UNIVERSITY OF CAPE TOWN

PhD Thesis

Biplots based on Principal Surfaces

Author:
Raeesa GANEY

Supervisor:
Dr Sebnem ER
Co-supervisor:
Professor Sugnet LUBBE

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

Department of Statistical Sciences

January, 2020

University of Cape Town

The copyright of this thesis vests in the author. No quotation from it or information derived from it is to be published without full acknowledgement of the source. The thesis is to be used for private study or non-commercial research purposes only.

Published by the University of Cape Town (UCT) in terms of the non-exclusive license granted to UCT by the author.

University of Cape Town

Declaration of Authorship

I, Raeesa GANEY, declare that this thesis titled, "Biplots based on Principal Surfaces" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Signed by candidate

Date: January 2020

UNIVERSITY OF CAPE TOWN

Abstract

Faculty of Science
Department of Statistical Sciences

Doctor of Philosophy

Biplots based on Principal Surfaces

by Raeesa GANEY

Principal surfaces are smooth two-dimensional surfaces that pass through the middle of a p-dimensional data set. They minimise the distance from the data points, and provide a nonlinear summary of the data. The surfaces are non-parametric and their shape is suggested by the data. The formation of a surface is found using an iterative procedure which starts with a linear summary, typically with a principal component plane. Each successive iteration is a local average of the p-dimensional points, where an average is based on a projection of a point onto the nonlinear surface of the previous iteration.

Biplots are considered as extensions of the ordinary scatterplot by providing for more than three variables. When the difference between data points are measured using a Euclidean embeddable dissimilarity function, observations and the associated variables can be displayed on a nonlinear biplot. A nonlinear biplot is predictive if information on variables is added in such a way that it allows the values of the variables to be estimated for points in the biplot. Prediction trajectories, which tend to be nonlinear are created on the biplot to allow information about variables to be estimated.

The goal is to extend the idea of nonlinear biplot methodology onto principal surfaces. The ultimate emphasis is on high dimensional data where the nonlinear biplot based on a principal surface allows for visualisation of samples, variable trajectories and predictive sets of contour lines. The proposed biplot provides more accurate predictions, with an additional feature of visualising the extent of nonlinearity that exists in the data.

Keywords: Biplots; Principal surfaces; Nonparametric principal components; Multidimensional scaling

Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor Professor Sugnet Gardner-Lubbe for the continuous support of my Ph.D study and related research, for her patience, motivation, and immense knowledge. Her guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study.

I would like to thank my dear friend, Hassan Sadiq for always inspiring me in the field of statistics. I would not be where I am today, if not for Hassan. Thank you to my colleagues at the University of Witwatersrand, Dr Cheryl Chamberlain for being my mentor, Professor Loyiso Nongxa for being a helpful neighbour, Professor Rukshana Osman for always giving me great advice and making the time to see me.

Last but not the least, I would like to thank my family: my husband, Zain for his love and spiritual support throughout the writing of this thesis and in life in general and to my parents for encouraging me in all of my pursuits and inspiring me to follow my dreams.

Contents

List of Figures

List of Tables

Dedicated to my dear friend, Hassan Sadiq.

Chapter 1

Introduction

Data visualisation is the graphical representation of data that brings together, in perfect concord, information science with modern presentation design techniques. Visualisations are able to bring data together in a coherent and insightful manner. It provides a powerful way to present data in multiple dimensions, and to simplify complex analyses. The visual presentation of data is not a new concept. In the 19th century Florence Nightingale was among one of the first to use statistical graphics, famously known as the 'Coxcombs' diagrams to highlight the greater deaths from disease compared to wounds in the Crimean War, thereby promoting sanitation in hospitals.

Visualisation techniques and the tools available to produce visualisations have developed in recent years. Where simple black points on white plots and charts are where it all began, intricate colour graphics and innovative dynamics in data visualisations have become available and are more popular. Particularly, in the age of 'big data', visualisation techniques have become attractive to use. Visualisation is often the first formal step of an analysis, but can also serve as an inferential tool in different stages of the analysis. The demand of new methods to deal with big data has become a common theme in most scientific fields.

There are other aspects of big data, that not only has its root in statistics, but from mathematics and computer science as well. Some of these include: data wrangling, which involves manipulating data by selecting certain variables and features to analyse; dimension reduction, which involves reducing dimensions to facilitate visualisation; sequential learning, which treats data successively that has been processed sequentially; representation learning, which involves feature and deep learning that aims to uncover hidden structures in data; and many more.

The theme of this thesis will be on visualisation, particularly in high dimensional data. The thesis will focus on two ideas in visualisation techniques, namely principal surfaces and biplots. The goal is to provide a new visualisation method that connects these ideas.

A principal surface is a surface that is positioned through a set of higher dimensional data, such that the orthogonal distances from the data and the surface are minimised. A principal surface therefore, follows the shape of the data, and tends to be nonlinear. A biplot, on the other hand, is a generalisation of a two-dimensional scatter plot of data that exists in higher dimensions. The biplot displays information on the observations in the data and the variables describing the data all in one plot. A biplot also provides an additional feature that is of significance in which observations can be predicted or new observations can be interpolated in the biplot.

The construction of a typical biplot uses a linear plane that passes through the center of the higher dimensional data, such that the orthogonal distances from the plane to the data are a minimum. The main aim of this thesis, is to construct a biplot using a principal surface instead of the flat plane used in classical biplot methodology.

Earlier work on combining principal curves and biplots exists in De'ath (1999), however the attempt involves superimposing a principal curve onto a principal component analysis biplot, whereas in this thesis, the focus is on the construction of biplots using a principal surface.

Since the principal surface is not constrained to be linear, but is rather designed to be flexible, the proposed biplot will be more accurate in terms of sample representation and prediction. Other interesting features will become visible, such as displaying the extent of the nonlinearity that exists in the data, better class separation of data points and trends in the sample representation. The proposed biplot will present itself in the same fashion as a classical biplot, in terms of visualisation and prediction. The thesis will highlight the benefits of using such a biplot through its configurations, as well as present differences and similarities with existing biplots.

The thesis is organised as follows:

- In Chapter 2, a theoretical overview on principal surfaces is provided. The principal surface algorithm for both probability distributions and for data sets is discussed. The chapter ends with a demonstration of the formation of a principal surface on two different sets of data.

- In Chapter 3, the basic understanding of biplots is defined. It includes an overview of the basic biplot, which is the principal component analysis biplot. The chapter illustrates in detail how samples and variables are represented in a biplot.

- Chapter 4 and 5, include further demonstrations of other types biplots, nonlinear biplots and spline-based nonlinear biplots, respectively. The nonlinear biplot is an extension of the PCA biplot that makes provision for other distance measures, where spline-based nonlinear biplots are constructed using B-splines. Chapter 5, containing spline-based nonlinear biplots, provides an application to principal surfaces.

- The original work of the thesis, which is biplots based on principal surfaces begins from Chapter 6. In Chapter 6, focus is placed on (1) representing samples in the biplot and (2) interpolating a larger number of points on the principal surface to (3) creating intersection spaces to predict samples.

- Chapter 7 uses the intersection spaces to construct biplot axes through three different designs. The best design to construct biplot axes is determined by the design that gives the best predictions of samples in the data.

- Chapter 8 defines a prediction accuracy measure that can be used to compare predictions from different biplots. Further in Chapter 8, a motivation is given to first center data before constructing the principal surface biplot to improve prediction of samples. It is then established in this chapter, that the constructed biplot axes are not always accurate for prediction due to the

nonlinearity that can exist in the data, and therefore, another approach is taken to predict samples in the principal surface biplot.

- Chapter 9 provides four main contributions that the principal surface biplot can make to multivariate visualisation. First, the nonlinearity of the intersection spaces and biplot axes show the extent of linearity that exists in data. Second, the prediction of edge points in data that could potentially be outliers in the data are predicted better in a principal surface biplot. Third, separation of classes in nonlinear data can be better seen in a principal surface biplot. Lastly, to present nonlinear prediction in a large data set with an added feature of presenting the extent of nonlinearity in the data, which other biplots does not show.

- Chapter 10 concludes the thesis with recommendations and limitations to the theory proposed.

Chapter 2

Principal Surfaces

In this chapter, principal surface models in the current literature are defined. The theory of principal surfaces stems from the idea of principal curves first introduced by Hastie and Stuetzle (1989). The chapter will begin by briefly defining a principal curve model with its link to principal surfaces. Thereafter, principal surface models will be defined for a p-dimensional probability distribution, and then for a p-dimensional finite data set. Lastly, demonstrations of the principal surface algorithm will be applied on two sets of data.

2.1 Introduction

Principal curves are smooth one-dimensional curves that pass through the middle of a p-dimensional data set and provide a nonlinear summary of the data. The curves are non-parametric and their shape is suggested by the data.

In order to construct a principal curve, the first principal component is used as the initial estimate. It is a one-dimensional line that approximates a p-variate random vector by minimising the sum of squared orthogonal distances to the line. The first linear principal component approximation goes through the mean of the distribution and is parallel to the eigenvector of the sample covariance matrix, Σ, corresponding to the largest eigenvalue. Principal curves are nonlinear generalisations of the first principal component line and are smooth curves that are used to summarise the data instead of a straight line.

The following example illustrates the notion of a principal curve going through the center of a simulated data set. A sample size of 200 data points is simulated with three variables: X_1 from a uniform distribution $\mathscr{U}(0, 2\pi)$, $X_2 = sin(X_1) + X_1 + U$ and $X_3 = sin(X_2) + U$, where U is simulated from a uniform distribution $\mathscr{U}(-0.4, 0.4)$. The data is simulated to look *S*-shaped. Consider first the two-dimensional case, using X_1 and X_2; the first principal component line is shown in the first panel and the fitted principal curve in the second panel of Figure 2.1. Then in the three-dimensional case, using X_1, X_2 and X_3, the first principal component line is shown in the first panel and the fitted principal curve in the second panel of Figure 2.2.

The principal curve, like linear principal components focuses on the orthogonal or shortest distance to the sample points. Principal curves, are therefore, known to be self-consistent for a distribution or a data set, which essentially means that if any point on the curve is picked, all the data points that project onto this point are collected and then averaged, it will coincide with the point on the curve (Hastie and Stuetzle, 1989). For a more detailed discussion on self-consistency, principal curves and their properties, see Ganey (2014).

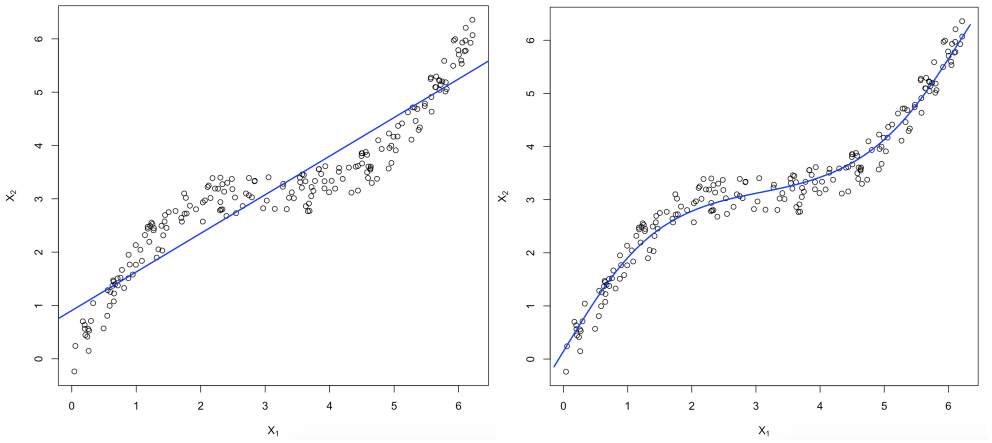

FIGURE 2.1: Illustration of a principal curve in two-dimensional data: variables X_1 and X_2 in the simulated *S*-shaped data. The first principal component line (in blue) is given in the first panel, and the fitted principal curve (in blue) in the second panel.

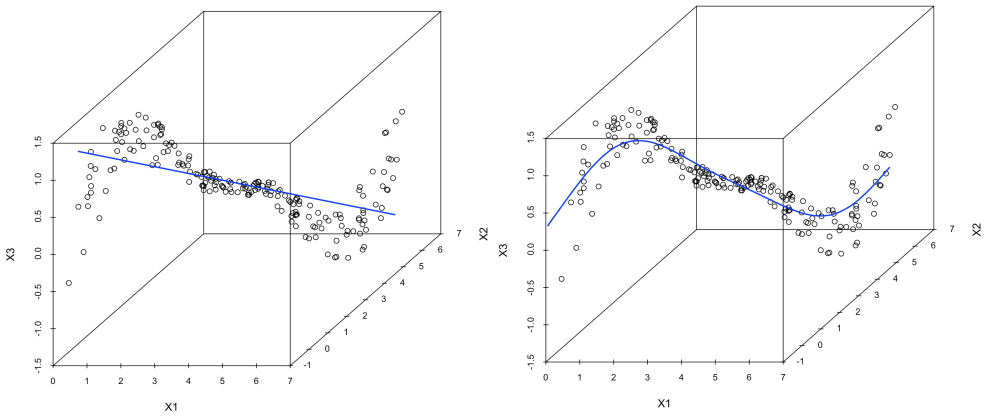

FIGURE 2.2: Illustration of a principal curve in three-dimensional data: variables X_1, X_2 and X_3 in the simulated *S*-shaped data. The first principal component line (in blue) is given in the first panel, and the fitted principal curve (in blue) in the second panel.

Principal surfaces have exactly the same form as principal curves, but are of higher dimension. That is why principal curves are sometimes referred to as one-dimensional principal surfaces. A principal surface is a generalisation of a principal curve to a smooth two- (or higher-) dimensional surface (Hastie, 1984). Two-dimensional surfaces are most commonly used, as dimensions greater than two have less attractive visualisation possibilities.

A principal surface is found using an iterative procedure which starts with a linear summary, typically the plane spanned by the first and second principal components. Each successive iteration is a smooth or local average of the p-dimensional points, where local is based on the projections of the points onto the surface of the previous iteration. The sum of squared distances between the points and the surface in each iteration of the procedure are found, such that this sum is minimised.

2.2 The principal surfaces of a probability distribution

To introduce the theoretical overview of principal surfaces, two-dimensional surfaces in a p-dimensional space will be addressed. A smooth two-dimensional globally parametrised surface f can be defined as p vectors of smooth functions:

$$\mathbf{f} = \begin{pmatrix} f_1(\lambda) \\ f_2(\lambda) \\ \vdots \\ f_p(\lambda) \end{pmatrix} = \begin{pmatrix} f_1(\lambda_1, \lambda_2) \\ f_2(\lambda_1, \lambda_2) \\ \vdots \\ f_p(\lambda_1, \lambda_2) \end{pmatrix}. \tag{2.1}$$

Let $\mathbf{X}$ be a random vector in p-space with a continuous probability density function $h(\mathbf{x})$. Let G^2 be the class of differentiable two-dimensional surfaces in $\mathbb{R}^p$, parameterised by $\lambda = (\lambda_1, \lambda_2) \in \Lambda_f$, a two-dimensional parameter vector.

For $\mathbf{f} \in G^2$ and $\mathbf{x} \in \mathbb{R}^p$, a projection index can be defined as,

$$\lambda_{\mathbf{f}}(\mathbf{x}) = \max_{\lambda_2} \max_{\lambda_1} \{\lambda : ||\mathbf{x} - \mathbf{f}(\lambda)|| = \inf_{\mu} ||\mathbf{x} - \mathbf{f}(\mu)||\}, \tag{2.2}$$

which is the value of λ corresponding to the point on the surface closest to $\mathbf{x}$. The projection index $\lambda_{\mathbf{f}}(\mathbf{x})$ is a measurable mapping from $\mathbb{R}^p$ into $\mathbb{R}^2$, and $\lambda_{\mathbf{f}}(\mathbf{X})$ is a random vector.

The principal surfaces of h are those members of G^2 that hold the self-consistent property:

$$E(X|\lambda_{\mathbf{f}}(\mathbf{x}) = \lambda) = \mathbf{f}(\lambda), \tag{2.3}$$

where each point on a principal surface is the average of the points that project there (Hastie, 1984).

2.2.1 Principal surface algorithm

Principal surfaces of a probability distribution are obtained the same way as principal curves (one-dimensional surfaces), which follows an iterative procedure between a projection and an estimation step. When seeking principal surfaces, the strategy is to start with a smooth surface that corresponds to a local minimum of the distance function.

Let $d(\mathbf{x}, \mathbf{f})$ denote the Euclidean distance from a point $\mathbf{x}$ to its projection on the surface $\mathbf{f}$ and $D^2(\mathbf{f}) = E(d^2(\mathbf{X}, \mathbf{f}))$. The minimisation problem can be written as finding $\mathbf{f}$ and λ such that the following distance function

$$D^2(\mathbf{f}, \lambda) = E||\mathbf{X} - \mathbf{f}(\lambda)||^2 \tag{2.4}$$

is a minimum.

