathbf{x}_i - \mathbf{x}_j)^T (\mathbf{x}_i - \mathbf{x}_j)$:

$$\begin{aligned}
\delta(C_1, C_2) &= \sum_{i \in C_1} \sum_{j \in C_2} \mathbf{x}_i^T \mathbf{x}_i - 2\mathbf{x}_i^T \mathbf{x}_j + \mathbf{x}_j^T \mathbf{x}_j \\
&= n_2 \sum_{i=1}^{n_1} \mathbf{x}_i^T \mathbf{x}_i + n_1 \sum_{j=1}^{n_2} \mathbf{x}_j^T \mathbf{x}_j - 2 \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \mathbf{x}_i^T \mathbf{x}_j \\
&= n_2 \sum_{i=1}^{n_1} \mathbf{x}_i^T \mathbf{x}_i + n_1 \sum_{j=1}^{n_2} \mathbf{x}_j^T \mathbf{x}_j - 2 n_1 n_2 \bar{\mathbf{x}}_1^T \bar{\mathbf{x}}_2.
\end{aligned}$$

Therefore:

$$2\frac{\delta(C_1, C_2)}{n_1 n_2} = \frac{2}{n_1} \sum_{i=1}^{n_1} \mathbf{x}_i^T \mathbf{x}_i + \frac{2}{n_2} \sum_{j=1}^{n_2} \mathbf{x}_j^T \mathbf{x}_j - 4\bar{\mathbf{x}}_1^T \bar{\mathbf{x}}_2, \tag{6.3.8}$$

and finally, putting (6.3.6), (6.3.7) and (6.3.8) together gives:

$$\begin{aligned}
2\frac{\delta(C_1, C_2)}{n_1 n_2} - \frac{\delta(C_1, C_1)}{n_1^2} - \frac{\delta(C_2, C_2)}{n_2^2} &= -4\bar{\mathbf{x}}_1^T \bar{\mathbf{x}}_2 + 2\bar{\mathbf{x}}_1^T \bar{\mathbf{x}}_1 + 2\bar{\mathbf{x}}_2^T \bar{\mathbf{x}}_2 \\
&= 2(\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2).
\end{aligned}$$

□

We have shown the lemma, and hence $J_D$ may be simplified like so:

$$J_D = 2\frac{n_1 n_2}{n} (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2). \tag{6.3.9}$$

It remains to show Result 6.3.1, that is:

$$J_2 = n\overline{\mathbf{y}^T \mathbf{y}} - \frac{1}{2} J_D = n\overline{\mathbf{y}^T \mathbf{y}} - \frac{n_1 n_2}{n} (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2). \tag{6.3.10}$$

*Proof. (Result 6.3.1)* Recall from Equation (6.2.2) the definition of $J_k$ with $k = 2$:

$$J_2 = \sum_{C_1} (\mathbf{x}_i - \bar{\mathbf{x}}_1)^T (\mathbf{x}_i - \bar{\mathbf{x}}_1) + \sum_{C_2} (\mathbf{x}_j - \bar{\mathbf{x}}_2)^T (\mathbf{x}_j - \bar{\mathbf{x}}_2).$$

After expanding and collecting terms this becomes:

$$\begin{aligned} J_2 &= \sum_{i=1}^{n} \mathbf{x}_i^T \mathbf{x}_i - n_1 \bar{\mathbf{x}}_1^T \bar{\mathbf{x}}_1 - n_2 \bar{\mathbf{x}}_2^T \bar{\mathbf{x}}_2 \\ &= \sum_{i=1}^{n} \mathbf{x}_i^T \mathbf{x}_i - (n_1 \bar{\mathbf{x}}_1^T \bar{\mathbf{x}}_1 + n_2 \bar{\mathbf{x}}_2^T \bar{\mathbf{x}}_2) \frac{n_1 + n_2}{n} \quad \{n_1 + n_2 = n\} \\ &= \sum_{i=1}^{n} \mathbf{x}_i^T \mathbf{x}_i - \frac{n_1^2 \bar{\mathbf{x}}_1^T \bar{\mathbf{x}}_1 + n_1 n_2 \bar{\mathbf{x}}_1^T \bar{\mathbf{x}}_1 + n_1 n_2 \bar{\mathbf{x}}_2^T \bar{\mathbf{x}}_2 + n_2^2 \bar{\mathbf{x}}_2^T \bar{\mathbf{x}}_2}{n}. \end{aligned}$$

We add and subtract terms with the intention of isolating a term that looks like $J_D$:

$$\begin{aligned} J_2 &= \sum_{i=1}^{n} \mathbf{x}_i^T \mathbf{x}_i - \frac{2n_1 n_2}{n} \bar{\mathbf{x}}_1^T \bar{\mathbf{x}}_2 - \frac{n_1^2}{n} \bar{\mathbf{x}}_1^T \bar{\mathbf{x}}_1 - \frac{n_2^2}{n} \bar{\mathbf{x}}_2^T \bar{\mathbf{x}}_2 - \left[ \frac{n_1 n_2}{n} (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2) \right] \\ &= \sum_{i=1}^{n} \mathbf{x}_i^T \mathbf{x}_i - n(\frac{n_1}{n} \bar{\mathbf{x}}_1 + \frac{n_2}{n} \bar{\mathbf{x}}_2)^T (\frac{n_1}{n} \bar{\mathbf{x}}_1 + \frac{n_2}{n} \bar{\mathbf{x}}_2) - \left[ \frac{n_1 n_2}{n} (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2) \right] \\ &= \sum_{i=1}^{n} \mathbf{x}_i^T \mathbf{x}_i - n \bar{\mathbf{x}}^T \bar{\mathbf{x}} - \left[ \frac{n_1 n_2}{n} (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2) \right] \\ &= \sum_{i=1}^{n} (\mathbf{x}_i - \bar{\mathbf{x}})^T (\mathbf{x}_i - \bar{\mathbf{x}}) - \left[ \frac{n_1 n_2}{n} (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2) \right] \\ &= n \overline{\mathbf{y}^T \mathbf{y}} - \frac{1}{2} J_D. \end{aligned}$$

□

Since $\overline{\mathbf{y}^T \mathbf{y}}$ is a constant, the problem of minimising $J_2$ becomes the problem of maximising $J_D$. Since $J_D$ as in Equation (6.3.2) is composed of a positive between-cluster distance and negative within-cluster distance, maximisation of $J_D$ can be interpreted as ensuring clusters are as distinct and tight as possible. In addition to this intuitive guide, Ding and He have also constructed $J_D$ to simplify the proof of their main theorem, which is that $J_D$ leads to a solution via the first principal component.

## 6.4 Theorem

One way of defining a particular cluster arrangement $\mathcal{W}$ is with a cluster indicator vector $\mathbf{q}$, defined in 6.4.1. The $i$th element of the vector takes some discrete value depending on which cluster the $i$th observation falls into. For the rest of this section, we will assume that $k = 2$.

**Definition 6.4.1.** *For $k = 2$, the ith component of* cluster indicator vector $\mathbf{q}$ *is defined like so:*

$$q(i) = \begin{cases} a & \textit{if } \mathbf{x}_i \in C_1, \\ b & \textit{if } \mathbf{x}_i \in C_2 \end{cases} \tag{6.4.1}$$

*where $a, b \in \mathbb{R}$ and $a \neq b$.*

Since the cluster indicator vector, $\mathbf{q}$, defines a particular cluster assignment $\mathcal{W}$, we can think of the $k$-means objective function $J_D$ in Equation 6.3.2 as a function of $\mathbf{q}$. The elements of a cluster indicator vector will only take one of two distinct values (since this is the only way a cluster indicator makes sense in practice). However, Ding and He argue that if we allow the cluster indicator vector to take *any* real value, then we can think about the maximisation of $J_D$ as an eigenvalue problem. It then follows that the $\mathbf{q}$ that maximises $J_D$ is the first principal component of $\mathbb{X}$. This idea works provided that we have a sensible way to move from a continuous vector back to a binary cluster indicator vector, which, as I will later discuss, may be something arbitrary, such as labelling positive and negative values. I will discuss both the proof and the implications of this result in some detail, but first let us consider the full theorem.

The following theorem is based on the theorem given by Ding and He as Theorem 2.2 in their paper. A major divergence from Ding and He is the removal of Equation 6.4.5, which is discussed in section 6.4.1.

**Theorem 6.4.1.** *Let $D$ be the matrix of squared Euclidean distances between observations and let $\hat{D}$ be the centred distance matrix. The following hold.*

- $\hat{D} = -2\mathbb{Y}^T\mathbb{Y}$
- *The eigenvector* $\mathbf{q}$ *of* $D$ *which corresponds to the largest negative eigenvalue of* $D$ *is a multiple of the first principal component direction* $\mathbf{v}_1$ *of* $\mathbb{X}$.