The algorithm is given as follows:

Principal surface algorithm. The following is an iterative algorithm

1. **Initialisation step:**
   Start with an initial surface by setting $\mathbf{f}^{(0)} = \mathbf{A}\lambda$, where $\mathbf{A}$ is a p x 2 matrix

consisting of the first two principal component direction vectors. Set $\lambda^{(0)} = \lambda_{\mathbf{f}^{(0)}}$.

2. **Repeat:** over iteration counter k
   (a) Set $\mathbf{f}^{(k)}(\cdot) = E(\mathbf{X}|\lambda^{(k-1)}(\mathbf{X}) = \cdot)$.
   (b) Choose $\lambda^{(k)} = \lambda_{\mathbf{f}^{(k)}}$.
   (c) Evaluate *Distance* $= D^{2(k)}(\mathbf{f}^{(k)}, \lambda^{(k)})$.
3. **Until:** *Distance* fails to decrease, that is $\frac{|D^{2(k)} - D^{2(k-1)}|}{D^{2(k-1)}}$ is below a certain threshold value, where $D^{2(k)}$ is the distance at the k-th iteration and $D^{2(k-1)}$ the distance at the $(k-1)$-th iteration, computed using equation 2.4.

The algorithm returns the final fitted principal surface $\mathbf{f}(\lambda)$ with a corresponding set of λ coordinates.

2.3 Principal surfaces for data sets

In the previous section, principal surfaces for a continuous multivariate probability distribution were considered. In reality, finite multivariate data sets exist and principal surfaces need to be defined for such cases.

Suppose $\mathbf{X}$ is an n x p matrix of n observations on p variables. The initial step in the algorithm will use the sample principal components as the initial estimates of $\mathbf{f}^{(0)}$:

$$\mathbf{f}^{(0)} = \mathbf{X}_0 \, \mathbf{V}_2 \mathbf{V'}_2 + \mathbf{1}'\bar{\mathbf{X}}', \tag{2.5}$$

where $\mathbf{X}_0$ are the centered values of $\mathbf{X}$, $\mathbf{V}_2$ consists of the first two columns of the right singular vectors of the centered matrix $\mathbf{X}_0$ and $\bar{\mathbf{X}}$ consists of p mean values for the variables in $\mathbf{X}$.

The k-th iteration consists of two steps:

Projection step: Given the current iterate of the surface, $\mathbf{f}^{(k-1)}$ of the principal surface, $\mathbf{x}$ is projected onto the surface to get an updated value of λ:

$$\lambda_{\mathbf{f}^{(k-1)}}(\mathbf{x}) = \lambda^{(k)} = \mathbf{f}^{(k-1)}\vartheta_2, \tag{2.6}$$

where ϑ_2 consists of the first two columns of the right singular vectors of the matrix $\mathbf{f}^{(k-1)}$ (Julian, 2008).

With a finite data set of n observations, the average of the observations that project onto a specific point is typically the mean of a single point. The average is therefore, computed within a small area around each point. For this, a scatterplot smoother is fitted on the surface $\mathbf{f}^{(k-1)}$ with $\lambda^{(k)}$. A locally weighted scatterplot smoothing (lowess) function is typically used as the scatterplot smoother. According to Hastie (1984), this is the fundamental building block in the principal surfaces procedure for data sets.

Expectation step: Given the set $\{\lambda_i^{(k)}\}$ from the projection step, the next iterate of the surface is computed by averaging all those points that project to nearby

points on the surface, thus invoking the self-consistent property:

$$\mathbf{f}^{(k)}(\lambda_i^{(k)}) = E(\mathbf{X}|\lambda_{\mathbf{f}^{(k-1)}}(\mathbf{x}_i) = \lambda_i^{(k)}) \tag{2.7}$$

(Julian, 2008). Given n observations of $\mathbf{x}_i$, the sum of the squared Euclidean distances are estimated by:

$$D^2(\mathbf{f}, \lambda) = E||\mathbf{X} - f(\lambda)||^2 = \sum_{i=1}^{n} ||\mathbf{x}_i - \mathbf{f}(\lambda_{\mathbf{f}}(\mathbf{x}_i))||^2. \tag{2.8}$$

The convergence criterion for the principal surface algorithm is the relative change in $D^2(\mathbf{f}, \lambda)$ going from the $(k-1)$-st iteration to the k-th iteration,

$$\textit{threshold}^{(k)} = \frac{|D^2(\mathbf{f}^{(k-1)}, \lambda) - D^2(\mathbf{f}^{(k)}, \lambda)|}{D^2(\mathbf{f}^{(k-1)}, \lambda)}. \tag{2.9}$$

The algorithm alternates between the projection and expectation step until the *threshold* is reduced below some specified value, such as 0.001. The final iteration will yield results of n tuples for the principal surface $\mathbf{f}$ and its coordinates λ.

Once the algorithm has converged, the final principal surface $\mathbf{f}(\lambda)$, of size n x p, with corresponding λ coordinates, of size n x 2, are returned. Since the λ coordinates are not uniformly spread, a set of additional points on the surface are inferred in Chapter 6. This creates a set of points on the surface that are equidistant to each other which helps present a fuller representation of the principal surface.

2.4 Demonstrations of the algorithm

This section presents the steps in the principal surface algorithm when applied to a set of data. Two sets of data are demonstrated. These data sets will be used to illustrate the theory of biplots based on principal surfaces in later chapters. The half-sphere data is used specifically because of its highly nonlinear structure in the data, and the blanket surface data is used for its simple nature to fluently introduce the concept of the biplots based on principal surfaces.

2.4.1 The half-sphere in the three-dimensional space

The half-sphere data is a simulated data set as demonstrated in a similar example in Hastie (1984). The simulated model consists of three variables, x_1, x_2 and x_3 and its coordinates are given as:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \sin(\theta)\sqrt{1-x_3^2} \\ \cos(\theta)\sqrt{1-x_3^2} \\ 2U-1 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix},$$

for all $x_3 < 0.1$ and where U is simulated from a uniform distribution $\mathscr{U}(0,1)$, and $\theta = \pi(2U-1)$. The vector of errors e are simulated from a uniform distribution $\mathscr{U}(0, 0.5)$. Initially the data is simulated with 200 points to give a full sphere shape of data. With the constraint $x_3 < 0.1$, the data produced forms a half-sphere shape, which turns out to have 141 points. Figure 2.3 shows 141 generated points with 7 randomly chosen points labelled from 1 to 7 in light blue. The 7 points are used

to show the change of their λ values as the principal surface is constructed at different iterations of the algorithm.

The initial stage with the principal component plane and initial plot of λ representing the 141 points are shown in Figure 2.4. The red points represent the 7 points projected onto the surface, with their respective λ_1 and λ_2 values. Figures 2.5 and 2.6 display the surface and the λ plot after the first and final iteration of the algorithm, respectively.

Table 2.1 provides the values of λ_1 and λ_2 of the 7 points at the initial stage, first iteration and final iteration of the principal surface algorithm. The squared distance from the data point to the surface is given by the D^2 value in the table. As the principal surface is formed, the λ and D^2 tend to change. Some points change dramatically while others make minimal changes. Most of data points that lie in the center of the initial λ plot moved further to the edge of the plot as the principal surface was formed. While data points already on the edge of the plot, only made slight changes and remained towards the edge.

The final iteration λ plot in Figure 2.6 shows that the bulk of the points moved to the edge of the plot, almost forming a circular shape. The distances D^2 for a few points tend to decrease but then increase slightly or vice versa as the principal surface is formed. Overall, the sum of D^2 has an overall decrease from the initial surface.

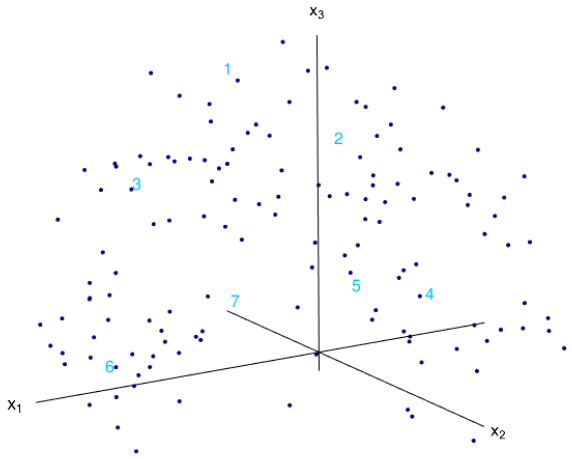

FIGURE 2.3: Simulated half-sphere data.

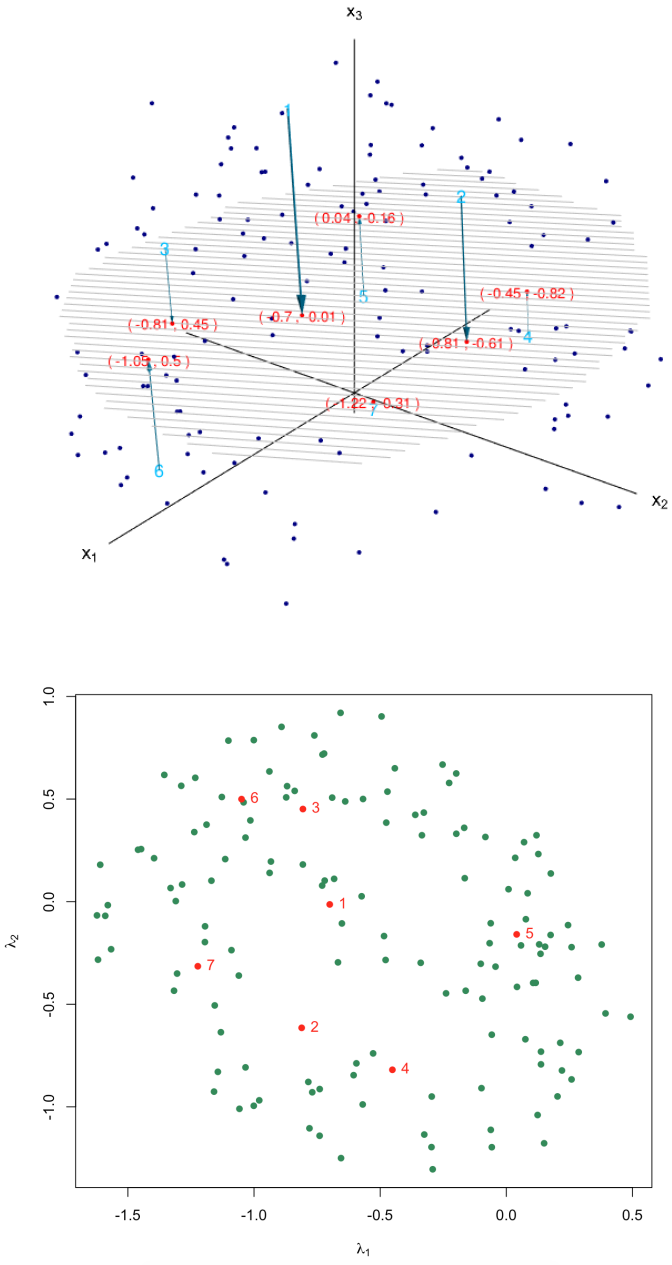

FIGURE 2.4: Simulated half-sphere data: Initial stage in the algorithm.

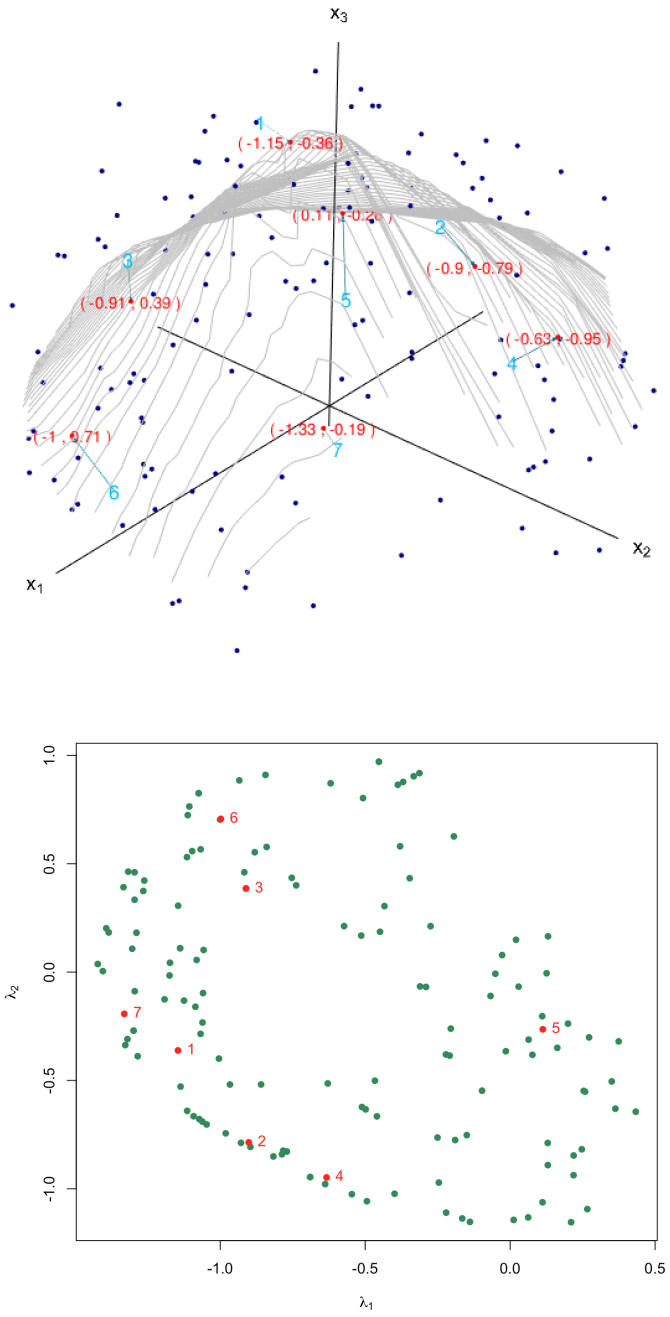

FIGURE 2.5: Simulated half-sphere data: After first iteration in the algorithm.

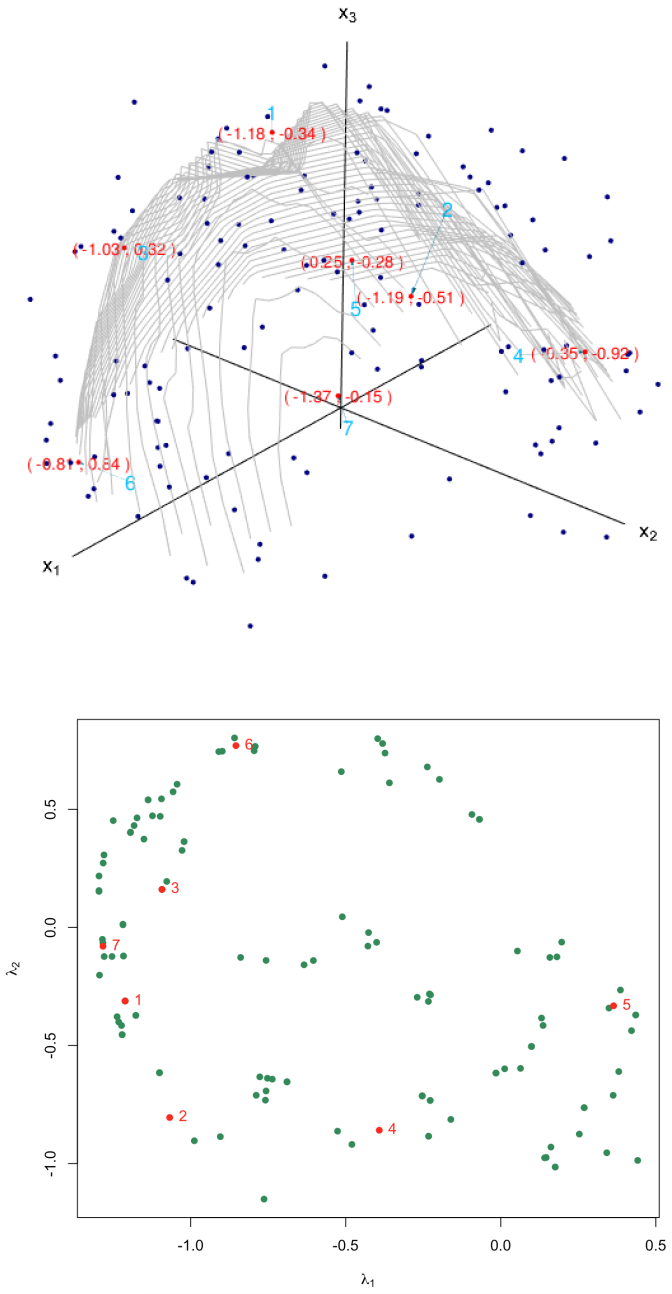

FIGURE 2.6: Simulated half-sphere data: Final iteration in the algorithm.

TABLE 2.1: Values of λ for 7 random points chosen from the half-sphere data at different stages of iteration in the principal surface algorithm. The squared distance from the data point to the surface in each iteration is given by D^2.

	Initial			1st Iteration			Final Iteration		
	λ_1	λ_2	D^2	λ_1	λ_2	D^2	λ_1	λ_2	D^2
1	-0.70	-0.01	0.691	-1.15	-0.36	0.019	-1.18	-0.34	0.017
2	-0.81	-0.61	0.337	-0.90	-0.79	0.099	-1.19	-0.51	0.104
3	-0.81	0.45	0.099	-0.91	0.39	0.028	-1.02	0.32	0.007
4	-0.45	-0.82	0.034	-0.63	-0.95	0.101	-0.34	-0.91	0.055
5	0.04	-0.16	0.127	0.11	-0.26	0.106	0.25	-0.27	0.066
6	-1.05	0.50	0.214	-1.00	0.71	0.096	-0.81	0.83	0.038
7	-1.22	-0.31	0.001	-1.33	-0.19	0.022	-1.37	-0.15	0.047

2.4.2 Blanket surface data

The blanket surface data is also a simulated data set. The data consists of three variables, x_1, x_2 and x_3 and are given as follows:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 2U-1 \\ \sin(\theta)\sqrt{1-x^3} \\ \sin(\theta)\sqrt{x^2} \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix},$$

where the errors e_1, e_2 and e_3 come from a uniform distribution $\mathscr{U}(0, 0.5)$, U from a uniform distribution $\mathscr{U}(0, 1)$ and $\theta = \pi(2U - 1)$. Figure 2.7 shows 100 simulated data points with 7 randomly selected points labelled in light blue. Figures 2.8, 2.9 and 2.10 display the principal surface fitted with its λ coordinates for the 7 random points at different stages of the principal surface algorithm, similarly to the half-sphere data.

Table 2.2 gives the λ coordinates of the 7 points and their squared distance D^2 to the surface at the different stages of the algorithm. One can notice no significant change in the λ values and D^2 as the principal surface is formed. There is also no change in the configuration of the points that lie in the λ plot. This could be due to the principal surface fitted that does not progress much from the initial surface (i.e. principal component plane) in terms of linearity. It also explains the change in the configuration of λ in the half-sphere data, as the principal surface fitted becomes highly nonlinear.

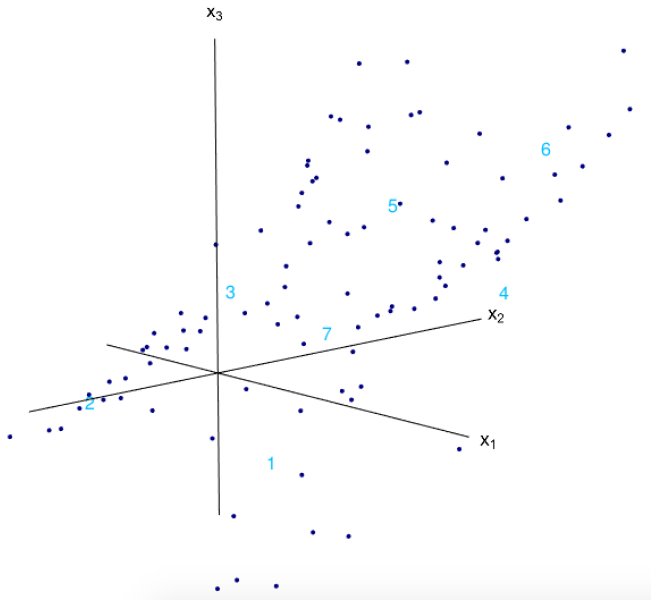

FIGURE 2.7: Simulated blanket surface data.

TABLE 2.2: Values of λ for 7 random points chosen from the blanket surface data at different stages of iteration in the principal surface algorithm. The distance from the data point to the surface in each iteration is given by D^2.

	Initial			1st Iteration			Final Iteration		
	λ_1	λ_2	D^2	λ_1	λ_2	D^2	λ_1	λ_2	D^2
1	-0.33	-0.97	0.0004	-0.31	-0.91	0.0106	-0.32	-0.91	0.0092
2	-0.49	0.13	0.0092	-0.49	0.14	0.0019	-0.51	0.14	0.0024
3	0.43	0.54	0.0081	0.43	0.55	0.0074	0.46	0.56	0.0082
4	0.81	-1.09	0.0001	0.85	-0.99	0.0208	0.80	-0.97	0.0251
5	1.23	0.34	0.0182	1.22	0.36	0.0001	1.22	0.35	0.0000
6	1.42	-0.79	0.0471	1.47	-0.81	0.0049	1.44	-0.84	0.0082
7	0.45	-0.18	0.0269	0.45	-0.17	0.0110	0.47	-0.19	0.0109

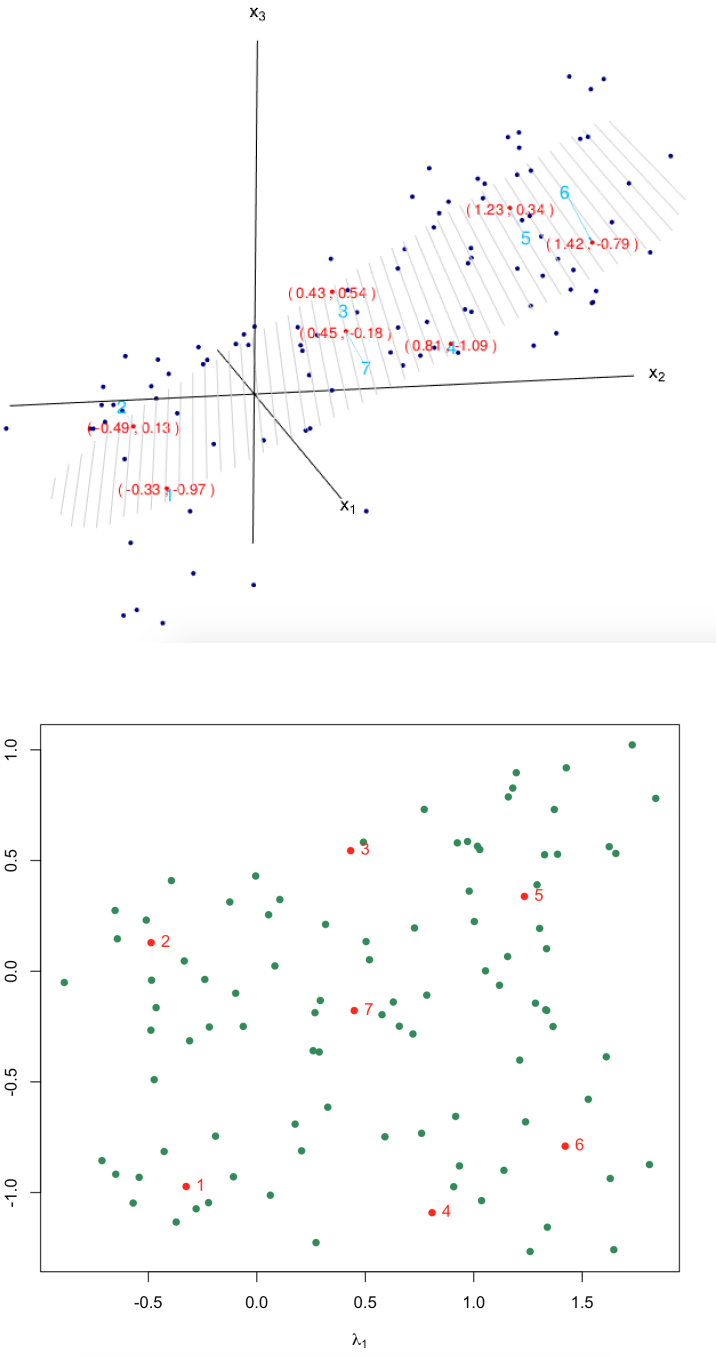

FIGURE 2.8: Simulated blanket surface data: Initial stage in the algorithm.

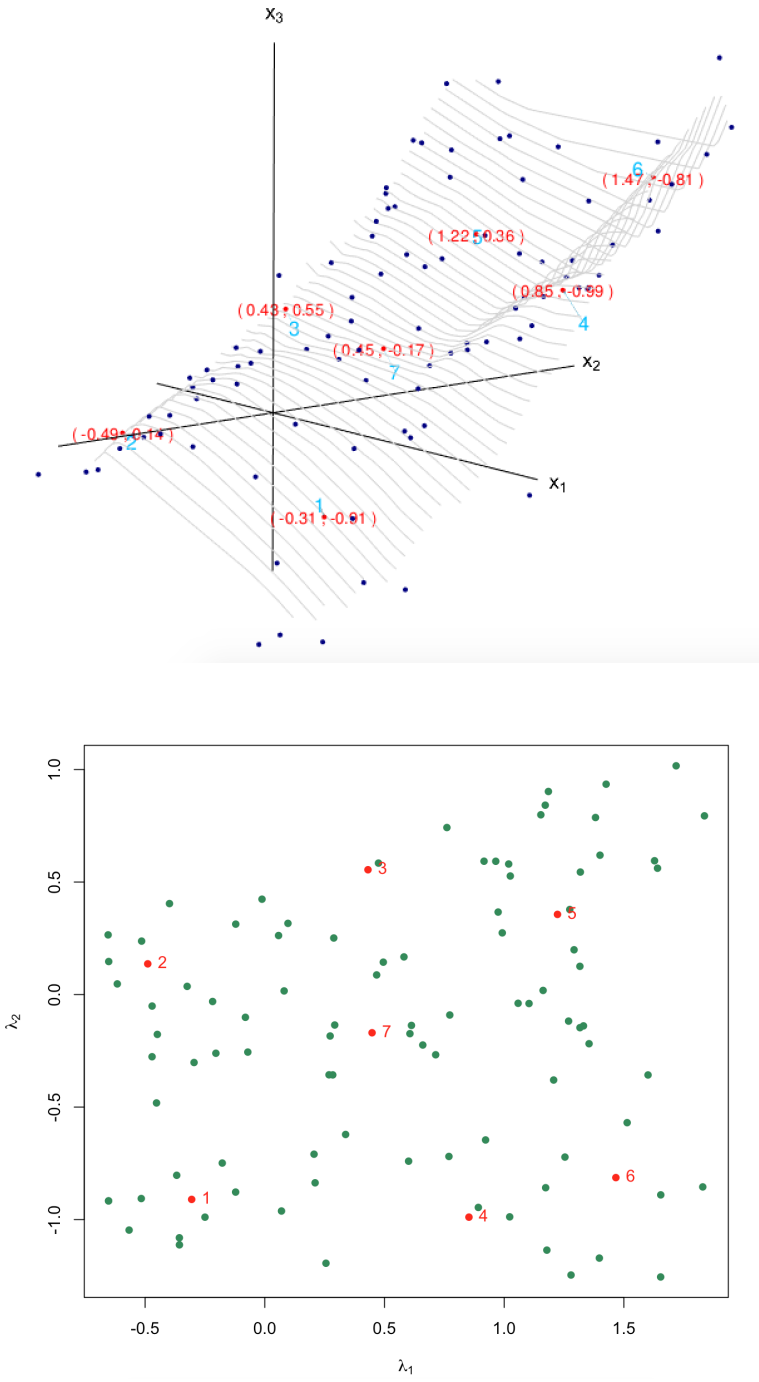

FIGURE 2.9: Simulated blanket surface data: After first iteration in the algorithm.

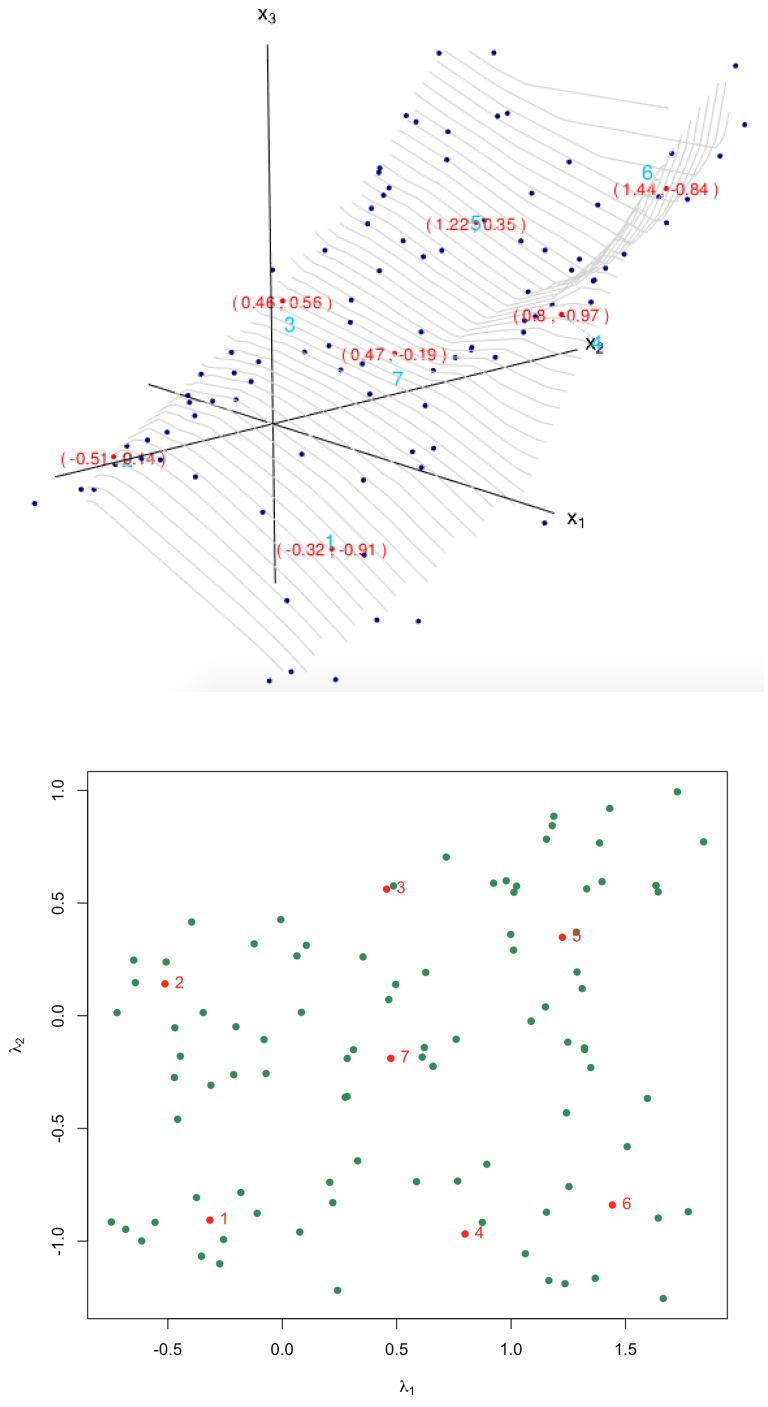

FIGURE 2.10: Simulated blanket surface data: Final iteration in the algorithm.

2.5 Summary

A principal surface is a two-dimensional nonlinear surface that summarises the variability in a higher dimensional data set. Other than being a useful data exploratory tool to identify structures in the data and identifying outliers as pointed out in Hastie (1984), its main attraction for the purposes of this thesis, is that it can approximate data more closely than other common methods such as principal component analysis (Pearson, 1901). In Chapters 3 and 4, it will be seen that biplots are also identified typically in a two-dimensional space and the thesis therefore, investigates whether principal surfaces can be used to construct biplots.

The start of the contributed work of this thesis, appearing in Chapter 6, will use the λ coordinates from principal surfaces that provides a two-dimensional summary of the original data. This plays an important role in the concept of biplots based on principal surfaces.

Chapter 3

Biplots

This chapter introduces the literature of biplot methodology. The purpose is to present the basic understanding of how biplots are constructed. The simplest type of biplot is called the principal component analysis biplot. The chapter illustrates in detail with an example how samples and variables are represented in a biplot.

3.1 Introduction

Consider an n x p data matrix containing numerical information on p variables for each of n samples. The n samples can be represented in a two-dimensional scatter diagram, where the coordinate axes can represent any two variables. Samples can also be represented in a three-dimensional diagram developed from Cartesian geometry and conceptually in multidimensional forms in which there are many coordinate axes to represent the n samples.

Although two-dimensional scatter diagrams are crucial for presenting data, multidimensional scatter diagrams tend not to be. Statisticians therefore, developed methods for approximating multidimensional scatter diagrams into a biplot. A biplot is a generalisation of a two-dimensional scatter diagram of data that exists in a higher dimensional space, where information on both samples and variables can be displayed graphically. Gabriel (1971) introduced the biplot in which the samples are represented as n points and the variables are represented as p vectors relative to the same axes and origin.

Prior to the period of the biplot, a landmark paper, *The Future of Data Analysis* (Tukey, 1962), issued a call for the recognition of data analysis, which included effective graphical displays of higher dimensional data under the rubric of "Exploratory Data Analysis". This included stem-leaf plots, boxplots, hanging rootograms, two-way table displays, and so forth. There are various attempts in the literature to visualise a data matrix consisting of n samples and p variables, and an overview of this is given in (Friendly, 2008).