A sketch of the proof of Theorem 6.4.1 is given by Ding and He. I will give a more detailed walkthrough.

*Proof.* We start by constructing a *particular* cluster indicator vector $\mathbf{q}_0$, where $a = \sqrt{n_2/nn_1}$ and $b = -\sqrt{n_1/nn_2}$. That is:

$$q_0(i) = \begin{cases} \sqrt{n_2/nn_1} & \text{if } \mathbf{x}_i \in C_1, \\ -\sqrt{n_1/nn_2} & \text{if } \mathbf{x}_i \in C_2. \end{cases} \tag{6.4.2}$$

Consider a matrix of squared Euclidean distances between observations, i.e. $D = (d_{ij})$ where $d_{ij} = (\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)$. Then it can be shown that $\mathbf{q}_0$ relates to the $k$-means objective function $J_D$, namely $\mathbf{q}_0^T D\mathbf{q}_0 = -J_D$:

$$\begin{aligned}
\mathbf{q}_0^T D\mathbf{q}_0 &= \sum_{j\in C_1}\sum_{i\in C_1} \frac{n_2}{nn_1}(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j) + \sum_{j\in C_1}\sum_{i\in C_2} -\frac{1}{n}(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j) \\
&+ \sum_{j\in C_2}\sum_{i\in C_1} -\frac{1}{n}(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j) + \sum_{j\in C_2}\sum_{i\in C_2} \frac{n_1}{nn_2}(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j) \\
&= \frac{n_2}{nn_1}\delta(C_1, C_1) - \frac{1}{n}\delta(C_1, C_2) - \frac{1}{n}\delta(C_2, C_1) + \frac{n_1}{nn_2}\delta(C_2, C_2) \\
&\text{(by the definition in Equation (6.2.3))} \\
&= \frac{1}{n}\left[-2\delta(C_1, C_2) + \frac{n_2}{n_1}\delta(C_1, C_1) + \frac{n_1}{n_2}\delta(C_2, C_2)\right] \\
&= \frac{n_1 n_2}{n}\left[-2\frac{\delta(C_1, C_2)}{n_1 n_2} + \frac{\delta(C_1, C_1)}{n_1^2} + \frac{\delta(C_2, C_2)}{n_2^2}\right] \\
&= -J_D.
\end{aligned}$$

Next we allow our generalised cluster indicator vector $\mathbf{q}$ to be any eigenvector of $D$, hence relaxing the form given in Definition 6.4.1.

The objective of $k$-means is to maximise $J_D$, or equivalently minimise $-J_D$, so in this sense we think of $J_D$ as a function, not a constant. If we let $J$ be a function of $\mathbf{q}$, where $\mathbf{q}$ is any unit vector, then the $\mathbf{q}$ that minimises $J(\mathbf{q}) = \mathbf{q}^T D\mathbf{q}$ will be the eigenvector corresponding to the lowest (largest negative) eigenvalue of the equation $D\mathbf{z} = \lambda\mathbf{z}$.

Consider a centred distance matrix called $\hat{D} = \hat{d}_{ij}$, in which the entries of $D$ have the row and column means subtracted (with the repeated term added back in). That is,

$$\hat{d}_{ij} = d_{ij} - \sum_j d_{ij}/n - \sum_i d_{ij}/n + \sum_{ij} d_{ij}/n^2. \tag{6.4.3}$$

If we write $\hat{d}_{ij} = d_{ij} - \tilde{d}_{ij}$ and $\hat{D} = D - \tilde{D}$ then it follows that $\mathbf{q}_0^T \tilde{D}\mathbf{q}_0 = 0$ and hence $\mathbf{q}_0^T \hat{D}\mathbf{q}_0 = \mathbf{q}_0^T D\mathbf{q}_0 = -J_D$.

Therefore, $\mathbf{q}$ vector that maximises $J(\mathbf{q}) = \mathbf{q}^T D\mathbf{q}$ is given by the eigenvector corresponding to the lowest (largest negative) eigenvalue of

$$\hat{D}\mathbf{z} = \lambda\mathbf{z}. \tag{6.4.4}$$

With some algebra, it can be shown that $\sum_j d_{ij} = n\mathbf{x}_i^T\mathbf{x}_i + n\overline{\mathbf{x}^T\mathbf{x}} - 2n\mathbf{x}_i^T\bar{\mathbf{x}}$ and $\sum_{ij} d_{ij} = 2n^2\overline{\mathbf{y}^T\mathbf{y}}$.

$$\begin{aligned}
\sum_j^n d_{ij} &= \sum_j^n (\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j) \\
&= \sum_j^n \left(\mathbf{x}_i^T\mathbf{x}_i - 2\mathbf{x}_i^T\mathbf{x}_j + \mathbf{x}_j^T\mathbf{x}_j\right) \\
&= n\mathbf{x}_i^T\mathbf{x}_i + n\overline{\mathbf{x}^T\mathbf{x}} - 2n\mathbf{x}_i^T\bar{\mathbf{x}} \\
\sum_{ij}^n d_{ij} &= \sum_i^n \sum_j^n (\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j) \\
&= 2n\sum_i^n \mathbf{x}_i^T\mathbf{x}_i - 2n^2\bar{\mathbf{x}}^T\bar{\mathbf{x}} \\
&= 2n^2\overline{\mathbf{y}^T\mathbf{y}}.
\end{aligned}$$

By substituting these into Equation (6.4.3), we find that:

$$\begin{aligned}
\hat{d}_{ij} &= \mathbf{x}_i^T\mathbf{x}_i - 2\mathbf{x}_i^T\mathbf{x}_j + \mathbf{x}_j^T\mathbf{x}_j \\
&\quad - \left(n\mathbf{x}_i^T\mathbf{x}_i + n\overline{\mathbf{x}^T\mathbf{x}} - 2n\mathbf{x}_i^T\bar{\mathbf{x}}\right)/n \\
&\quad - \left(n\mathbf{x}_j^T\mathbf{x}_j + n\overline{\mathbf{x}^T\mathbf{x}} - 2n\mathbf{x}_j^T\bar{\mathbf{x}}\right)/n \\
&\quad + \left(2n\sum_i^n \mathbf{x}_i^T\mathbf{x}_i - 2n^2\bar{\mathbf{x}}^T\bar{\mathbf{x}}\right)/n^2 \\
&= -2\left(\mathbf{x}_i^T\mathbf{x}_j - \mathbf{x}_i^T\bar{\mathbf{x}} - \mathbf{x}_j^T\bar{\mathbf{x}} + \bar{\mathbf{x}}^T\bar{\mathbf{x}}\right) \\
&= -2(\mathbf{x}_i - \bar{\mathbf{x}})^T(\mathbf{x}_j - \bar{\mathbf{x}})
\end{aligned}$$

Equivalently $\hat{D} = -2\mathbb{Y}^T\mathbb{Y}$. It follows that the eigenvector corresponding to the lowest eigenvalue of Equation (6.4.4) is also the eigenvector corresponding to the largest eigenvalue of $\mathbb{Y}^T\mathbb{Y}$ which is precisely the unit principal component vector $\mathbf{v}_1$.

□

### 6.4.1 Discussion

If allowing the cluster indicator vector to be continuous gives us the principal component as an optimal solution, then the obvious question to ask is what does this mean for clustering? If a cluster indicator vector takes continuous values then it loses its functionality for allocating observations into clusters. In order to perform clustering we must somehow discretise the unit principal component vector.

As part of their Theorem, Ding and He define cluster allocation like so:

$$C_1 = \{\mathbf{x}_i | \mathbf{v}_1(i) \leq 0\}, \quad C_2 = \{\mathbf{x}_i | \mathbf{v}_1(i) > 0\}. \tag{6.4.5}$$

In other words, the observations are sorted into clusters based on the sign of the unit principal component vector. This rule is intuitive, since the principal component projection centres the data so we might expect two clusters to fall either side of zero. However, in practice this is not always the case, as I will illustrate in a small simulation study.

In the first case, two spherical clusters with $n = 1000$ observations are normally distributed with covariance matrices $\Sigma_1 = 0.2I$ and $\Sigma_2 = 0.9I$ respectively. In the second case, the sizes of the clusters have been changed; the first cluster now only has $n = 200$.