Biplots makes information in a data matrix become functional by revealing the main structures in the data, for example patterns of correlations between variables or similarities between samples. Biplots can be distinguished by two main types:

- Asymmetric biplots where information on sample units and variables of a data matrix is given.
- Symmetric biplots where information on rows and columns of a two-way table is given. The data used in symmetric biplots are contingency tables of two categorical variables, but can be extended to frequency tables and ratio-scaled data.

In a symmetric biplot, rows and columns can be used interchangeably without any loss of information. A typical symmetric biplot is a correspondence analysis biplot covered extensively in Greenacre (2010). In asymmetric biplots, variables and sample units are different kinds of objects that may not be interchanged. For the purpose related to this thesis, only asymmetric biplots will be considered.

An asymmetric biplot can represent the distances between samples, relationships between variables and the inner products between samples and variables. However, an optimal representation in a biplot can only contain two of these characteristics. Where the biplot specifically refers to the simultaneous representation of the samples and variables, a plot with either only information on the observations or only on the variables can be constructed is referred to as a monoplot.

3.2 Constructing a biplot

There are considerations to be made when constructing a biplot, which include the following:

1. the types of variables (quantitative, qualitative, ordinal, etc.);
2. the method used for displaying samples;
3. what the biplot will display, either for predicting a sample point or interpolating a new sample point. The difference is explained in Section 3.5.

In this thesis, the focus will be on (1) continuous data, (2) the method used to display samples through approximations of the data, and (3) predicting samples instead of interpolating new sample points.

Constructing a biplot requires for a data matrix $\mathbf{X}$: n x p to be approximated. There are many ways of representing sample points in two dimensions such that their actual inter-point distances in the p dimensions are approximated. One of the simplest methods of approximating the data is done in terms of its singular value decomposition (SVD) (Gabriel, 1971). The way of proceeding with the SVD is,

$$\mathbf{X} : nxp = \mathbf{U}^*\mathbf{\Sigma}^*(\mathbf{V}^*)', \tag{3.1}$$

where, assuming $n \geq p$, $\mathbf{U}^*$ is an n x n orthogonal matrix with columns known as the left singular vectors of $\mathbf{X}$, the matrix $\mathbf{V}^*$ is a p x p orthogonal matrix with columns known as the right singular vectors of $\mathbf{X}$, while the matrix $\mathbf{\Sigma}^*$ is of the form

$$\mathbf{\Sigma}^* : nxp = \begin{bmatrix} \mathbf{\Sigma} & 0 \\ 0 & 0 \end{bmatrix}, \tag{3.2}$$

where $\mathbf{\Sigma}$ is a q x q diagonal matrix and q denotes the rank of $\mathbf{X}$. The diagonal elements are the nonzero singular values of $\mathbf{X}$, and are assumed to be presented in non-increasing order. It follows that equation 3.1 can be written as

$$\mathbf{X} : nxp = \mathbf{U}\mathbf{\Sigma}\mathbf{V}', \tag{3.3}$$

where $\mathbf{U}$: n x q and $\mathbf{V}$: p x q consist of the first q columns of $\mathbf{U}^*$ and $\mathbf{V}^*$ respectively. The matrices $\mathbf{U}$ and $\mathbf{V}$ are both orthonormal.

The r-dimensional approximation also known as the Eckart-Young approximation, of $\mathbf{X}$ is given by:

$$\hat{\mathbf{X}}_{[r]} = \mathbf{U}\boldsymbol{\Sigma}_{[r]}\mathbf{V}', \tag{3.4}$$

where $\boldsymbol{\Sigma}_{[r]}$ replaces the $p - r$ smallest diagonal values of $\boldsymbol{\Sigma}$ by zero. The Eckart-Young approximation is based on minimising a least-squares criterion, discussed further in the next section.

There are many other ways of representing data points in fewer dimensions other than the SVD approach such that their actual inter-point distances are approximated. These methods lie in the area of multidimensional scaling (MDS). Further chapters will look at these methods. The next section looks at a type of biplot that uses the Eckart-Young approximation, and how samples and variables are represented in the biplot.

3.3 Principal component analysis biplot

The simplest type of biplot is the principal component analysis (PCA) biplot. The approaches to PCA date back to two separate views given by Pearson (1901) and Hotelling (1933), who independently arrived at PCA. While Pearson (1901) searched to find the straight line or hyperplane which is best fitting to a higher dimensional configuration of points, Hotelling (1933) aimed to summarise the total sample variance associated with the set of measured variables by means of a few uncorrelated linear combinations of the measured variables (Brand, 2013). PCA, in this chapter, is viewed from Pearson's perspective and is therefore, directed at the fundamental problem of approximating $\mathbf{X}$ by $\hat{\mathbf{X}}_{[r]}$.

Geometrically the rows of $\mathbf{X}$ are given as coordinates of n samples in the p-dimensional space $\mathbb{R}^p$. The distance d_{ij} between pairs of points is given by Euclidean distances. The aim is to seek an r-dimensional plane that contains the points whose coordinates are given by the rows of $\hat{\mathbf{X}}_{[r]}$ which minimises a least squares criterion given by,

$$||\mathbf{X} - \hat{\mathbf{X}}_{[r]}||^2 = tr\{(\mathbf{X} - \hat{\mathbf{X}}_{[r]})(\mathbf{X} - \hat{\mathbf{X}}_{[r]})'\}. \tag{3.5}$$

The plane that minimises the least-squares criterion passes through the centroid of points and its coordinates are an orthogonal projection of $\mathbf{X}$. The best approximation that minimises the least squares criterion is the r-dimensional Eckart-Young approximation given by $\hat{\mathbf{X}}_{[r]} = \mathbf{U}\boldsymbol{\Sigma}_{[r]}\mathbf{V}'$.

In the biplot, the rows (n samples) and columns (p variables) of the data matrix $\mathbf{X}$ are approximated by the rows and columns of $\hat{\mathbf{X}}_{[r]}$. A standard result when $r = 2$ from Gower and Hand (1996) is that the row vectors of $\hat{\mathbf{X}}_{[2]}$ are the orthogonal projections of the corresponding row vectors of $\mathbf{X}$ onto the column space of $\mathbf{V}_2$. The projections are therefore, given by

$$\mathbf{X}\mathbf{V}_2. \tag{3.6}$$

The projections in equation 3.6 are points expressed in terms of the coordinates of the original p dimensions. A scatterplot can be constructed of the two-dimensional approximation of $\mathbf{X}$ by plotting the samples as the rows of $\mathbf{XV}_2$.

The next step to construct a PCA biplot, is to represent the columns (variables) of the data matrix $\mathbf{X}$. Let the rank of $\mathbf{X}$ be denoted by q, $q \leq p \leq n$. The construction of the biplot is based on the fact that any n x p matrix of rank q can be expressed as the inner product of an n x q matrix of rank q and a q x p matrix of rank q. It follows that $\mathbf{X}$ can be written as $\mathbf{X} = \mathbf{AB}'$, where $\mathbf{A}$ is an n x q matrix of rank q and $\mathbf{B}$ is a p x q matrix of rank q, then each element of $\mathbf{X}$ is given by

$$x_{ij} = \mathbf{a}_i'\mathbf{b}_j, \tag{3.7}$$

which is inner product of the rows of $\mathbf{A}$ and columns of $\mathbf{B}$.

When $q = r$, $\mathbf{X}$ can be expressed as the inner product of an n x r matrix of rank r, and a r x p matrix of rank r. From equation 3.3, $\mathbf{X} = \mathbf{U\Sigma V}'$, which implies that $\mathbf{XV} = \mathbf{U\Sigma}$. Since the approximation of the rows of $\mathbf{X}$, when $r = 2$, is $\mathbf{XV}_2$, set $\mathbf{A} = \mathbf{U\Sigma}$ and it follows that $\mathbf{B}' = \mathbf{V}'$. Therefore, the columns of $\mathbf{X}$ are approximated by the first two rows of $\mathbf{V}$, which now represent the axes for each variable.

Gabriel (1971) shows that this leads to a biplot where the inner products in equation 3.7 and the distances between the samples are optimally represented. Gabriel (1971) further shows that selecting $\mathbf{A} = \mathbf{U}$ and $\mathbf{B} = \mathbf{V\Sigma}$ optimally represents the correlations between the variables rather than the distance between samples. The focus will be on the former representation.

3.3.1 PCA biplot illustration

Consider the three-dimensional data set given in the first three columns of Table 3.1. The data set is simulated similar to that used in Gower, Lubbe, and Le Roux (2011) to illustrate the construction of a PCA biplot. The data is represented graphically in Figure 3.1. To represent Figure 3.1 optimally in two dimensions, a plane passing through the centroid must be found that minimises the sum of squares of the distances from the plane i.e. the approximation $\hat{\mathbf{X}}_{[r]}$, where $r = 2$. This plane is called the biplot plane and is denoted by $\mathscr{L}$. The biplot plane in the three-dimensional space with the orthogonal projections of the original points onto the plane can be seen in Figure 3.2. The first two principal components (the columns $\mathbf{V}_2$) span the two-dimensional biplot space and the sum of squared distances represented by the arrows is minimised.

To represent the blue projected samples on the biplot plane in Figure 3.2 in a standard x-y plot, the first two columns of $\mathbf{XV}$ are plotted. This is shown in Figure 3.3 where the points are labelled according to their sample number in Table 3.1. The variables X_1, X_2 and X_3 are represented by the first two rows of $\mathbf{V}$ in the form of vectors. This is illustrated in Figure 3.4.

TABLE 3.1: Three-dimensional example data set

	Actual			Predicted		
Sample no.	X_1	X_2	X_3	X_1	X_2	X_3
1	5.4188	5.0542	8.7112	6.2213	5.216	8.2019
2	3.1299	1.7832	3.3859	3.6338	1.8848	3.0661
3	6.1281	2.1732	8.1736	6.6053	2.2694	7.8707
4	6.7811	4.7533	8.7316	6.6908	4.7351	8.7889
5	7.3466	5.8932	11.3030	7.8358	5.9919	10.9925
6	7.2082	3.7440	10.0758	7.5540	3.8137	9.8563
7	7.0394	5.2136	8.6088	6.6519	5.1355	8.8548
8	5.4657	4.4926	5.5965	4.9448	4.3876	5.9271
9	7.7232	4.7081	11.3575	8.1410	4.7924	11.0923
10	7.1096	4.9875	7.7328	6.3199	4.8283	8.2340
11	8.1358	4.3920	9.2648	7.3919	4.2420	9.7369
12	6.2875	5.9087	7.4882	5.8311	5.8167	7.7779
13	4.6489	7.1983	8.5737	5.6249	7.3951	7.9544
14	5.7986	6.1201	8.2548	5.9885	6.1584	8.1344
15	8.0846	3.2348	8.9666	7.4066	3.0981	9.3967
16	6.1578	5.7435	9.8990	6.8743	5.8879	9.4443
17	3.5567	2.0263	3.8476	3.9340	2.1024	3.6082
18	6.9637	2.4695	9.2881	7.3106	2.5395	9.0680
19	7.7058	5.4015	9.9223	7.4078	5.3414	10.1114
20	8.3484	6.6968	12.8444	8.7090	6.7695	12.6155
21	8.1911	4.2545	11.4497	8.3887	4.2943	11.324
22	7.9994	5.9246	9.7828	7.3636	5.7964	10.1862
23	6.2110	5.1053	6.3597	5.4237	4.9465	6.8593
24	8.7764	5.3501	12.9062	9.0558	5.4065	12.7289
25	8.0790	5.6676	8.7873	6.9863	5.4473	9.4808

3.4 Calibrated biplot axes

Although in the previous example, Figure 3.4 is a biplot, the variables represented by the vectors (arrows) have no calibration. That meaning, there are no markers on the vectors representing the variables analogous to ordinary scatterplots. The scaffolding axes in Figure 3.4 represent the principal components which are the first columns of the matrix $\mathbf{V}$. This section addresses this problem of converting the vectors into calibrated axes.

The biplot axes in Figure 3.4 are shown as vectors $\mathbf{v}_k$ whose end points V_k have coordinates given by the first two elements of the k-th row of $\mathbf{V}$. To illustrate the construction of a calibrated biplot axes, the first variable, X_1 of the previous example in Table 3.1 will be used.

To construct a biplot axis with relevant markers for a variable, a $(p-1)$-dimensional hyperplane $\mathscr{N}$ perpendicular to the Cartesian axis, X_1 is required. From Table 3.1, $p = 3$ therefore, a two-dimensional hyperplane is constructed perpendicular to X_1 through a specific value of X_1, say μ. The intersection of $\mathscr{L}$ and $\mathscr{N}$ is an $(r-1)$-dimensional intersection space, which in this case will be indicated by a line. All the points on this intersection line in $\mathscr{L}$ will predict the value for μ for the X_1-axis.

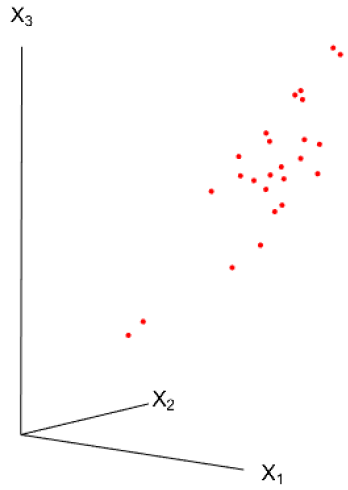

FIGURE 3.1: Three-dimensional representation of the data from Table 3.1.

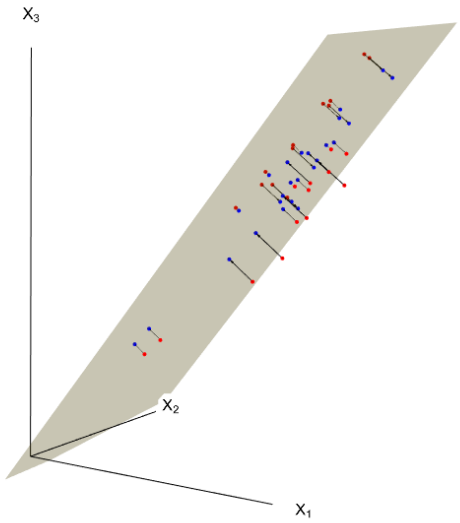

FIGURE 3.2: Orthogonal projection of three-dimensional sample points onto the two-dimensional biplot space $\mathscr{L}$. Red points in full space are projected orthogonally onto the blue points in $\mathscr{L}$ such that the sum of the squared distances ϵ (indicated by the black arrows) is minimised.

This is illustrated in Figure 3.5, where $\mu = 4$.

The plane $\mathscr{N}$ is shifted orthogonally through another value on X_1 and all the points on the intersection line of $\mathscr{L}$ and $\mathscr{N}$ will predict that value that the plane goes through. This is shown in Figure 3.6, where $\mu = 6$.

As the plane $\mathscr{N}$ is shifted along the X_1-axis, a series of parallel intersection spaces

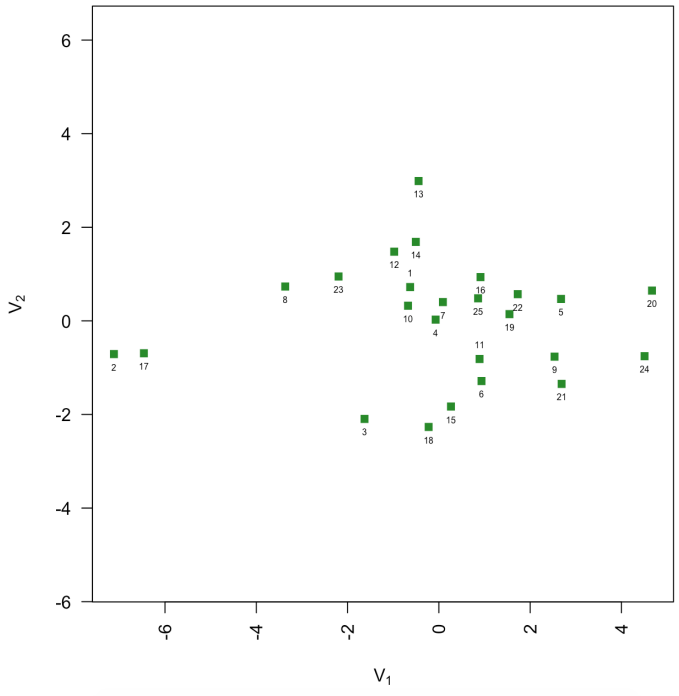

FIGURE 3.3: PCA display of the sample points of Table 3.1.

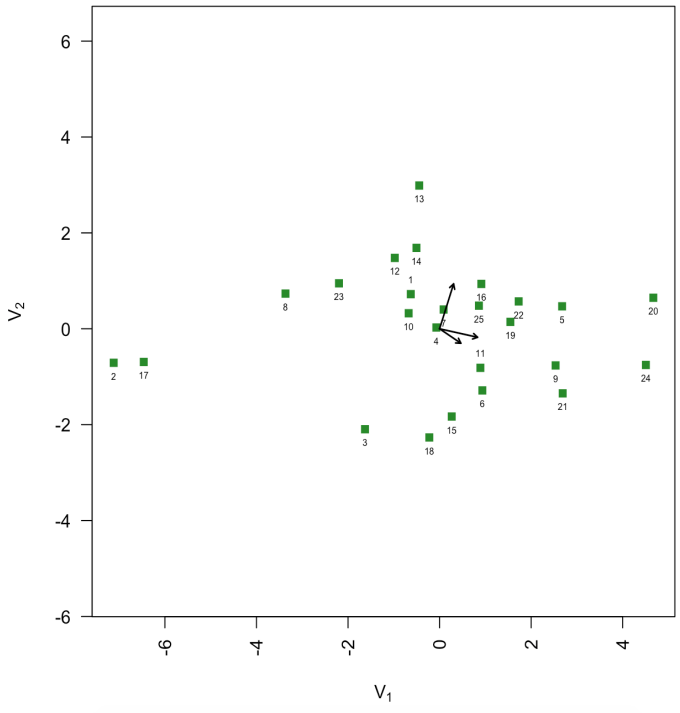

FIGURE 3.4: PCA display of the sample points of Table 3.1 with scaffolding axes, V_1 and V_2 representing principal components, and vectors representing variables.

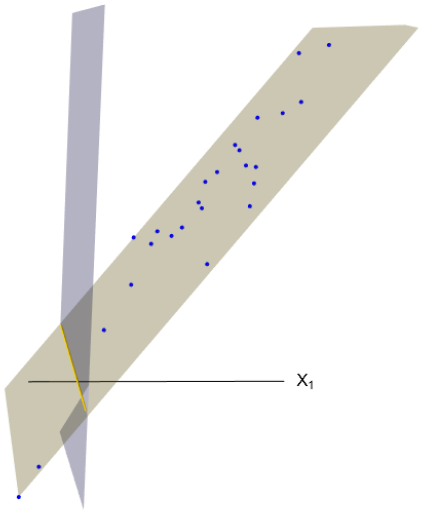

FIGURE 3.5: Plane $\mathcal{N}$ constructed orthogonal to Cartesian axis X_1 through the marker "4". Intersection line is represented by the yellow line.

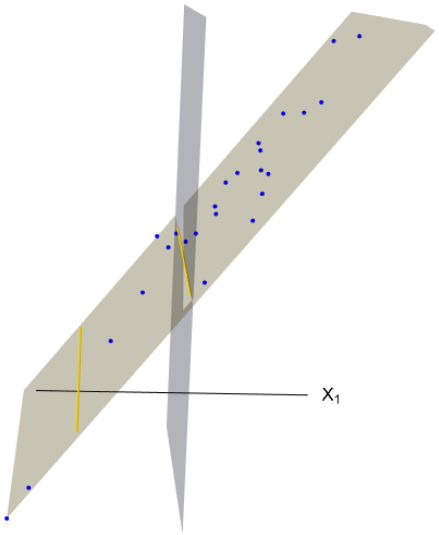

FIGURE 3.6: Plane $\mathcal{N}$ constructed orthogonal to Cartesian axis X_1 through the marker "6".

is obtained, as shown in Figure 3.7. Any line passing through the origin will pass through these intersection spaces and can be used as an axis fitted with markers according to the value associated with the particular intersection space. To facilitate orthogonal projection onto the axis, similar to an ordinary scatterplot, the line orthogonal to these intersection spaces is chosen.

For the data in Table 3.1 and for variables X_1, X_2 and X_3 the corresponding axes are calibrated the same way and the resulting PCA biplot is shown in Figure 3.8.

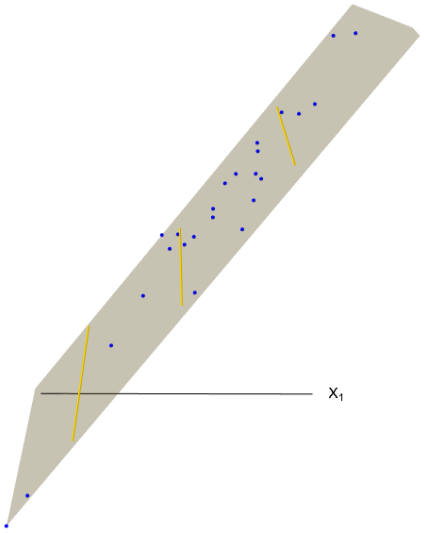

FIGURE 3.7: Parallel intersection spaces for predicting the values of the Cartesian axis X_1.

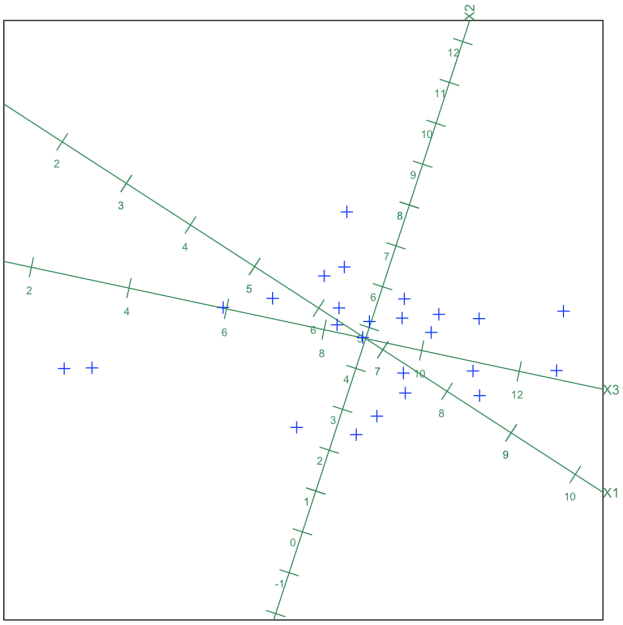

FIGURE 3.8: PCA biplot of data in Table 3.1, where the samples and variables are both represented. Each axes is calibrated with marker points.

3.5 Prediction biplot axes

The concern thus far has been on the visualisation of n samples and p variables of a data matrix in a biplot. In addition to the aspect of visualisation, the aim is to use a biplot analogous to a two-dimensional scatterplot, where the biplot axes are

used to read off values of any variable for any sample in **X**. This process is called prediction.

The alternative process to prediction is interpolation. In this case the axes are utilised to place a new sample point by vector addition in the display. Unlike ordinary two-dimensional scatterplots where a single set of calibrated orthogonal axes are used for both interpolation and prediction; the biplot axes need to be constructed differently for prediction and interpolation. Therefore, a single set of nonorthogonal axes for both interpolation and prediction will result in inconsistent representation of the same sample point.

The calibration of axes in the previous section have been equipped as prediction axes and for the purposes of this thesis, only the process of prediction will be considered. In order to predict a sample point in the data using the biplot, the sample point is orthogonally projected onto each calibrated axis, and read off using the markers of that axis to give a predicted value for each variable. To illustrate this, consider the sample points in Figure 3.9. Prediction of all sample points are given in the last three columns of Table 3.1.

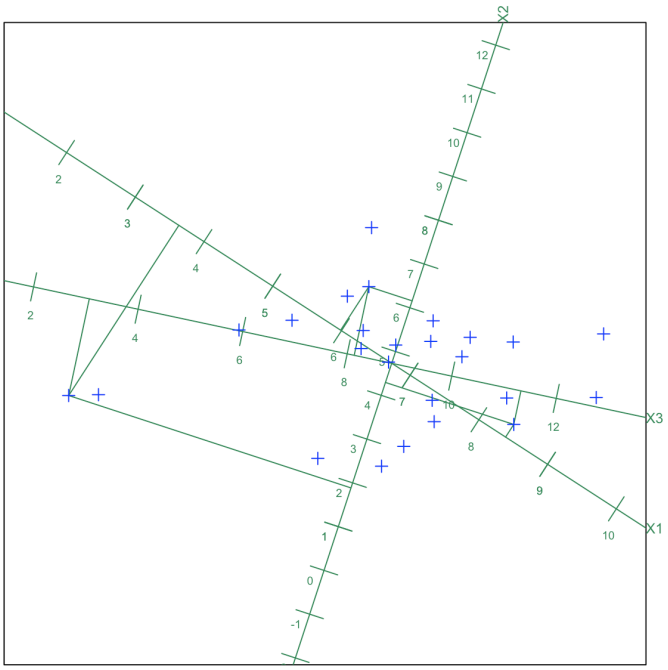

FIGURE 3.9: Prediction PCA biplot of the data in Table 3.1. Lines show predictions for samples for **2**, **14** and **21**.

3.6 Summary

PCA is discussed in this chapter as a method of approximating a data matrix, a technique in multivariate linear dimension reduction. The PCA biplot is seen as an extension of an ordinary scatterplot, detailing the construction and further the prediction of samples. There is literature on the understanding of PCA biplots, and the further extensions it can provide. Some of which include rotation and reflection of biplots, interpolation of new sample points, the quality of predictions

for each axis called axis predictivity and adequacy, three-dimensional PCA biplots and more. For a detailed discussion, see Gower, Lubbe, and Le Roux (2011).

The PCA biplots in this chapter were constructed with the help of R (R Core Team, 2018) functions provided in Gower, Lubbe, and Le Roux (2011).

Chapter 4

Nonlinear Biplots

4.1 Introduction

In Chapter 3 PCA is discussed as the lower rank approximation of a rank-q n x p data matrix. The PCA biplot is the two-dimensional scatterplot approximation for visualising a p-dimensional data set. It was noted that the PCA biplots constructed, optimally approximate the distances between samples. This statement implied the optimal representation of the Euclidean distances between samples as Euclidean distance is the most natural intuitive quantification of distance.

The nonlinear biplot (Gower and Harding, 1988) is a generalisation of the PCA biplot introduced in Chapter 3 when distances other than Euclidean distance is used. Multidimensional scaling (MDS) (Borg and Groenen; Cox and Cox, 2005; 2001) attempt to find points in a low-dimensional Euclidean space that generate distances that approximate the inter-sample distances given in an n x n symmetric matrix. One of the MDS methods, principal coordinate analysis, will be mentioned in this chapter. PCA can be treated as a matrix approximation, which is concerned with finding a set of points in a few dimensions whose inter-sample distances generate approximations to the Euclidean distances between the samples. Therefore, it can be brought into the scope of approximating distances as in MDS, where the distance metric associated with PCA is the Euclidean distance metric.

There are many other ways to quantify distances between samples. For a non-exhaustive list of distance metrics, see Cox and Cox (2001). The specific definitions and differences between distances, dissimilarities and the like will be discussed in Section 4.2. Suffice to say that different measures of distances between samples can be computed.

The following sections describe how the samples are represented in a biplot using the approximation of distances, and how the variables are then constructed in the Euclidean embedded space. Lastly, details on how prediction biplot axes are constructed in the biplot space to predict new samples, are discussed.

4.2 Distance metrics

Let $\mathbf{X}$ be an n x p matrix of n samples with observations on p variables. Let d_{ij} denote the distance between samples $\mathbf{x}_i$ and $\mathbf{x}_j$, creating an n x n distance matrix. A metric is a distance measure if:

1. $d_{ij} = 0$ if and only if $i = j$;
2. $d_{ij} = d_{ji}$ for all i, j;
3. $d_{ij} \leq d_{ik} + d_{kj}$ for all i, j, k known as the triangle inequality.

When the distances in the matrix $\mathbf{D} = \{-\frac{1}{2}d_{ij}^2\}$ can be generated as Euclidean distances by a set of n points in $n-1$ dimensions, the distance is said to be *Euclidean embeddable*. The distance between two points approximates a function of d_{ij}. Other Euclidean embeddable distance measures exist, namely among others the square root of the Manhattan distance discussed in Gower and Legendre (1986) and the Clark's distance discussed in Gower and Ngouenet (2005) and originally in Clark (1952). Note that the Manhattan distance is not Euclidean embeddable, but the square root of the distance is.

The square root of the Manhattan distance is given as

$$d_{ij} = \sqrt{\sum_{k=1}^{p} |x_{ik} - x_{jk}|}, \tag{4.1}$$

whereas the Clark's distance defined for nonnegative values x_{ik}, x_{jk} is given by

$$d_{ij} = \sqrt{\sum_{k=1}^{p} \left(\frac{x_{ik} - x_{jk}}{x_{ik} + x_{jk}}\right)^2}. \tag{4.2}$$

4.3 Representation of samples

In order to represent the samples of $\mathbf{X}$, the matrix $\mathbf{D} = \{-\frac{1}{2}d_{ij}^2\}$ is used to define the double centered matrix

$$\mathbf{B} = (\mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}')\mathbf{D}(\mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}'). \tag{4.3}$$

A consequence of Euclidean embeddability is that $\mathbf{B}$ is a positive semi-definite matrix, Vines (2015). A necessary and sufficient condition for Euclidean embeddability is that $\mathbf{B} = (\mathbf{I} - \mathbf{1}\mathbf{s}')\mathbf{D}(\mathbf{I} - \mathbf{s}\mathbf{1}')$ be positive semi-definite for any $\mathbf{s}$ such that $\mathbf{s}'\mathbf{1}$ and $\mathbf{s}'\mathbf{D} \neq \mathbf{0}'$. The proof of this statement is given in Gower (1982). A simple choice for $\mathbf{s}$ can be $\frac{1}{n}, \ldots, \frac{1}{n}$, but any other set of weights can be used.

Through the spectral decomposition of $\mathbf{B} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}'$, it is possible to find a matrix $\mathbf{Y}$ such that $\mathbf{B} = \mathbf{Y}\mathbf{Y}'$. The rows of $\mathbf{Y}$ provide coordinates of the samples that generate the distances d_{ij} and therefore, a sufficient and necessary condition for Euclidean embeddability is that $\mathbf{B}$ be positive semi-definite.

The matrix $\mathbf{Y}$ is given by the eigenvectors satisfying $\mathbf{B}\mathbf{Y} = \mathbf{Y}\mathbf{\Lambda}$, scaled so that $\mathbf{Y}'\mathbf{Y} = \mathbf{\Lambda}$. This means that $\mathbf{B} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}'$ provides $\mathbf{Y} = \mathbf{V}\mathbf{\Lambda}^{1/2}$. Let $\mathbf{y}_i'$ be the ith row of $\mathbf{Y}$, then $\mathbf{y}_i'$ can be regarded as the location of the ith sample in an m-dimensional space, where m is generally defined as $n-1$; such that for $j = 1, \ldots, n$, the distance between $\mathbf{y}_i$ and $\mathbf{y}_j$ matches the Euclidean distance between $\mathbf{x}_i$ and $\mathbf{x}_j$.

Since, $m = n - 1$, it means that the exact correspondence between inter-point distances and dissimilarities cannot normally be directly plotted in a low dimensional plot. However, since $\mathbf{Y}$ is already referred to its principal axes, the best rank r approximation of $\mathbf{B}$ is obtained taking the first r columns of $\mathbf{Y}$ as the position of the samples.

To illustrate the representation of samples, consider the following example similar to one in Gower, Lubbe, and Le Roux (2011). Consider the data matrix

$$\mathbf{X} = \begin{bmatrix} a: & 5 & 3 \\ b: & 3 & 7 \\ c: & 3 & 1 \\ d: & 1 & 1 \end{bmatrix} \tag{4.4}$$

with two variables X_1 and X_2. The Clark's distance is calculated between the four samples, and the following distances $\{d_{ij}\}$ are obtained:

$$\mathbf{D}^* = \begin{bmatrix} 0 & 0.415 & 0.559 & 0.833 \\ 0.415 & 0 & 0.763 & 0.960 \\ 0.559 & 0.763 & 0 & 0.500 \\ 0.833 & 0.960 & 0.500 & 0 \end{bmatrix}. \tag{4.5}$$

To embed the matrix $\mathbf{X}$ into a Euclidean space, the matrix $\mathbf{B}$ in equation 4.3 is first calculated. The matrix $\mathbf{B} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}'$, then provides $\mathbf{Y} = \mathbf{V}\mathbf{\Lambda}^{1/2}$. $\mathbf{Y}$ is referred to principal axes through the centroid as in PCA and is given as:

$$\mathbf{Y} = \begin{bmatrix} -0.271 & 0.129 & 0.146 & 0 \\ -0.448 & -0.167 & -0.084 & 0 \\ 0.217 & 0.206 & -0.116 & 0 \\ 0.502 & -0.168 & 0.054 & 0 \end{bmatrix}. \tag{4.6}$$

The Euclidean embedded representation of the matrix $\mathbf{X}$ is shown in Figure 4.1. The best-fitting r-dimensional subspace $\mathscr{L}$ is obtained from the first $r = 2$ principal components given by the first two columns of $\mathbf{Y}$.

Finding the representation $\mathbf{Y}$ is also known as *principal coordinate analysis* (PCO). If $\mathbf{B}$ is not positive semi-definite, then it will not be possible to find $\mathbf{Y}$. The above example shows how four samples are embedded in a three-dimensional space. When there are more than four samples, the embedding cannot be visualised even though it can be done algebraically in any $(n-1)$ high dimensional space. Thus, one has to resort to approximations given by PCO, which amounts to the same thing as doing PCA on the points embedded by the simple case as in the example.