Table 6.4.1: Cluster simulation parameters.

| | $n_1$ | $n_2$ | $\Sigma_1$ | $\Sigma_2$ |
|---|---|---|---|---|
| Sim 1 | 1000 | 1000 | $\begin{pmatrix} 0.2 & 0 \\ 0 & 0.2 \end{pmatrix}$ | $\begin{pmatrix} 0.9 & 0 \\ 0 & 0.9 \end{pmatrix}$ |
| Sim 2 | 200 | 1000 | $\begin{pmatrix} 0.2 & 0 \\ 0 & 0.2 \end{pmatrix}$ | $\begin{pmatrix} 0.9 & 0 \\ 0 & 0.9 \end{pmatrix}$ |

These simulated data are clustered according to Equation (6.4.5). In Figure 6.4.1 the colours red and blue represent the cluster allocation. By construction, the two clusters are fairly distinct and obvious.

When the size of the clusters is equal, the sign of $\mathbf{v}_1$ seems to cluster accurately. However, when one cluster is significantly larger than the other, we get a cluster allocation that is no longer intuitive - observations that we expect to be in the big red cluster have been allocated to the blue cluster. This is because $\mathbf{v}_1$ is skewed by the different weights of the clusters, and hence its sign is no longer able to divide them.

This example suggests that even though $\mathbf{v}_1$ may be an optimal solution in a *continuous* setting, clustering is not a continuous problem, and hence it might not be so helpful in practice.

## 6.5 Application for $k = 2$ and $k > 2$

Having seen some examples of PCA clustering on simulated data, let us now see what happens when we try to cluster real data. In the following examples I will

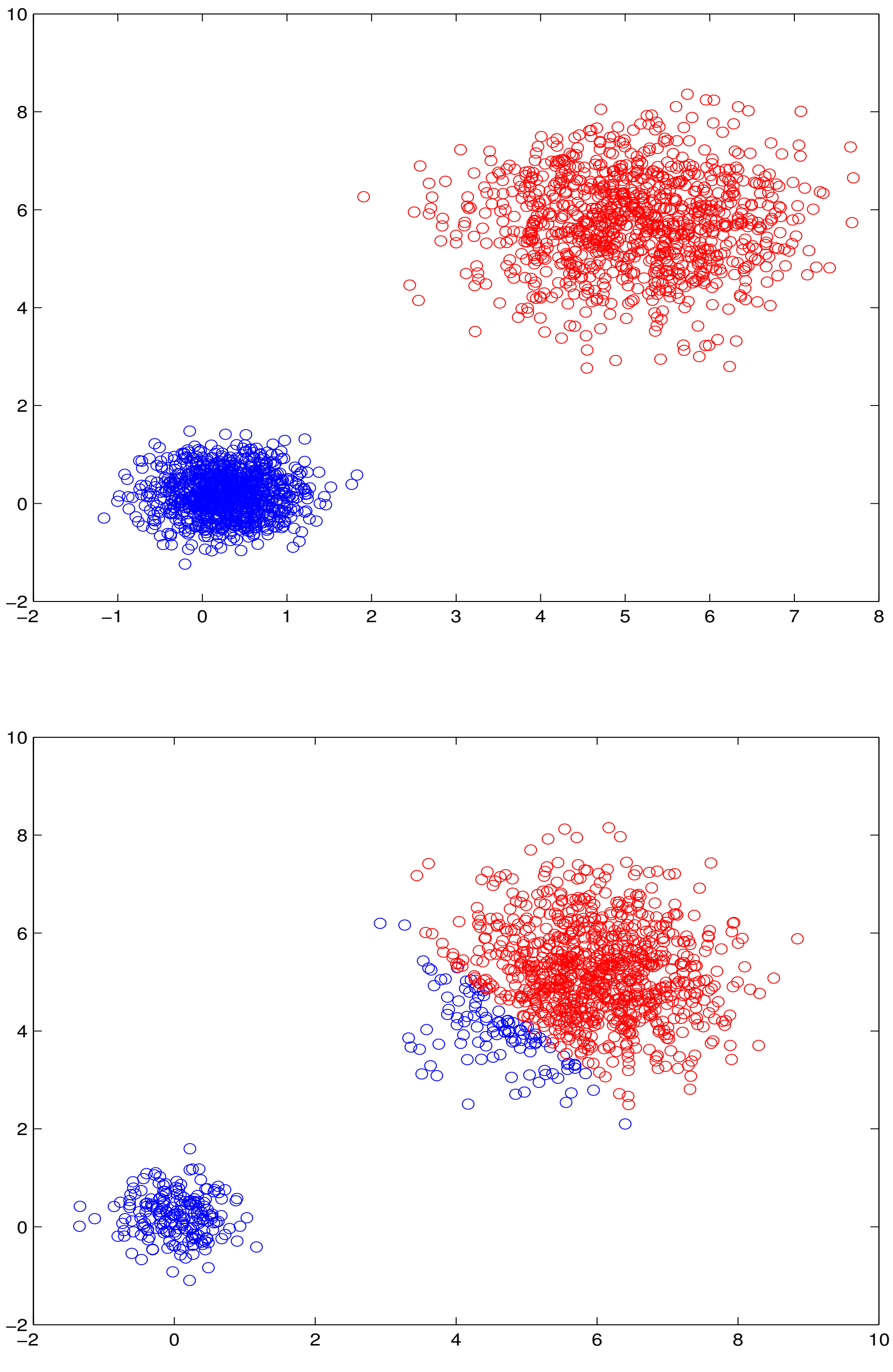


Figure 6.4.1: Two simulated clusters examples. The colours represent the cluster membership, as decided by the sign of the first unit principal component vector.

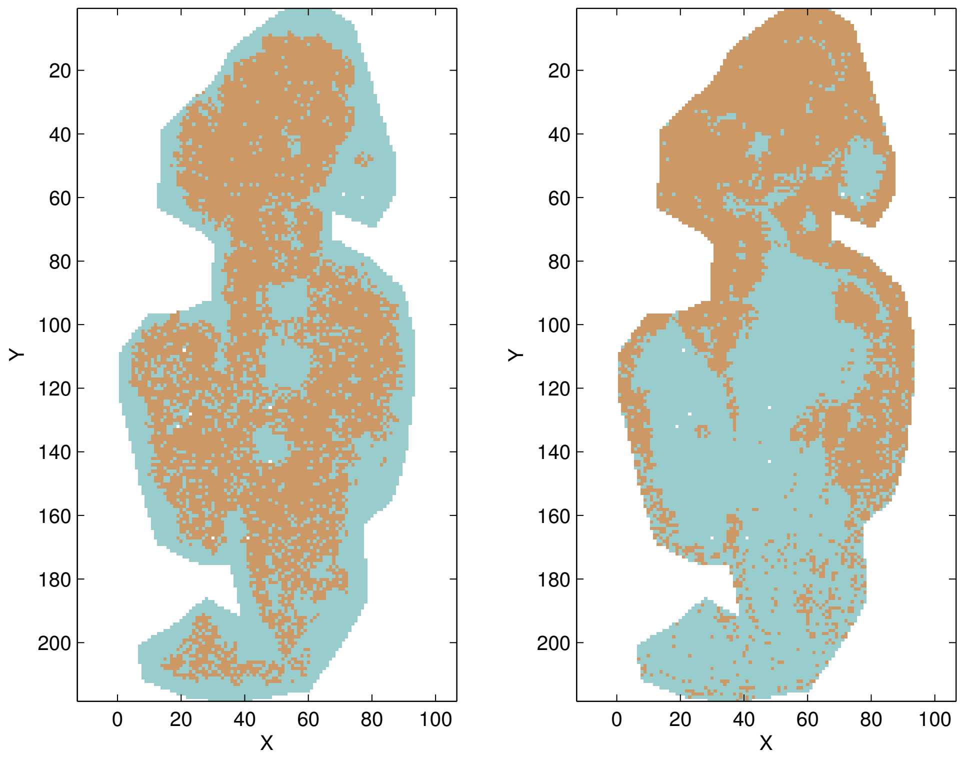


Figure 6.5.1: IMS data: clustering with the sign of the first principal component *(left)* and second principal component *(right)*.

attempt to cluster the log-transformed IMS data.

The results of clustering with the sign of the first principal component and then independently with the sign of the second principal component can be seen in Figure (6.5.1). By visual inspection, the sign of the first principal component distinguishes the cancer and the background from the other tissue types. The sign of the second principal component distinguishes the two tissue types (adipose and stroma) but does not distinguish the cancer tissue.

If we can partition data into two groups with the sign of the first or second principal component, then it seems logical to create more clusters by considering the first and second principal components together. If we consider the signs of the first two principal components we can partition the data into 4 clusters corresponding

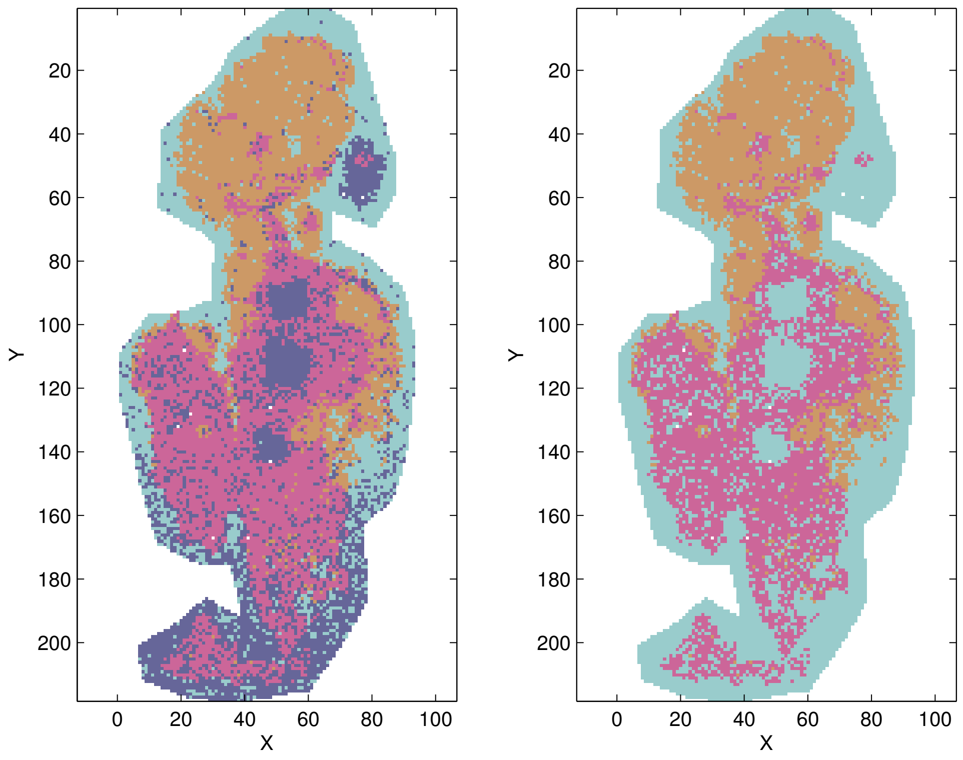


Figure 6.5.2: IMS log transformed data: clustering with the signs of the first and second principal components, 4 clusters *(left)* and 3 clusters *(right)*.

to the combination of signs $\{++, -+, +-, --\}$. Similarly, 3 principal components would create 8 clusters, 4 principal components 16 and so forth.

To create a number of clusters that is not a power of 2, clusters may be combined or collapsed into each other according to hierarchical clustering rule. Recall that in hierarchical clustering we determine the proximity of clusters with *linkage*. In other words, the two clusters with the smallest linkage may be combined to form one cluster. A simple example of a linkage is the Euclidean distance between cluster centres.

Figure (6.5.2) shows the results of clustering with 4 and 3 clusters respectively. In the 4 cluster case, the different tissue types are distinguished, but there is some blurring between the cancer tissue and background. By considering the distances between cluster centroids we find that the cancer and background clusters have the closest centroids. Combining these two clusters gives the 3-cluster arrangement shown in Figure (6.5.2). The 3-cluster results appears to similar to the the result for $k$-means clustering with Euclidean distance found in Chapter 5, with the exception of there being no distinction between cancer and background. Since differentiating between tissue types is more valuable in practice, the 3-cluster PCA result is reasonably successful.

## 6.6 Further comments

It should be noted that Ding and He are not the first to notice that the unit principal component vector is the continuous solution to the $k$-means objective. Drineas et al. (2004) find that the continuous relaxation of the discrete clustering problem can be found via calculating the SVD of the data matrix (which is equivalent to principal component analysis, as I discussed earlier).

Furthermore, the idea of principal component analysis as a tool for clustering has been discussed by Jolliffe (2002, Section 9.2). He suggests that representing the data in two dimensions by the first two PCs can help to visually identify the presence of clusters: "provided that most of the variation, and in particular the between-cluster variation, falls in the two-dimensional subspace defined by the first two PCs" (p. 212). He also makes the point that PCA can be useful for clustering variables as opposed to observations.

Koch (2013, Section 6.5) discusses the role of PCA "as an exploratory tool for choosing the number of clusters." She also clusters observations according to the positive and negative elements of the principal direction vector, and finds that in some cases the first and second principal components are able to identify clear clusters.

## 6.7 A practical application of PCA clustering

As my earlier counter-examples have shown, the PCA clustering method suggested by Ding and He (2004) is not always optimal. However, it may still be useful in practice. A possible application, which was also suggested by Ding and He, is to use the PCA clustering solution as the first step, or seed, in the $k$-means algorithm.

One of the main problems with the $k$-means algorithm is that a solution may not necessarily be optimal, i.e. we may find a local minimum. This can usually be avoided by careful seeding, but choosing a seed and even knowing whether or not the solution obtained is optimal provides further problems.

The upside of the PCA solution is that it is unique (since principal components are always unique up to the sign of the eigenvector). Although not necessarily perfect, the first few principal components can predict the location of the data, and hence this can give us a good starting point for $k$-means.

## 6.8 Conclusion

In this chapter I have examined the theoretical relationship between principal component analysis (PCA) and $k$-means clustering with the Euclidean distance. PCA is an approach that focuses on the variance of data, and variance is based on the Euclidean distance between observations. Similarly, $k$-means with Euclidean distance is also based on variance; variance between and within clusters. It should not be surprising that these two methods are related.

Since the relationship between PCA and $k$-means is based around the Euclidean distance, PCA cannot be easily compared with $k$-means using any other sort of distance measure, such as cosine.

# Chapter 7

# Comparison of Clustering Methods

Up until this point we have considered a number of ways to cluster the IMS data. This includes clustering with $k$-means using Euclidean and cosine distance, clustering after various transformations of the data and clustering using PCA. To conclude the thesis I would like to explore some quantitative ways of comparing these clustering results.

Unlike with classification, it is not usually possible to check a cluster arrangement against 'truth.' By its design, clustering is an exploratory method for finding patterns and groupings within data. Due to this fact, it is not always reasonable to talk about which cluster arrangement is the 'best.' Having said that, there are obviously some criteria for a good clustering arrangement for the IMS data, namely one that separates the tissue types. At this stage the true separation into tissue types cannot be obtained, so we only have a rough visual guide based on the H&E stain.

Regardless of whether we can conclude on a 'best' cluster arrangement or method, comparing methods in itself is a worthwhile exercise. It can be interesting to know which cluster methods produce similar results, which ones are different and by how much.

In this chapter I will discuss a number of ways to compare similarity between two cluster arrangements. This will include simple measures based around the similarity of sets (the Jaccard distance), an entropy based measure (variation of information) and pairwise comparisons (the Rand index). I will be considering ‘cluster validation by prediction strength’ proposed by Tibshirani and Walther (2005), and a comparison measure based on this methodology.

## 7.1 Methods of comparison

I begin by providing details about each comparison method, and I will apply these approaches to the clustering results obtained previously at the end of the chapter.

### 7.1.1 Jaccard distance

The Jaccard index or Jaccard similarity coefficient (first introduced by Jaccard (1901)) is a simple way of measuring the similarity of two sets. It may also be extended to the Jaccard distance, which measures dissimilarity or distance between sets.

**Definition 7.1.1.** *For two finite sets, $A$ and $B$, the Jaccard distance between these two sets is given by:*

$$d_J(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}, \tag{7.1.1}$$

*where $|A|$ is the number of elements in set $A$.*

Identical sets have a Jaccard distance of 0 and disjoint sets have a Jaccard distance of 1.

We may think of a cluster arrangement as a collection of sets. Hence to compare two cluster arrangements, we may do so by calculating the Jaccard distance between corresponding clusters. The first step in this process is to match the clusters between each arrangement. For simplicity, we assume that the number of clusters $k$ is the same for each arrangement.

Consider two cluster arrangements $\mathcal{C} = \{C_1, C_2, \ldots C_k\}$ and $\mathcal{K} = \{K_1, K_2, \ldots K_k\}$. The following steps outline how to calculate $d(\mathcal{C}, \mathcal{K})$, the Jaccard distance between the two arrangements.

1. Calculate the Jaccard distance between all sets in $\mathcal{C}$ and $\mathcal{K}$. That is, $d_J(C_i, K_j)$ for all $i, j = 1, \ldots, k$.

2. Match clusters from each arrangement based on these distances using an optimal assignment algorithm. I have used Munkres algorithm to do this, (see Cao, 2008). Munkres algorithm returns pairs $\{C_i, K_j\}$ such that the sum of Jaccard distances between pairs is minimised. Once matched, label pairs $\{C_1, K_1\}, \{C_2, K_2\}$ etc.