4.4 Representation of variables

A consequence of working with distances is there is no provision for representing the original variables in $\mathscr{R}$, only the sample points. Gower and Harding (1988) show this can be done by considering particular sequences of new points that can be added to the plot. For the k-th variable, a sequence of new points is chosen to

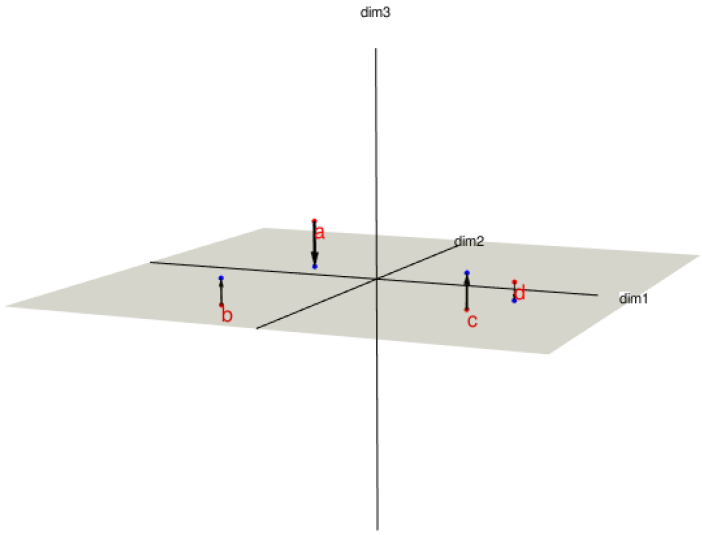

FIGURE 4.1: Euclidean embedded representation of the matrix $\mathbf{X}$ in $\mathscr{R}$ shown by the red points; the blue points are the projected points onto the r-dimensional subspace $\mathscr{L}$ given by the first two columns on $\mathbf{Y}$.

represent the sequence of points along the k-th axis of a p-dimensional scatterplot.

For variable k, particular sequences of points correspond to $\mu_k \mathbf{e}_k$, where μ_k varies over the range of the k-th variable and $\mathbf{e}_k$ is a p x 1 vector with all the elements equal to zero except the k-th element equals to 1. The sequence of $\mu_k \mathbf{e}_k$ points can be regarded as a *pseudo-axis* as μ_k varies, and if embedded in the Euclidean space, a k-th axis is defined. There is no guarantee the axis will be linear, and will therefore, be referred to as a *trajectory*.

To embed a new point, $\mathbf{x}^*$ from the sequence of points $\mu_k \mathbf{e}_k$ representing the k-th axis, the vector $d_{n+1} = \{-\frac{1}{2} d^2_{n+1,i}\}$ between the samples $\mathbf{x}_i$ and the new point $\mathbf{x}^*$ must be calculated according to the chosen Euclidean embeddable distance measure. Gower (1968) show that for embedding this sample into the m-dimensional Euclidean space $\mathscr{R}$, a further $(m+1)$-th dimension is needed where m is defined as $n-1$. The coordinates of the new point $\mathbf{x}^*$ is thus given by:

$$\mathbf{y}^{*\prime} = (\mathbf{y}^{\prime} : 1 \text{ x } m, y_{m+1}) \tag{4.7}$$

with

$$\mathbf{y} = \mathbf{\Lambda}^{-1} \mathbf{Y}^{\prime} (\mathbf{d}_{n+1} - \frac{1}{n} \mathbf{D1}) \tag{4.8}$$

and

$$y^2_{m+1} = \frac{1}{n^2} \mathbf{1}^{\prime} \mathbf{D1} - \frac{2}{n} \mathbf{1}^{\prime} \mathbf{d}_{n+1} - \mathbf{y}^{\prime} \mathbf{y}. \tag{4.9}$$

Every embedded point creates a new extra dimension in order to represent all the dissimilarities exactly. However, Gower and Hand (1996) give arguments why the

process of adding new points may be as if there were only one extra dimension.

The original variables, axes of X_1 and X_2 in the example from Section 4.3 are embedded into $\mathscr{R}$ by calculating the distance vector $\mathbf{d}_{n+1}$ from the pseudo-sample $\mathbf{x}^* = \mu_k \mathbf{e}_k$ for a series of μ_k values and for $k = 1, 2$. The embedded axes using Clark's distance are shown in Figure 4.2 and can be seen to be highly nonlinear.

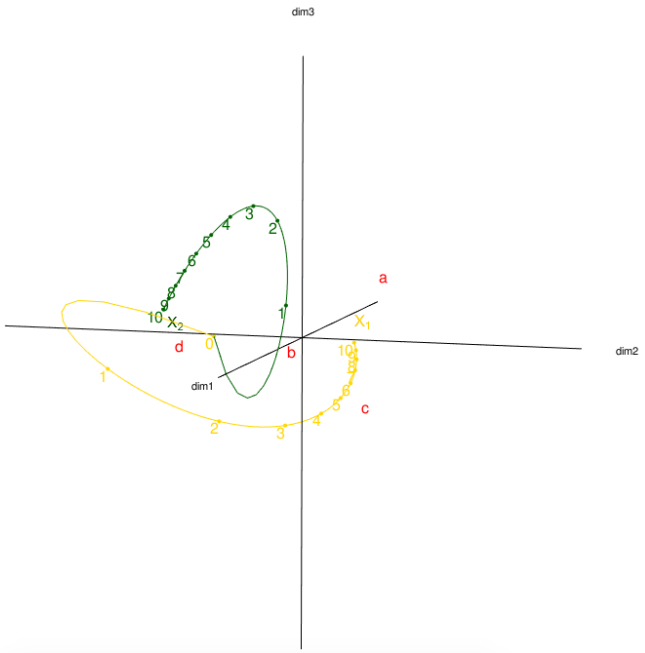

FIGURE 4.2: Embedded axis for variables X_1 and X_2 with 4 sample points.

4.5 Prediction biplot axes

To construct prediction biplot axes in $\mathscr{L}$, the process follows similar principles to those described in Chapter 3 for PCA biplots. To predict the marker μ_k on the k-th biplot axis, a plane $\mathscr{N}$ is constructed at the point corresponding to $\mu_k \mathbf{e}_k$. The plane $\mathscr{N}$ is normal to the embedded Cartesian axes in the higher dimension $\mathscr{R}^+$. All points in the plane predict the value μ_k for the k-th variable. The plane intersects the r-dimensional approximation space $\mathscr{L}$, in an $(r-1)$-dimensional linear subspace $\mathscr{L} \cap \mathscr{N}$.

Since embedding the marker $\mu_k \mathbf{e}_k$ induces an extra dimension, it is not possible to visually represent the marker and the plane $\mathscr{N}$ normal to the marker. However the intersection spaces can be visually represented as they exist in the $(r-1)$ dimension, as there are $(r-1)$-dimensional subspaces in the r-dimensional plane. The intersection spaces for all marker points for the k-th trajectory tend not to be parallel as in the PCA case, which leads to nonlinear biplot trajectories. This is illustrated in Figures 4.3 and 4.4 for original variables X_1 and X_2.

Gower and Ngouenet (2005) give details of three different methods to obtain biplot prediction trajectories, namely normal projection, circular projection and back

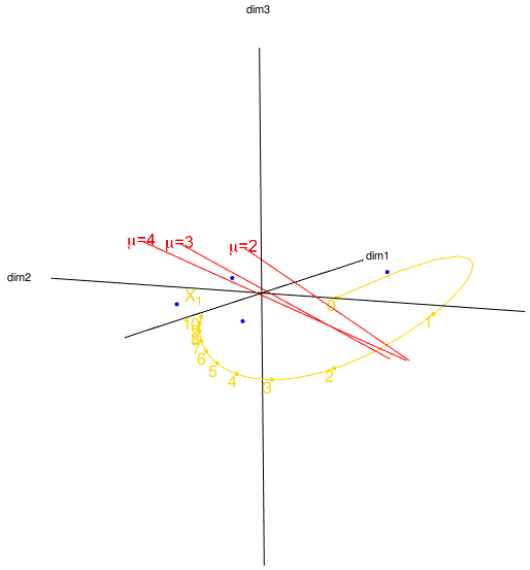

FIGURE 4.3: Intersection spaces $\mathscr{L} \cap \mathscr{N}$ as seen by red lines for the original variable X_1 at points $\mu = 2, 3$ and 4.

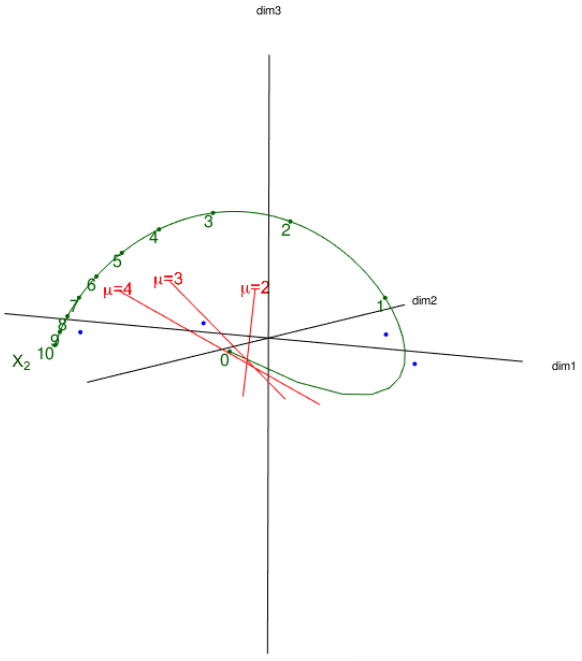

FIGURE 4.4: Intersection spaces $\mathscr{L} \cap \mathscr{N}$ as seen by red lines, for the original variable X_2 at points $\mu = 2, 3$ and 4.

projection. The different trajectories are constructed by identifying different 'optimal' points on the intersection spaces $\mathscr{L} \cap \mathscr{N}$. Each prediction method identify the same intersections and hence will the yield the same predictions.

With normal projection, a trajectory is found that is normal to all intersection spaces $\mathscr{L} \cap \mathscr{N}$. To compute normal projection trajectories involves numerical

integration, which is explored in Gower, Lubbe, and Le Roux (2011). Circular projection defines the marker μ on the intersection space as the orthogonal projection of the origin O. Circular projection will be further discussed in Chapter 7. The third method for obtaining prediction biplot trajectories, back projection is based on finding the point on the intersection space, nearest to the marker μ on the embedded original variable axis. The nearest point is found by back projecting the point from the intersection space onto the axis.

The biplot prediction trajectories for any method can be viewed in the r-dimensional biplot for all p variables. Each method predicts a new point P differently in the biplot. In normal projection, the projection of a point P on the biplot, is a point P^* on the trajectory where the line PP^* intersects the trajectory orthogonally.

In circular projection, a circle is constructed with OP as the diameter. Let $\beta_k(\mu)$ correspond to the k-th two-dimensional biplot trajectory at a specified marker μ. Then the diameter OP subtends a right angle at the point $\beta_k(\mu)$ where the circle intersects the biplot axis ensuring that P is orthogonally projected onto the line $O\beta_k(\mu)$. The predicted value of the original variables for point P corresponds to the point $\beta_k(\mu)$ on the biplot trajectory. If the biplot trajectory intersects the circle more than once, the intersection point closest to O is selected.

In the back projection case, Gower and Ngouenet (2005) derived formulae for constructing prediction biplot axes. Two methods are proposed for reading off the prediction for a sample from a curved biplot axis representing a variable: through normal projection onto the line or through circle projection. However the methodology relies on embedding the original p-dimensional observations and Cartesian axes in Euclidean space. This results in limiting the methodology to dissimilarities that are Euclidean embeddable. The methodology is also restricted to metrics that are additive in the variables for the squared distances relies on classical MDS. This becomes very complicated to implement, and computationally expensive.

It is important to note that once the prediction biplot trajectory is constructed in $\mathscr{L}$, there is no need to consider $\mathscr{R}$ or $\mathscr{R}^+$ nor for the embedded representation of the original axis in these spaces. What is available to the user is the biplot space $\mathscr{L}$ which contains everything necessary for prediction.

4.6 Example

Consider the data in Table 3.1. A nonlinear biplot can be constructed of this data using the Clark's distance as the Euclidean embeddable distance. Figure 4.5 displays the nonlinear biplot using circular projection as the construction of axes. This is different to the PCA biplot in Figure 3.8 of the same data set.

One could compare predictions of the sample points in Table 3.1 for all variables using the two different biplots. However, the main attraction of the nonlinear biplot is that if data were to be approximated using different distance metrics other than Euclidean, provided that they are Euclidean embeddable, a biplot can still be constructed for such data.

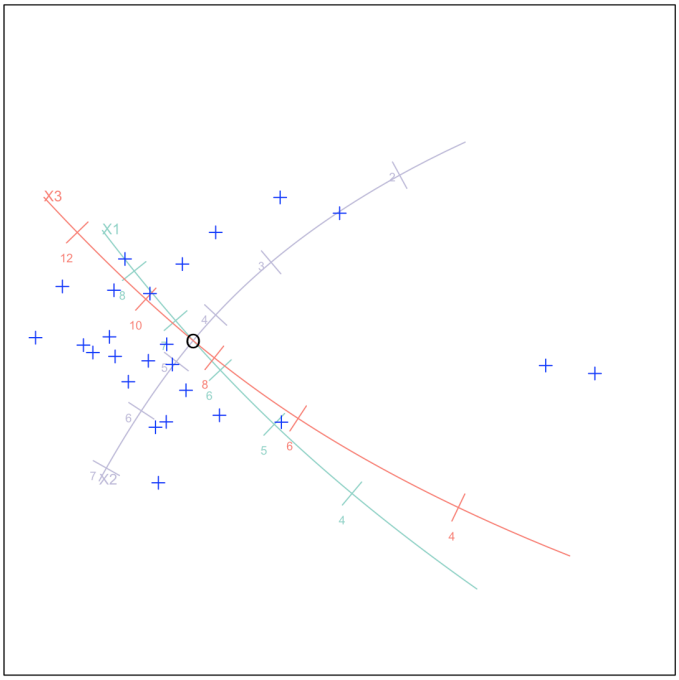

FIGURE 4.5: nonlinear biplot of Table 3.1 constructed with circular projection axes and distance measure set to Clark's distance. The origin is denoted by O and sample points by blue crosses.

4.7 Summary

The nonlinear biplot is an extension of the PCA biplot that makes provision for distances other than the Euclidean distance. When the data is approximated using distances that are Euclidean embeddable like the Clark's or square root of the Manhattan distance, a nonlinear biplot is constructed. It should be noted that if the Euclidean distance is used as the approximation of distances between sample points in the construction of a nonlinear biplot, then the biplot will result in a PCA biplot. Therefore, predictions using nonlinear biplots cannot be less accurate than predictions obtained from PCA biplots, since a PCA biplot is a special case of a nonlinear biplot.

The nonlinear biplot in this chapter is constructed with the help of R (R Core Team, 2018) functions provided in Gower, Lubbe, and Le Roux (2011).

Chapter 5

Spline-based nonlinear biplots

This chapter discusses spline-based nonlinear biplots proposed by Groenen, Le Roux, and Gardner-Lubbe (2015). It is another type of biplot just like the PCA and nonlinear biplot in previous chapters. The chapter discusses its construction using B-splines and ends of with a brief application to principal surfaces.

5.1 Introduction

Groenen, Le Roux, and Gardner-Lubbe (2015) proposed a simple nonlinear biplot that represents the marker points of a variable on a curved line that is governed by splines. Spline-based nonlinear biplots construct nonlinear biplot axes for any representation of sample points. The sample points can be configured using a multitude of techniques that produce a representation of sample points in the two-dimensional space. This could be through PCA, MDS techniques using Euclidean and non-Euclidean embeddable dissimilarity measures, etc.

It can be seen as a geometric form of nonlinear regression where a curve represents the variable to be predicted and the marker point on the curve nearest to the sample gives the predicted value for that sample. As proposed by Groenen, Le Roux, and Gardner-Lubbe (2015), spline-based nonlinear biplots are far simpler, both in its computation and interpretation than nonlinear biplots.

5.2 Constructing nonlinear axes using B-splines

Consider any configuration of sample points in the two-dimensional space. In PCA in Chapter 3, the configuration is the first two columns of $\mathbf{XV}$, for a data matrix $\mathbf{X}$: n x p and $\mathbf{V}$ is the right singular vectors of $\mathbf{X}$. In nonlinear biplots or MDS techniques in Chapter 4, the samples are represented using an approximation of distances given by the first two columns of $\mathbf{Y}$ (equation 4.1).

For a data matrix, $\mathbf{X}$:

$$\begin{bmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix} \tag{5.1}$$

and its configuration of sample points, say $\mathbf{Y}$:

$$\begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \\ \vdots & \vdots \\ y_{n1} & y_{n2} \end{bmatrix}, \tag{5.2}$$

the goal is to predict the value of the variable by the marker value of the nearest marker point on the trajectory.