3. Calculate the average Jaccard distance between matched pairs. That is,

$$d(\mathcal{C}, \mathcal{K}) = \frac{1}{k} \sum_{i=1}^{k} d_J(C_i, K_i). \tag{7.1.2}$$

The resulting distance, $d(\mathcal{C}, \mathcal{K})$ will be a number between 0 and 1, close to 0 if the two arrangements are very similar and close to 1 if the two arrangements are very different.

### 7.1.2 Variation of information

The Jaccard distance is a set based distance, which makes sense when we treat cluster arrangements as collections of sets. We may also think of a cluster arrangement as a distribution of data. To compare distributions, we may use techniques from information theory.

Meilă (2007) proposes an information based criterion for comparing cluster arrangements. This criterion, called *variation of information*, uses entropy and mutual information between the two cluster arrangements to measure 'the amount of information lost and gained' in changing from one to the other.

Consider a cluster arrangement $\mathcal{C} = \{C_1, C_2, \ldots, C_K\}$. Consider a map that sends an observation to a cluster. Given an observation $\mathbf{x}$ in $\mathbb{X}$, the probability that it falls into cluster $C_k$ will be given by:

$$P(k) = \frac{|C_k|}{n} = \frac{n_k}{n}. \tag{7.1.3}$$

In this way we define a discrete random variable associated with the observations and cluster arrangement $\mathcal{C}$. The *entropy* of this random variable is defined like so:

$$\mathcal{H}(\mathcal{C}) = -\sum_{k=1}^{K} P(k)\log(P(k)). \tag{7.1.4}$$

The entropy measures the uncertainty of the random variable. It only takes the value 0 when there is no uncertainty, in other words when there is only one cluster.

Now consider two cluster arrangements $\mathcal{C}$ and $\mathcal{K}$ (the number of clusters in each is not necessarily equal). The probability that an observation $\mathbf{x}$ falls into cluster $C_k$ in $\mathcal{C}$ and $K_{k'}$ in $\mathcal{K}$ is given by:

$$P(k,k') = \frac{|C_k \cap K_{k'}|}{n} = \frac{n_{k,k'}}{n}. \tag{7.1.5}$$

We also define the *mutual information* between two cluster arrangements, $\mathcal{C}$ and $\mathcal{K}$:

$$\mathcal{I}(\mathcal{C},\mathcal{K}) = \sum_{k=1}^{K}\sum_{k'=1}^{K'} P(k,k')\log\left(\frac{P(k,k')}{P(k)P(k')}\right). \tag{7.1.6}$$

The mutual information tells us how much information one cluster arrangement has about the other. In other words, given that we know the cluster allocation of $\mathbf{x}$ in $\mathcal{C}$, how much does that reduce the uncertainty of its allocation in $\mathcal{K}$? The mutual information is always less than or equal to the individual entropies of $\mathcal{C}$ and $\mathcal{K}$, with equality only when the cluster arrangements are identical.

**Definition 7.1.2.** *For two cluster arrangements $\mathcal{C}$ and $\mathcal{K}$, the* variation of information (VI) *is given by:*

$$VI(\mathcal{C},\mathcal{K}) = \mathcal{H}(\mathcal{C}) + \mathcal{H}(\mathcal{K}) - 2\mathcal{I}(\mathcal{C},\mathcal{K}). \tag{7.1.7}$$

The variation of information is always positive, and 0 when the two cluster arrangements are identical. Unlike the Jaccard distance, the VI may exceed 1.

### 7.1.3 The Rand index

Cluster arrangement comparisons can also be made by considering pairs of observations. If we consider two observations $\mathbf{x}_1$ and $\mathbf{x}_2$ from $\mathbb{X}$, and two cluster arrangements $\mathcal{C}$ and $\mathcal{K}$ then there are four possibilities for the pair $(\mathbf{x}_1, \mathbf{x}_2)$:

- They appear in the same cluster in both $\mathcal{C}$ and $\mathcal{K}$.
- They appear in different clusters in both $\mathcal{C}$ and $\mathcal{K}$.
- They appear in the same cluster in $\mathcal{C}$ and different clusters in $\mathcal{K}$.
- They appear in different clusters in $\mathcal{C}$ and the same cluster in $\mathcal{K}$.

The number of pairs in each of these cases are denoted $N_{00}, N_{11}, N_{01}$ and $N_{10}$ respectively.

The *Rand index*, first introduced by Rand (1971), provides a measure of similarity between two cluster arrangements and is defined as 'the number of similar assignments of point-pairs normalised by the total number of point-pairs.'

**Definition 7.1.3.** *For two clusterings $\mathcal{C}$ and $\mathcal{K}$, where $N_{00}$ and $N_{11}$ denote the number of point-pairs that appear in the same cluster in both $\mathcal{C}$ and $\mathcal{K}$, and different clusters in both $\mathcal{C}$ and $\mathcal{K}$ respectively, the* Rand index (RI) *is given by:*

$$RI = \frac{N_{00} + N_{11}}{\binom{n}{2}}. \tag{7.1.8}$$

The RI is a simple way to measure *consistency* between two clusterings. Identical clusterings will have a RI of 1, and two completely different clusterings will have a RI close to 0. To implement RI I use the code of De Bie (2003).

### 7.1.4 Prediction strength

Tibshirani and Walther (2005) have borrowed ideas from discriminate analysis to determine what they call the prediction strength of a cluster arrangement. Prediction strength compares how well a clustering derived from a training set performs on a testing set, where the training and testing sets are both subsets of the original data. In other words, how well does a cluster arrangement trained on a subset of data predict the cluster allocation of the rest of the data? They use this idea to help decide the best number of clusters to use; the best cluster arrangement should have the highest prediction strength.

With some adjustment, prediction strength can also be applied to the comparison of different cluster arrangements. The question can be changed to: how well does one cluster arrangement predict a different cluster arrangement?

As with the Rand index, this measure is also based on pair-points. We consider the *co-membership matrix*, which can either be defined for a single cluster arrangement, or one cluster arrangement relative to another.

**Definition 7.1.4.** *For fixed $k > 0$, let $\mathcal{C}(\mathbb{X}, k)$ be a $k$-cluster arrangement on $\mathbb{X}$. The* co-membership matrix *$D$ corresponding to $\mathbb{X}$ has entries:*

$$D_{ij} = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \in \mathbb{X} \text{ belong to the same cluster,} \\ 0 & \text{otherwise.} \end{cases}$$

**Definition 7.1.5.** *Let $\mathbb{X}'$ be data of the same dimension as $\mathbb{X}$, with $k$-cluster arrangement $\mathcal{C}(\mathbb{X}', k)$. The* co-membership matrix *$D[\mathcal{C}(\mathbb{X}, k), \mathbb{X}']$ of $\mathbb{X}'$ relative to the cluster arrangement $\mathcal{C}(\mathbb{X}, k)$ of $\mathbb{X}$ has entries:*

$$D[\mathcal{C}(\mathbb{X}, k), \mathbb{X}']_{ij} = \begin{cases} 1 & \text{if } \mathbf{x}'_i \text{ and } \mathbf{x}'_j \text{ belong to the same cluster in } \mathcal{C}(\mathbb{X}, k), \\ 0 & \text{otherwise.} \end{cases}$$

These ideas of co-membership matrices can also be extended to considering two different clusterings on the same data, if we take $\mathbb{X} = \mathbb{X}'$. As with the RI, the co-membership matrices give us an idea of the consistency between two cluster arrangements. This leads to Tibshirani and Walther's *prediction strength*,

$$PS(\mathcal{C}) = \min_{1 \le \ell \le k} \left\{ \frac{1}{n'_\ell(n'_\ell - 1)} \sum_{\mathbf{x}'_i \neq \mathbf{x}'_j \in C'_\ell} D[\mathcal{C}(\mathbb{X}, k), \mathbb{X}']_{ij} \right\} \tag{7.1.9}$$

which I have modified for the problem of comparing two clusterings, $\mathcal{C}$ and $\mathcal{K}$ of the same data $\mathbb{X}$ in the following way:

$$PS(\mathcal{K}, \mathcal{C}) = \min_{1 \le \ell \le k} \left\{ \frac{1}{n_\ell(n_\ell - 1)} \sum_{\mathbf{x}_i \neq \mathbf{x}_j \in K_\ell} D[\mathcal{C}(\mathbb{X}, k), \mathbb{X}]_{ij} \right\} \tag{7.1.10}$$

where $n_\ell$ is the number of elements in cluster $K_\ell$ of $\mathcal{K}$. For two cluster arrangements $\mathcal{C}$ and $\mathcal{K}$, we consider each cluster in $\mathcal{K}$ one at a time, and count the proportion of point-pairs in that cluster that are also in the same cluster in $\mathcal{C}$. Prediction strength will then be equal to the lowest proportion out of all the clusters.