The method uses the full data $\mathbf{X}$ and its configuration of sample points $\mathbf{Y}$ in two dimensions to construct a spline-based trajectory. A similar approach can be thought of as in principal curves in Chapter 2. However, a principal curve can be found such that it is as close as possible to all observations in only one set of data, either in the full data $\mathbf{X}$ or its configuration $\mathbf{Y}$, but not both. The spline-based method can rather be seen as a geometric form on nonlinear regression, where a curve represents a variable to be predicted and a marker point on the curve nearest to the sample gives the predicted value for that sample.

A calibrated trajectory is to move through the points which represent the samples. For each variable, a set of markers μ are determined from the range of the variable to give n_μ markers. The curved line of n_μ connected markers is represented by an n_μ x r matrix $\mathbf{Z}$ and constitutes the nonlinear biplot axis for a given variable. A calibrated trajectory is constructed for each variable. The smoothness of the trajectory $\mathbf{Z}$ for a given variable, is maintained through splines which can be thought of as piecewise polynomial functions.

The input will be a vector μ of n_μ equally spaced marker values. The output will be coordinates of these marker points (a column of $\mathbf{Z}$) that are piecewise polynomials. To obtain a smooth polynomial, its degree u should be at least 2. The pieces are formed by consecutive intervals that are joined by the interior knots. The higher the number of interior knots v, the more flexible the spline. B-splines are conveniently used, as they can be expressed in terms of basis vectors. (Groenen, Le Roux, and Gardner-Lubbe, 2015).

The B-spline provides an n_μ x $(u + v + 2)$ basis matrix $\mathbf{M}$ and any linear combination yields a spline. As the coordinates of the trajectories in two dimensions are needed, $\mathbf{Z}$ is the linear combination $\mathbf{Z} = \mathbf{MB}$ with $\mathbf{B}$ the $(u + v + 2)$ x r matrix of unknown weights. There are more computational details on B-splines given in Chapter 5, Appendix in Hastie, Tibshirani, and Friedman (2009).

5.3 Algorithm

To construct a nonlinear biplot axis for variable j, the following is proposed (Groenen, Le Roux, and Gardner-Lubbe, 2015):

1. The vector of marker values μ is taken to have n_μ equally spaced elements ranging from the minimum of $\mathbf{X}_j$ to the maximum of $\mathbf{X}_j$. The range is further extended by a factor 0.6. The number of markers n_μ is chosen to be large ($n_\mu = 1000$) to have a sufficiently fine grid of marker points.

2. The B-spline basis matrix $\mathbf{M}$ is calculated from μ for a spline degree of u with v interior knots. The interior knots are spaced at equally spaced quantiles of μ.

3. For any $\mathbf{B}$, let $\mathbf{Z} = \mathbf{MB}$ be the trajectory of n_μ marker points. The aim of the algorithm is to find $\mathbf{B}$ such that for each fixed y_{ij}, the closest point μ_k on the trajectory, corresponds as closely as possible to x_{ij}. The objective can be given by the least-squares loss function

$$L(\mathbf{B}|u, v) = \sum_{i=1}^{n} (x_{ij} - \hat{x}_{ij})^2, \tag{5.3}$$

that is minimised over the spline coefficient matrix $\mathbf{B}$. $\hat{x}_{ij}$ is the marker in μ closest to y_{ij} and $\mathbf{Z} = \mathbf{MB}$ determines the coordinates of the n_μ marker points in μ, that is,

$$\hat{x}_{ij} = \mu_{k^*} \quad \text{where } k^* = \text{argmin}_k ||z_{kj} - y_{ij}||^2, \tag{5.4}$$

where $||z_{kj} - y_{ij}||^2$ is the squared Euclidean distance between marker points μ_k and the sample point y_{ij} in the configuration.

4. Lastly, the calibrated spline for variable j is marked with markers in the original units of measurement.

The interpretation is straightforward; to obtain the value predicted for a sample point on a variable of interest, one has to find the marker point of the trajectory that lies closest to the point representing the sample point. Groenen, Le Roux, and Gardner-Lubbe (2015) give further details on minimising the loss function $L(\mathbf{B}|u, v)$ using the Nelder-Mead algorithm and give approaches to selecting u and v.

5.4 Example

Consider the same example from Chapter 3 and 4 using data from Table 3.1. To construct spline-based nonlinear biplot axes for the variables X_1, X_2 and X_3, combined code from R (R Core Team, 2018) and Fortran compiled by Groenen, Le Roux, and Gardner-Lubbe (2015) is used.

The configuration of points is that of PCA, MDS using Clark's distance and MDS using the Stress Majorisation of a complicated function (SMACOF) method. The SMACOF method is another method in MDS to solve the problem of locating points in a low dimensional Euclidean space in such a way that the distances between the points approximate the given dissimilarities. The SMACOF method uses a matrix of weights to minimise a quadratic stress function. More details on this method can be seen in Borg and Groenen (2005).

Figures 5.1, 5.2 and 5.3 presents the biplots using spline-based axes with PCA, MDS Clark's and MDS SMACOF as the configuration of sample points, respectively. The blue crosses represent the 25 sample points in Table 3.1. Tables 5.1, 5.2 and 5.3 provide predictions of samples under each variable X_1, X_2 and X_3. This

is done to compare the predictiveness of each biplot; the PCA biplot from Chapter 3, the nonlinear biplot using Clark's distance from Chapter 4 and the spline biplots from this chapter. The sums of squared differences between the actual observations and the predictions, in this chapter are referred to as squared errors, are taken for each biplot method and used a measure of comparison.

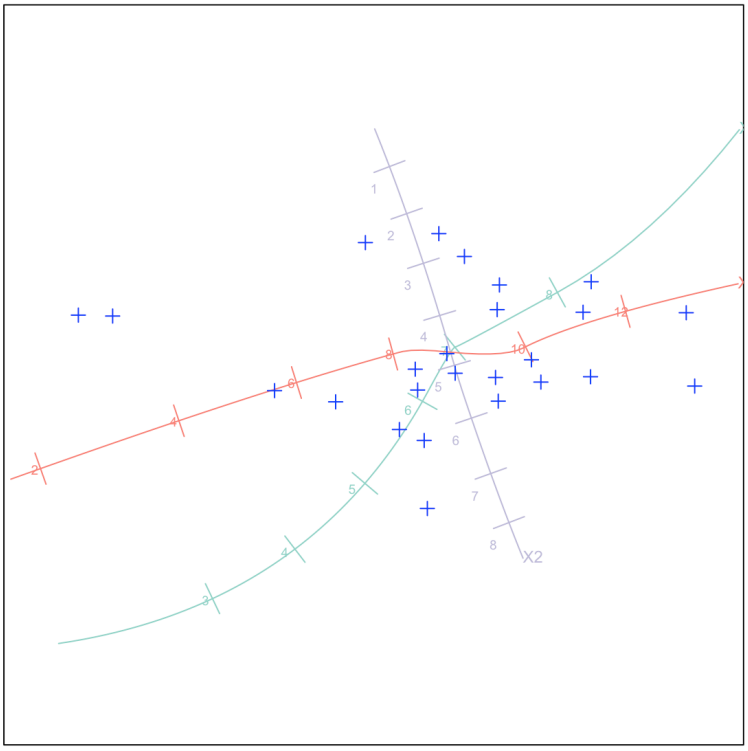

FIGURE 5.1: Spline biplot using the configuration of PCA.

5.4.1 Discussion

The spline-based nonlinear biplots for the data in Table 3.1 all do significantly better in predicting the sample points. One can compare the predictions of the PCA with the spline PCA biplot and the nonlinear biplot with Clark's distance with the spline Clark's distance biplot, for all variables. Although the spline-based nonlinear biplots' predictions do not predict better for some individual cases, the overall squared errors show an overall improvement in predictions.

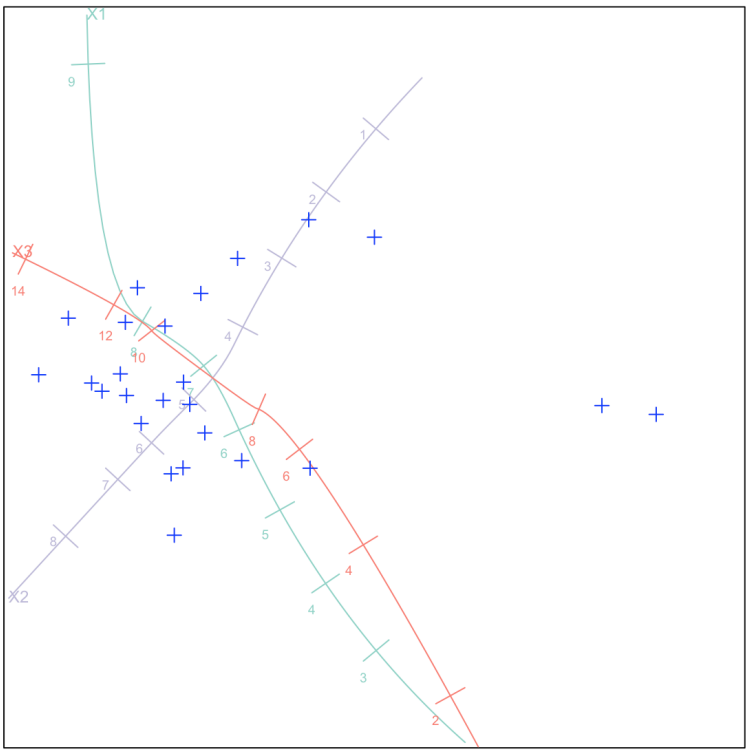

FIGURE 5.2: Spline biplot using the configuration of MDS with dissimilarity measure Clark's distance.

5.5 Using λ as a configuration

Since spline-based nonlinear biplots can use any configuration of points, an idea would be to use the $\lambda = (\lambda_1, \lambda_2)$ coordinates from a principal surface presented in Chapter 2. Consider fitting a principal surface onto the same data set in Table 3.1, shown in Figure 5.4. Included in the figure are the λ coordinates of the fitted surface. Using the λ coordinates as the configuration of points in the spline-based method, what will be called a *spline principal surface biplot*, is constructed and is shown in Figure 5.5. Each sample point is predicted using the three spline axes and are given in Table 5.4 with the squared error, for all variables.

The squared errors for all biplots considered in this chapter are summarised in Table 5.5. There is not one biplot that outperforms in terms of the lowest squared error for all variables in the data. For example, the Spline SMACOF biplot has the lowest squared error for variables X and Z, but not for Y. If one were to add the total squared error for each biplot method, all the spline biplots do relatively better than the non-spline biplots like PCA and nonlinear (Clark's distance) biplots. Their total squared errors average around 8.22.

It should be noted that the usability by practitioners is another important dimension on which any axes should be evaluated. Table 5.5 only shows approximation accuracy assuming that it is read off correctly. Therefore, it is not only the squared error to take into consideration, but also the ease of use that is important. The first focus is on accuracy, since ease of use of inaccurate axes would not help.

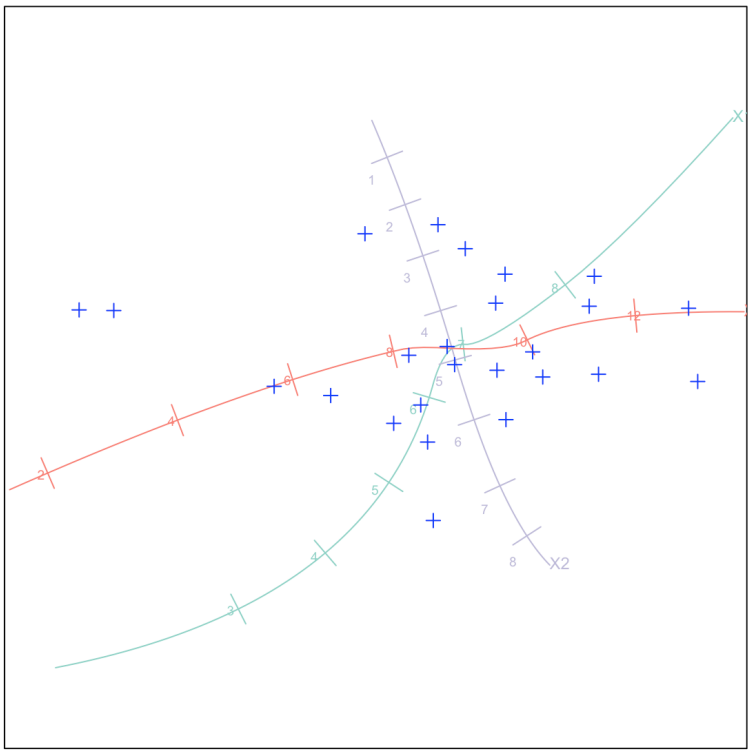

FIGURE 5.3: Spline biplot using the configuration of MDS with dissimilarity measure SMACOF.

TABLE 5.1: Predictions for variable X_1 in Table 3.1 in different biplots, with sums of squared differences between actual observations and predictions (squared error) given in the last row.

	X_1	PCA	Nonlinear (Clark's)	Spline PCA	Spline Clark's	Spline SMACOF
1	5.419	6.221	6.119	6.090	6.090	5.816
2	3.130	3.634	3.384	2.896	2.987	2.896
3	6.128	6.605	6.552	6.911	6.820	6.729
4	6.781	6.691	6.817	6.638	6.911	6.820
5	7.347	7.836	7.641	7.915	8.006	7.915
6	7.208	7.554	7.597	7.733	7.459	7.824
7	7.039	6.652	6.815	6.455	6.911	7.276
8	5.466	4.945	4.961	5.177	5.086	5.360
9	7.723	8.141	7.976	8.006	8.006	8.006
10	7.110	6.320	6.532	6.273	6.638	7.094
11	8.136	7.392	7.577	7.641	7.641	7.733
12	6.287	5.831	5.977	5.634	5.816	5.634
13	4.649	5.625	5.457	5.177	5.177	4.995
14	5.799	5.988	6.023	5.725	5.816	5.543
15	8.085	7.407	7.635	7.641	7.459	7.824
16	6.158	6.874	6.778	6.455	6.820	5.999
17	3.557	3.934	3.629	3.717	3.535	3.626
18	6.964	7.311	7.385	7.459	7.185	6.729
19	7.706	7.408	7.478	7.733	7.824	7.824
20	8.348	8.709	8.189	8.280	8.280	8.189
21	8.191	8.389	8.232	8.189	8.098	8.098
22	7.999	7.364	7.462	7.733	8.000	7.824
23	6.211	5.424	5.552	5.543	5.634	5.634
24	8.776	9.056	8.516	8.463	8.371	8.189
25	8.079	6.986	7.197	7.368	7.641	7.641
$\sum(x_i - \hat{x}_i)^2$		**8.253**	**5.204**	**5.583**	**4.270**	**3.302**

TABLE 5.2: Predictions for variable X_2 in Table 3.1 in different biplots, with sums of squared differences between actual observations and predictions (squared error) given in the last row.

	X_2	PCA	Nonlinear (Clark's)	Spline PCA	Spline Clark's	Spline SMACOF
1	5.054	5.216	5.282	5.235	5.147	5.585
2	1.783	1.885	1.887	1.734	1.821	1.821
3	2.173	2.269	2.330	2.259	2.172	2.259
4	4.753	4.735	4.875	4.710	4.972	4.622
5	5.893	5.992	5.714	6.000	5.935	5.935
6	3.744	3.814	3.766	3.834	3.922	3.747
7	5.214	5.135	5.281	5.147	5.235	5.147
8	4.493	4.388	4.666	4.359	4.710	4.447
9	4.708	4.792	4.734	4.710	4.885	4.622
10	4.988	4.828	5.080	4.885	5.060	4.622
11	4.392	4.242	4.414	4.272	4.622	4.184
12	5.909	5.817	5.905	5.760	5.847	5.760
13	7.198	7.395	6.753	7.248	7.160	7.160
14	6.120	6.158	6.078	6.110	6.197	6.110
15	3.235	3.098	3.189	3.134	3.309	3.134
16	5.743	5.888	5.751	5.847	5.760	6.110
17	2.026	2.102	2.024	1.996	2.172	2.084
18	2.470	2.539	2.548	2.522	2.434	2.522
19	5.401	5.341	5.361	5.410	5.410	5.322
20	6.697	6.770	6.056	6.722	6.722	6.635
21	4.255	4.294	4.257	4.272	4.359	4.184
22	5.925	5.796	5.718	5.847	5.935	5.760
23	5.105	4.947	5.267	5.060	5.060	5.147
24	5.350	5.406	5.191	5.410	5.322	5.410
25	5.668	5.447	5.555	5.497	5.585	5.497
$\sum(x_i - \hat{x}_i)^2$		**0.336**	**0.906**	**0.202**	**0.289**	**0.751**

TABLE 5.3: Predictions for variable X_3 in Table 3.1 in different biplots, with sums of squared differences between actual observations and predictions (squared error) given in the last row.