Unlike all of the cluster comparison methods we have seen so far, this prediction strength is not symmetric, as in general $PS(\mathcal{K}, \mathcal{C}) \neq PS(\mathcal{C}, \mathcal{K})$.

Tibshirani and Walther (2005) also discuss the *prediction error loss* for a cluster arrangement, but once again, this may also be extended to a method for comparing two cluster arrangements.

Let $\mathcal{C}^*(\mathbb{X})$ denote the true grouping of the data $\mathbb{X}$. Define the *prediction error loss* of the clustering procedure $\mathcal{C}$ by

$$\mathcal{E}(\mathcal{C}) = \frac{1}{n^2} \sum_{i,j=1}^{n} \left| D[\mathcal{C}^*(\mathbb{X}, k), \mathbb{X}] - D[\mathcal{C}(\mathbb{X}, k), \mathbb{X}] \right|. \tag{7.1.11}$$

This can be easily adapted for cluster comparison: instead of comparing a cluster arrangement against a true grouping, compare two arbitrary cluster arrangements $\mathcal{C}$ and $\mathcal{K}$, by defining:

$$\mathcal{E}(\mathcal{C}, \mathcal{K}) = \frac{1}{n^2} \sum_{i,j=1}^{n} \left| D[\mathcal{C}(\mathbb{X}, k), \mathbb{X}] - D[\mathcal{K}(\mathbb{X}, k), \mathbb{X}] \right|. \tag{7.1.12}$$

Unlike the prediction strength, this comparison is symmetric. It can be interpreted as the proportion of point-pairs that are differently assigned to the same group plus the proportion of point-pairs that are differently assigned to different groups.

## 7.2 Application of measures of comparison

In the following section I am going to use the comparison methods to compare some of the clustering results from earlier chapters.

### 7.2.1 Jaccard distance and variation of information: comparisons with the binary transformation

In earlier chapters, we found that using a binary transformation before clustering gave a result that seemed to match well to the tissue types as noted by the pathologist. One undesirable feature of this transformation was that it involved throwing away information about the ion intensities. Consequently, we would like to find a transformation or normalisation that gives a similar clustering result to the binary, but does not involve throwing away information. In previous chapters we found that the log transformation and log transformation with PIC seem to lead to similar clustering results to the binary.

I will now use the Jaccard distance and variation of information (VI) to quantitatively assess the similarity of these cluster arrangements. Eight clustering results have been compared to binary with Euclidean and binary with cosine (called Bin. E and Bin. C in Table 7.2.1 respectively):

1. raw data with Euclidean

2. raw data with cosine

3. log data with Euclidean

4. log data with cosine
5. PIC normalised data with Euclidean
6. PIC normalised data with cosine
7. log data PIC with Euclidean
8. log data PIC with cosine

The numerical results can be found in Table 7.2.1, and Figure 7.2.1 shows a visual comparison. The closer the Jaccard distance is to 0, the more similar the cluster arrangements. The same is true for VI, but VI can be greater than 1.

There are essentially two factors of interest: how much does clustering change after different transformations of the data, and how much does clustering change after changing the clustering distance (Euclidean vs. cosine distance). The information we get from the Jaccard and VI tell us more or less the same story.

The binary cluster arrangements are closest to the log transformed PIC cluster arrangements (3,4,7,8), and furthest from the raw and PIC scaled without log (1,2,5,6). The Euclidean clusterings are most similar to each other and the cosine clusterings are also closest to each other (illustrated by the graph in Figure 7.2.1, where the red and blue lines alternate being on top). When the difference to the binary clustering is small, the actual clustering distance (Euclidean or cosine) seems to matter more (see points 3,4,7,8 on Figure 7.2.1).

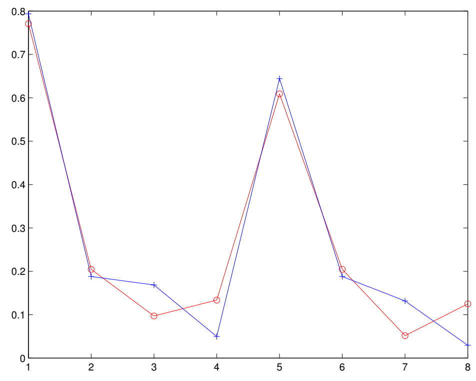
### 7.2.2 Comparing PCA clustering with $k$-means clustering

One advantage that the VI has over the Jaccard distance is that it compares cluster arrangements holistically, and hence it is possible to compare arrangements where the number of clusters are different. This may have useful application in choosing the best number of clusters - i.e. how much information is lost or gained by adding another cluster?

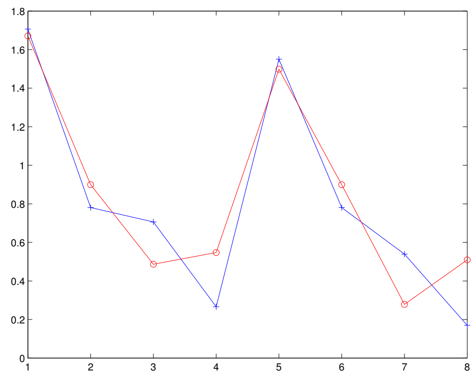
Table 7.2.1: Jaccard (and VI distances) between cluster arrangements

| | 1. Raw E. | 2. Raw C. | 3. Log E. | 4. Log C. |
|---|---|---|---|---|
| Bin. E. | 0.7709 (1.6708) | 0.2046 (0.8994) | 0.0971 (0.4864) | 0.1337 (0.5476) |
| Bin. C. | 0.7935 (1.7063) | 0.1881 (0.7808) | 0.1685 (0.7063) | 0.0497 (0.2661) |

| | 5. PIC E. | 6. PIC C. | 7. Log-PIC E. | 8. Log-PIC C. |
|---|---|---|---|---|
| Bin. E. | 0.6090 (1.4982) | 0.2046 (0.8994) | 0.0517 (0.2786) | 0.1249 (0.5092) |
| Bin. C. | 0.6444 (1.5505) | 0.1881 (0.7808) | 0.1317 (0.5390) | 0.0295 (0.1700) |



(a) Jaccard distances



(b) VI

Figure 7.2.1: Jaccard distances (a) and VI (b) between binary data clustered with Euclidean *(red)* and binary data clustered with cosine *(blue)* with arrangements listed 1-8 in Table 7.2.1.

In Chapter 6 I discussed some cluster arrangements formed with principal component analysis (PCA) on the log transformed data. I considered arrangements of two, three and four clusters. I would like to compare how close these are to the $k$-means cluster result on the log data (using Euclidean distance).

The results of this comparison are displayed in Table 7.2.2 and Figure 7.2.2. Note that the clusters formed with PCs are closer to each other than to the $k$-means

Table 7.2.2: VI between $k$-means and PCA clusterings

| | PC-4 | PC-3 | PC-2 |
|---|---|---|---|
| $k$-means (4 clusters) | 0.9984 | 0.7936 | 1.0442 |
| PC-4 | | 0.3140 | 0.6897 |
| PC-3 | | | 0.3757 |

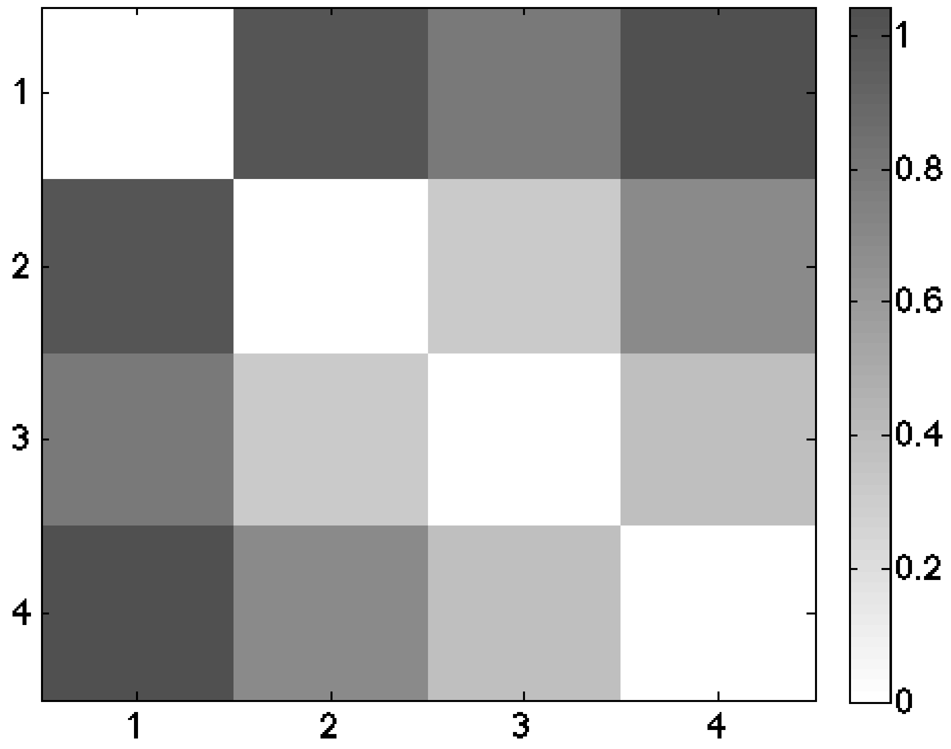


Figure 7.2.2: Grayscale heat map of the VI between 1. $k$-means, 2. PC-4, 3. PC-3, 4. PC-2. VI of 0 indicates that the cluster arrangements are identical.

cluster arrangement. This should not be surprising, since the 2-cluster and 3-cluster PC arrangements can be constructed by combining clusters from the 4-cluster PC arrangement, hence there should be some preservation of information. The PC-4 and PC-3 arrangements are closest, followed by the PC-3 and PC-2 arrangements.

The VI increases (indicates less similarity) when we compare the PC-2 with the PC-4, which is understandable when comparing an arrangement with 2 clusters to an arrangement with 4 clusters.

The $k$-means cluster arrangement has higher VI when compared with the other three. It is closest to the PC-3 arrangement, which was also what I found based on the images in Chapter 6: when the background cluster is combined with the cancer cluster, the result looks similar to the $k$-means.

### 7.2.3 Rand index: compare clustering on full dataset with clustering on principal components

So far we have seen what happens when we cluster a dataset with principal components and how this is similar to clustering with $k$-means based on Euclidean distance. Previously, I clustered the data with the signs of the first two components, but in this case I would like to use $k$-means clustering on the principal component data in fewer than $p$ dimensions. In particular, how many principal components do we need for the resulting cluster arrangement to become close to the cluster arrangement formed on the full dataset? I have used the Rand index (RI) to investigate this question.

I considered the $k$-means Euclidean cluster arrangement on the log data. I then calculated the RI between this and the cluster arrangement on up to 50-dimensional principal component data. The RI for these is shown in Figure 7.2.3. From the graph it would appear that only as few as 10 principal components are needed before the clustering results have a RI of 0.99 with the clustering result on the full data.

Once again the relationship between PCA and $k$-means clustering is heavily underlined by the use of the Euclidean distance (as PCA is based on Euclidean variance). If we were to cluster the principal components using the cosine distance, we would see a very different result. In Figure 7.2.4 the RI for cosine clustering on PCs and the full dataset shows that adding more PCs does not help cosine clustering:

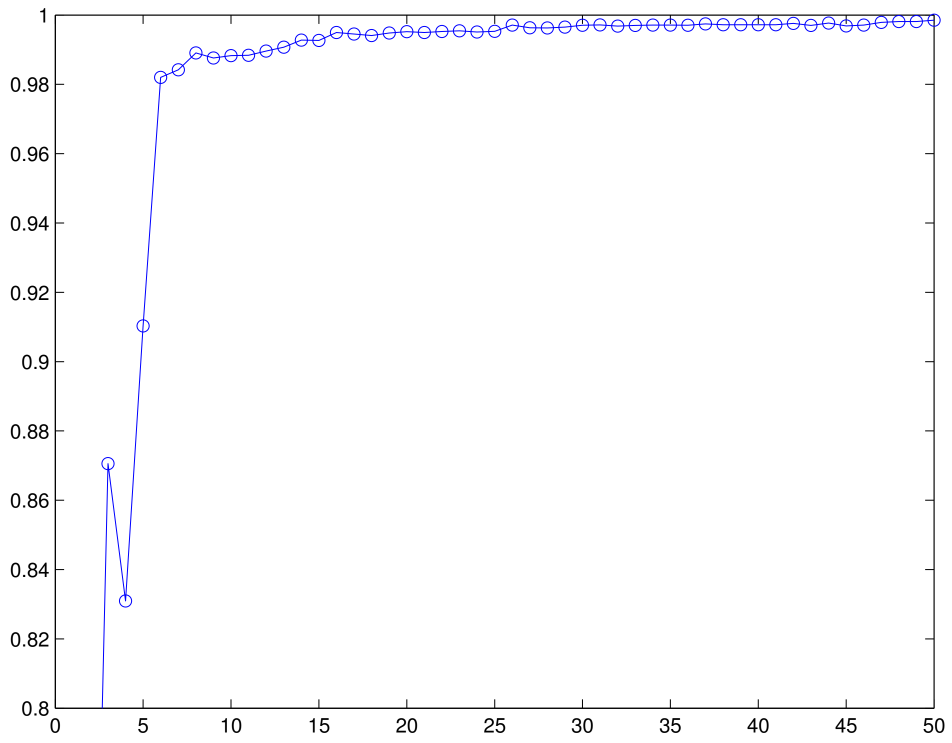


Figure 7.2.3: RI of $k$-means result on $q$-dimensional principal component data vs $k$-means with full data set *(vertical)* vs number of principal components *(horizontal)*.

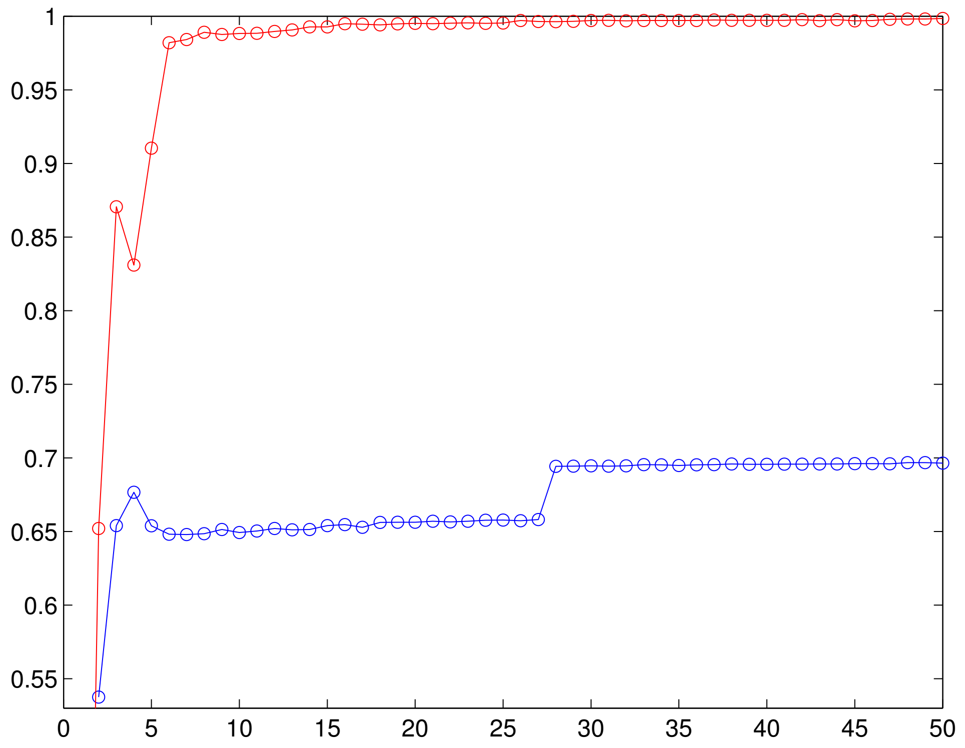


Figure 7.2.4: RI with full data set *(vertical)* vs number of principal components *(horizontal)*, Euclidean *(red)* and cosine *(blue)*.

the RI remains static at around 0.7. This could once again emphasise the fact that Euclidean and cosine distance are very different concepts, often giving very different results.

### 7.2.4 Prediction strength and prediction error loss: a four-way comparison

I have used the remaining pairwise methods (prediction strength and prediction error loss) to compare cluster arrangements formed with just the Euclidean distance on four versions of the data: binary, log, PIC and log-PIC. These quantities are presented in grayscale heat maps (Figure 7.2.5). A colour bar has been added to

Table 7.2.3: Prediction strength *(left)* and prediction error loss *(right)*. Legend: 1. binary, 2. log, 3. PIC, 4. log-PIC.

| | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0.8391 | 0.4150 | 0.9177 | 0 | 0.0654 | 0.3388 | 0.0324 |
| 2 | 0.8117 | 1 | 0.4232 | 0.8648 | 0.0654 | 0 | 0.3220 | 0.0598 |
| 3 | 0.3570 | 0.3884 | 1 | 0.3800 | 0.3388 | 0.3220 | 0 | 0.3226 |
| 4 | 0.8887 | 0.8461 | 0.4277 | 1 | 0.0324 | 0.0598 | 0.3226 | 0 |

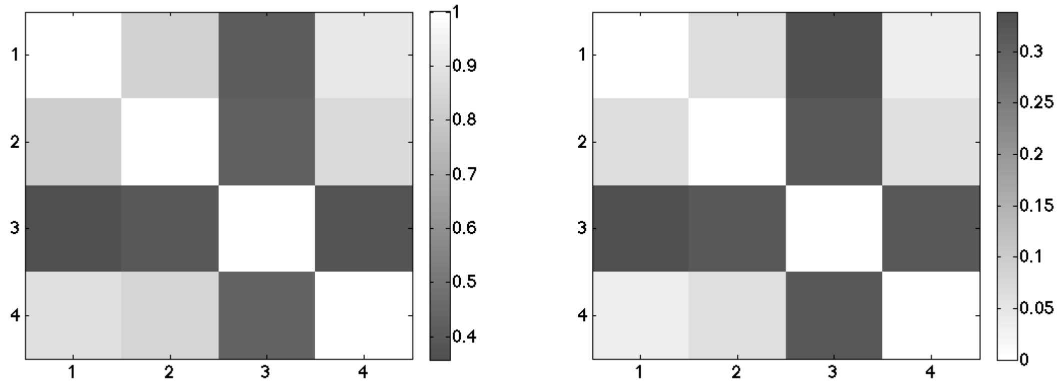


Figure 7.2.5: Grayscale heat map of the comparisons with the prediction strength *(left)* and error prediction loss *(right)*. Legend: 1. binary, 2. log, 3. PIC, 4. log-PIC.

each figure as the maps are not all based on the same scale, for instance a value of 1 on prediction strength is a perfect match and a value of 1 on the prediction error loss is a perfect mismatch (a perfect match has value 0). The colour white represents a perfect match.

The results of the comparison with prediction strength and prediction error loss are similar. Note that prediction strength is not symmetric: for example, the binary clustering ‘predicts’ the log-PIC clustering better than the log-PIC predicts the binary (0.9177 vs. 0.8887). The general pattern is the same: the PIC clustering is the odd one out, being quite different from the other three. This is consistent with what we saw in Chapter 5. The two closest clustering results are from the binary data and log-PIC.

## 7.3 Conclusion

The various methods for cluster arrangement comparison that I have discussed have all produced results that are comparable. For example, each method tells us that the binary data clustered with the Euclidean distance is "closest" to the data transformed with log-PIC, clustered with Euclidean distance. The comparison methods mostly differ in their philosophy towards comparisons, and whether we treat a cluster arrangement as a collection of sets or a distribution of data. Good agreement between different methods gives the user confidence that the cluster arrangements are indeed close, or not close as the case may be.

In practice, any of these methods should be useful and relatively easy to use. Even with a dataset as a large as 14,000 observations it is not computationally expensive to implement multiple comparisons.

# Chapter 8

# Conclusion

The general aim of this thesis has been to present a toolbox for exploratory analysis of proteomics IMS data. I have presented a number of different approaches to clustering by varying such quantities as clustering method, parameters such as distance, and carrying out various preparatory transformations of the data.

As I show in the data analysis, some of the naively applied methods do not produce interpretable results. However, application of transforms prior to clustering, appropriate normalisation or use of different distance measures can lead to interpretable results which are of interest to the biologist and medical experts.

The comparisons between different cluster approaches, which I introduce and apply to the data, provide quantitative answers to how similar alternate cluster approaches are, thereby allowing exclusion of methods that produce very different and non-interpretable results.

A natural next step in the analysis of proteomics data is the classification of IMS data and prediction of cancerous regions of patients with tumorous tissue. The different approaches and the comparison measures I introduced and applied in Chapter 7 are expected to be informative in a subsequent classification of spectra.

# Bibliography

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

David Bonnel, Rémi Longuespee, Julien Franck, Morad Roudbaraki, Pierre Gosset, Robert Day, Michel Salzet, and Isabelle Fournier. Multivariate analyses for biomarkers hunting and validation through on-tissue bottom-up or in-source decay in maldi-msi: application to prostate cancer. *Analytical and bioanalytical chemistry*, 401(1):149–165, 2011.

Jocelyne Bruand, Theodore Alexandrov, Srinivas Sistla, Maxence Wisztorski, Céline Meriaux, Michael Becker, Michel Salzet, Isabelle Fournier, Eduardo Macagno, and Vineet Bafna. Amass: algorithm for msi analysis by semi-supervised segmentation. *Journal of proteome research*, 10(10):4734–4743, 2011.

Yi Cao. Munkres' assignment algorithm. `http://au.mathworks.com/matlabcentral/fileexchange/20328-munkres-assignment-algorithm/content/munkres.m`, June 2008.

Tilj De Bie. Adjusted rand index. `http://www.kernel-methods.net/matlab/algorithms/adjrand.m`, February 2003.

Sören-Oliver Deininger, Matthias P Ebert, Arne Fütterer, Marc Gerhard, and Christoph Röcken. Maldi imaging combined with hierarchical clustering as a new tool for the interpretation of complex human cancers. *Journal of proteome research*, 7(12):5230–5236, 2008.

Sören-Oliver Deininger, Dale S Cornett, Rainer Paape, Michael Becker, Charles Pineau, Sandra Rauser, Axel Walch, and Eryk Wolski. Normalization in maldi-tof imaging datasets of proteins: practical considerations. *Analytical and bioanalytical chemistry*, 401(1):167–181, 2011.

Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning*, page 29. ACM, 2004.

Petros Drineas, Alan Frieze, Ravi Kannan, Santosh Vempala, and V Vinay. Clustering large graphs via the singular value decomposition. *Machine learning*, 56 (1-3):9–33, 2004.

Frédéric Ferraty and Philippe Vieu. *Nonparametric functional data analysis: theory and practice*. Springer, 2006.

Johann A Gagnon-Bartsch and Terence P Speed. Using control genes to correct for unwanted variation in microarray data. *Biostatistics*, 13(3):539–552, 2012.

Paul Garrett. Compact operators on hilbert space, February 2012.

Jürgen H Gross. *Mass spectrometry: a textbook, 2nd Edition.* Springer Science & Business Media, 2010.

Johan OR Gustafsson. *Molecular characterization of metastatic ovarian cancer by MALDI imaging mass spectrometry.* PhD thesis, The University of Adelaide, 2011.

Johan OR Gustafsson, Martin K Oehler, Andrew Ruszkiewicz, Shaun R McColl, and Peter Hoffmann. Maldi imaging mass spectrometry (maldi-ims) - application of spatial proteomics for ovarian cancer classification and diagnosis. *International journal of molecular sciences*, 12(1):773–794, 2011.

Peter Hoffmann. Lecture notes in peptides, proteins and structure determination by mass spectrometry, October 2012.

Paul Jaccard. *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz, 1901.

Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.

Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

Inge Koch. *Analysis of Multivariate and High Dimensional Data*. Cambridge University Press, New York, 2013.

Marina Meilă. Comparing clusteringsan information based distance. *Journal of multivariate analysis*, 98(5):873–895, 2007.

Nawin C Mishra. *Introduction to proteomics: principles and applications*, volume 148. John Wiley & Sons, 2011.

Jeffrey S Morris. Statistical methods for proteomic biomarker discovery based on feature extraction or functional modeling approaches. *Statistics and its interface*, 5(1):117–136, 2012.

J.O. Ramsay and B.W. Silverman. *Functional Data Analysis*. Springer Series in Statistics. Springer, 1997.

William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971.

Robert Tibshirani and Guenther Walther. Cluster validation by prediction strength. *Journal of Computational and Graphical Statistics*, 14:511–528, 2005.

Dennis Trede, Stefan Schiffler, Michael Becker, Stefan Wirtz, Klaus Steinhorst, Jan Strehlow, Michaela Aichler, Jan Hendrik Kobarg, Janina Oetjen, Andrey Dyatlov, et al. Exploring three-dimensional matrix-assisted laser desorption/ionization imaging mass spectrometry data: three-dimensional spatial segmentation of mouse kidney. *Analytical chemistry*, 84(14):6079–6087, 2012.

Jane-Ling Wang, Jeng-Min Chiou, and Hans-Georg Mueller. Review of functional data analysis. *arXiv preprint arXiv:1507.05135*, 2015.

Lyron Winderbaum. *PhD Thesis*. PhD thesis, The University of Adelaide, 2015.