	X_3	PCA	Nonlinear (Clark's)	Spline PCA	Spline Clark's	Spline SMACOF
1	8.711	8.202	8.030	8.531	8.531	8.531
2	3.386	3.066	3.720	3.146	3.453	3.453
3	8.174	7.871	7.870	8.000	7.608	8.069
4	8.732	8.789	8.995	8.838	8.992	8.838
5	11.303	10.993	10.352	11.146	11.300	11.300
6	10.076	9.856	9.902	10.069	9.762	10.223
7	8.609	8.855	9.069	8.838	8.992	8.838
8	5.597	5.927	6.187	5.607	5.607	5.607
9	11.357	11.092	10.645	11.300	11.300	11.454
10	7.733	8.234	8.610	8.377	8.838	8.223
11	9.265	9.737	10.012	9.454	9.762	9.454
12	7.488	7.778	7.931	7.761	8.377	7.454
13	8.574	7.954	7.287	8.685	8.223	8.685
14	8.255	8.134	8.034	8.531	8.531	8.531
15	8.967	9.397	9.807	8.992	9.608	8.992
16	9.899	9.444	9.105	9.454	8.992	9.454
17	3.848	3.608	4.014	3.607	3.915	3.761
18	9.288	9.068	9.200	8.685	9.300	8.685
19	9.922	10.111	10.052	10.000	9.762	10.000
20	12.844	12.616	11.190	12.839	12.839	12.839
21	11.450	11.324	10.916	11.608	11.454	11.608
22	9.783	10.186	10.097	9.916	9.762	9.916
23	6.360	6.859	7.170	6.684	6.838	6.377
24	12.906	12.729	11.488	12.839	12.839	12.839
25	8.787	9.481	9.681	9.300	9.300	9.300
$\sum(x_i - \hat{x}_i)^2$		**3.324**	**13.940**	**1.869**	**4.893**	**1.396**

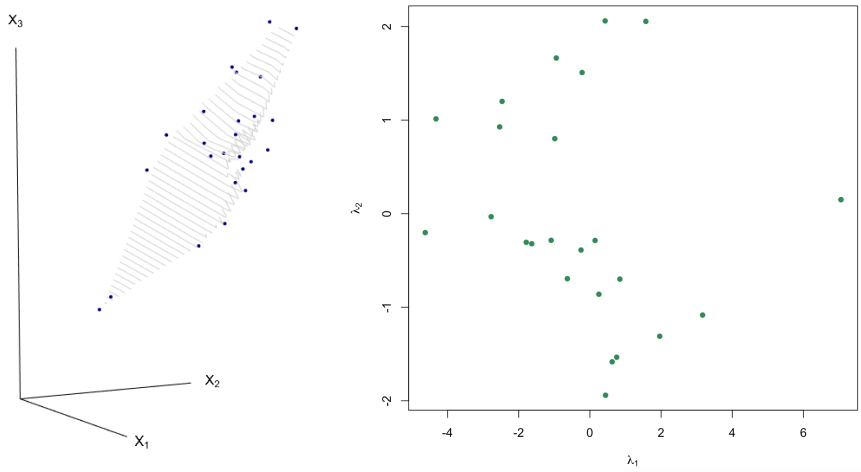

FIGURE 5.4: Principal surface fitted to data in Table 3.1 and configuration of sample point given by $bf\lambda$.

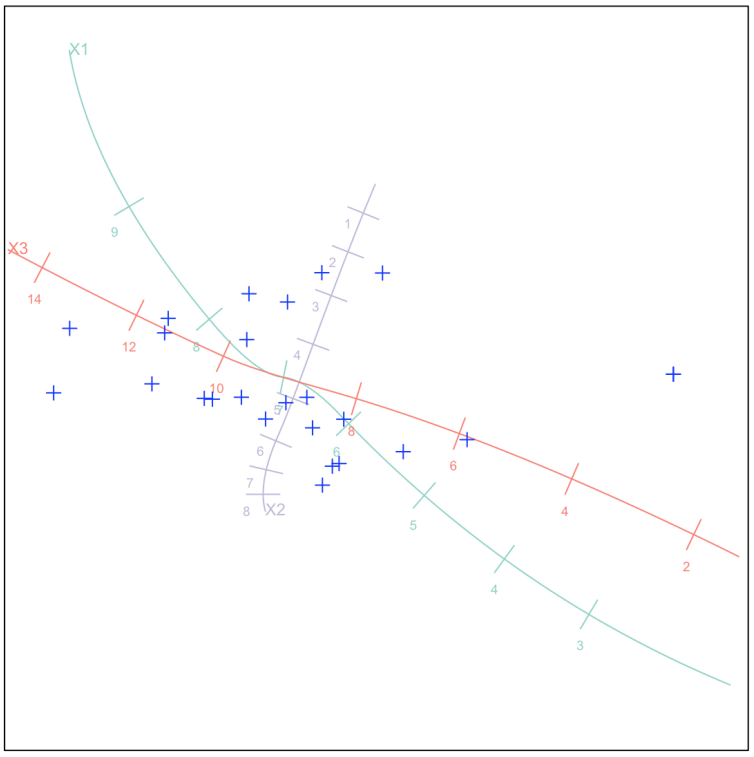

FIGURE 5.5: Spline biplot using the configuration of λ from the principal surface fitted to the data.

TABLE 5.4: Individual predictions for the Spline principal surface (PS) biplot with sums of squared differences between actual observations and predictions (squared error) given in the last row for variables X_1, X_2 and X_3.

Spline PS	X_1	X_2	X_3
1	6.2726	5.322	8.685
2	3.7174	1.909	3.299
3	6.5463	2.084	8.223
4	6.5463	4.885	8.838
5	7.9152	5.935	11.146
6	7.8239	3.484	10.069
7	6.7288	5.147	8.992
8	5.0862	4.622	5.607
9	8.0977	4.710	11.454
10	6.0900	5.000	8.069
11	7.6414	4.359	9.454
12	5.9075	5.935	7.915
13	5.8163	7.160	8.531
14	5.9075	6.110	8.069
15	7.5502	3.309	8.992
16	6.9114	5.585	9.146
17	3.7174	1.909	3.299
18	7.0026	2.522	8.685
19	7.5502	5.585	9.916
20	8.3715	6.722	12.685
21	8.1889	4.447	11.454
22	7.5502	5.760	9.916
23	5.4513	5.147	6.684
24	8.6452	5.322	13.147
25	7.3676	5.322	9.454
$\sum(x_i - \hat{x}_i)^2$	**7.3980**	**0.4742**	**2.4536**

TABLE 5.5: Summary of sum of squared differences between actual observations and predictions (squared error) for each variable for different biplots, and the total squared error.

Biplot	X_1	X_2	X_3	$\sum$
PCA	8.2527	0.3355	3.3236	11.9118
Nonlinear (Clark's)	5.2039	0.9057	13.9402	20.0498
Spline PCA	5.5834	0.2019	1.8686	7.6539
Spline Clark's	4.2703	0.2889	4.8932	9.4524
Spline SMACOF	3.3025	0.7505	1.3960	5.4490
Spline PS	7.3980	0.4742	2.4536	10.3258

5.6 Summary

Spline-based nonlinear biplots' main attraction is that it can use any configuration of sample points to create its biplot axes. It is also much faster and easier to compute than normal projection with nonlinear biplot trajectories described in

Chapter 4. In this chapter, the λ coordinates were used as the configuration of sample points in the spline-based nonlinear biplot methodology. The idea, in this chapter was to present another type of biplot that exists in the literature, with a brief application of using two-dimensional representation of samples from principal surfaces. The representation of sample points λ coordinates will be used in the following chapters that form part of the main chapters of this thesis, introducing the idea of biplots based on principal surfaces.

Chapter 6

Biplots based on Principal Surfaces

The original work of the thesis, biplots based of principal surfaces, starts from this chapter. The chapter begins by introducing the requirements for constructing such a biplot, which includes (1) the representation of samples and (2) biplot axes. However, the focus here will be first on the representation of samples, and then creating sets of intersection spaces to construct biplot axes in later chapters.

6.1 Introduction

The goal is to extend biplot methodology with the use principal surfaces. In PCA biplots from Chapter 3, the plane consisting of the first two principal components is considered the biplot plane. In nonlinear biplots from Chapter 4, the biplot is found to be the plane passing through the centroid of embedded sample points. The planes in each scenario are seen to be a linear plane that passes through the centroid of sample points, such that the distances from the points to the plane are minimised. This is a simple consequence of *Huygens' principle* that the sum of squares about the mean is smaller than the sum of squares about any other point.

A principal surface of a set of data can thus been seen as a smooth nonlinear biplot plane that passes through sample points. The principal surface algorithm also minimises a distance function, however it does not fix the surface to be linear, but rather to curve with the data.

As in biplots, sample points of a data matrix are represented in a two-dimensional space. The different representations of sample points in different biplots was mentioned in Chapter 5. In this case, using the principal surface as the biplot plane, the approach is to use the coordinates, λ of the principal surface $f(\lambda)$ as a representation of sample points. Each p-dimensional point on the principal surface has two coordinates λ_1 and λ_2, as seen in Chapter 2. Representing the samples in two dimensions is a starting point. Thereafter, new theory on constructing axes for each variable can be presented. The emphasis in a biplot based on a principal surface will be on the visualisation of samples and the predictive variable trajectories.

6.2 Full representation of principal surface

What the principal surface function produces is $f(\lambda)$ of size n x p, which are projections of a data matrix $\mathbf{X}$: n x p onto the surface. The principal surface $f(\lambda)$

points tend to be irregularly spaced and there are no set of equations that defines the surface. In order to create a set of surface points that are regularly spaced, it is then necessary to interpolate a further set of points in between the points $f(\lambda)$ to create a fuller representation of the surface. A fuller representation of the surface is needed as it offers better visualisation of the shape of the surface, and it will also be later used to construct biplot axes.

The interpolation method to find a set of surface points f_{grid} that are smooth is based on a method of bivariate interpolation in Akima (1978). For each point on the surface $f(\lambda)$, there are (λ_1, λ_2) coordinates. The interpolation method is performed for each variable j, for $j = 1, \ldots, p$. For each implementation, the interpolation method requires a vector of grid points for (λ_1, λ_2) such that they are evenly spaced out over the range of (λ_1, λ_2).

Let λ_1^0 be a vector that ranges from the minimum of λ_1 to the maximum of λ_1 and λ_2^0 be a vector that ranges from the minimum of λ_2 to the maximum of λ_2, both of size N. The function uses a combination of the vectors, λ_1^0 and λ_2^0, and each variable on the surface to interpolate an N x N grid of points. There will be p such N x N matrices each corresponding to a variable. Each coordinate of f_{grid} consisting of p values, will correspond to the same element in each of the p N x N matrices, thus ultimately producing an N^2 x p matrix. The precise details on how the implemented method of bivariate interpolation works can be seen in Akima (1978) and will not be covered here.

To illustrate how the matrix f_{grid} is produced, consider an n x 3 matrix, $\mathbf{X}$ with variables x, y and z. The fitted principal surface $f(\lambda)$ will consist of n points in 3 dimensions: an n x 3 matrix, and each point will have coordinates $\lambda = (\lambda_1, \lambda_2)$: an n x 2 matrix. λ_1^0 is a sequence of values ranging from the minimum of λ_1 to the maximum of λ_1 and λ_2^0 is a sequence of values ranging from the minimum of λ_2 to the maximum of λ_2, both of length N.

For variable x, the function uses the first column of the surface $f(\lambda)$, λ_1^0 and λ_2^0 to produce an N x N matrix:

$$\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1N} \\ x_{21} & x_{22} & \cdots & x_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{NN} \end{bmatrix}.$$

Similarly for variables y and z, the function produces the following matrices:

$$\begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1N} \\ y_{21} & y_{22} & \cdots & y_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ y_{N1} & y_{N2} & \cdots & y_{NN} \end{bmatrix}$$

and

$$\begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1N} \\ z_{21} & z_{22} & \cdots & z_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ z_{N1} & z_{N2} & \cdots & z_{NN} \end{bmatrix}.$$

The interpolated surface f_{grid} will be an N^2 x 3 matrix with coordinates:

$$\begin{bmatrix} x_{11} & y_{11} & z_{11} \\ x_{12} & y_{12} & z_{12} \\ \vdots & \vdots & \vdots \\ x_{1N} & y_{1N} & z_{1N} \\ x_{21} & y_{21} & z_{21} \\ x_{22} & y_{22} & z_{22} \\ \vdots & \vdots & \vdots \\ x_{2N} & y_{2N} & z_{2N} \\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \\ x_{NN} & y_{NN} & z_{NN} \end{bmatrix}.$$

The f_{grid} provides N^2 points of the surface instead of n points from $f(\lambda)$ and thus gives a fuller representation of the principal surface. The choice of N is dependent on the size of the data. If a data set has 100 points for example, a choice of $N = 50$ will produce a surface with 2500 points, which from practical experience appeared to be sufficient to demonstrate the full representation of the surface.

As there are λ coordinates for $f(\lambda)$, there should be λ_{grid} coordinates for f_{grid}. The N^2 x 2 grid of coordinates is calculated as:

$$\lambda_{grid} = f_{grid}\vartheta_2, \tag{6.1}$$

where ϑ_2 are the first two columns of the right singular vectors of the final fitted principal surface $f(\lambda)$. This comes from the projection step in the principal surface algorithm for data sets (equation 2.6).

The purpose of interpolating such a surface f_{grid} not only offers better visualisation of the surface as implemented to the surfaces in Chapter 2, it also plays an important role in the following sections in ultimately helping to construct principal surface biplot axes.

Consider simulating 100 points from the blanket surface data in Chapter 2 with three variables X, Y and Z. Figure 6.1 displays the principal surface $f(\lambda)$ of $n = 100$ points shown in blue, and f_{grid} points shown in grey. Here $N = 50$, that produces a grid surface of $N^2 = 2500$ points.

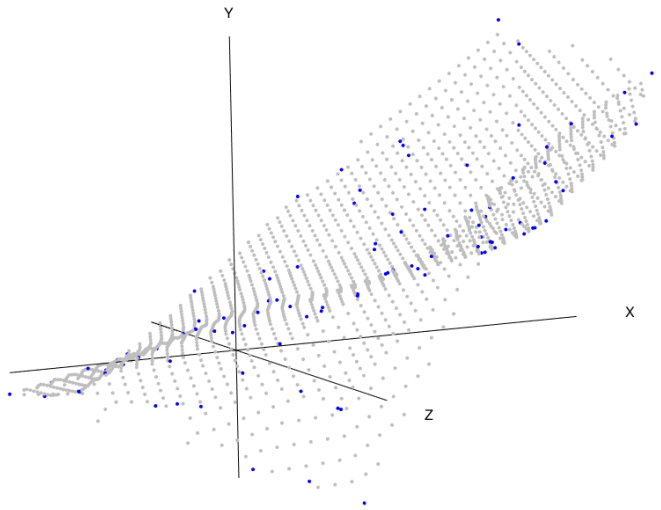

FIGURE 6.1: Principal surface fitted to blanket surface data of $n = 100$ points shown in blue. The grey points represent $N^2 = 2500$ f_{grid} points.

6.3 Interpolation axes

In Chapter 3, Section 3.5, the process of interpolation as opposed to prediction in a biplot is mentioned. The biplots thus far have been equipped for the use of prediction. This section shows how to construct interpolation biplot axes with principal surfaces in the same manner as interpolation axes would be constructed in a typical PCA biplot.

To illustrate this idea, consider the blanket surface data in Figure 6.1, that presents the full representation of a principal surface fitted to the simulated data, when $N = 50$. An interpolation axis for variable X is constructed by first projecting each of the marker points on the axis in p-dimensions to the closest point on the principal surface. The markers of X, are determined by a sequence of values ranging from the minimum to the maximum that X holds and then finding a *convenient* set of markers from the range. The convenient set of markers can be computed from the `pretty` function in R (R Core Team, 2018), as done in PCA and other biplots. The projection of the markers onto the surface are shown in Figure 6.2. For the other biplots considered in previous chapters, the projection of the axis is done on the biplot plane.

Each projected point on the surface now represents the marker points of the axis. The corresponding projected points are plotted in two dimensions in terms of their λ_{grid} coordinates. The points are labelled according to the marker points of the axis and are joined by a line to form the X interpolation biplot axis. This is shown in Figure 6.3. Similarly, this is done for variables Y and Z and their respective axes are given in Figure 6.4.

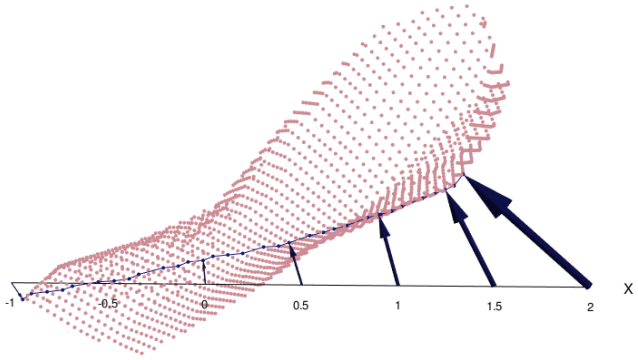

FIGURE 6.2: Projection of X axis onto the surface.

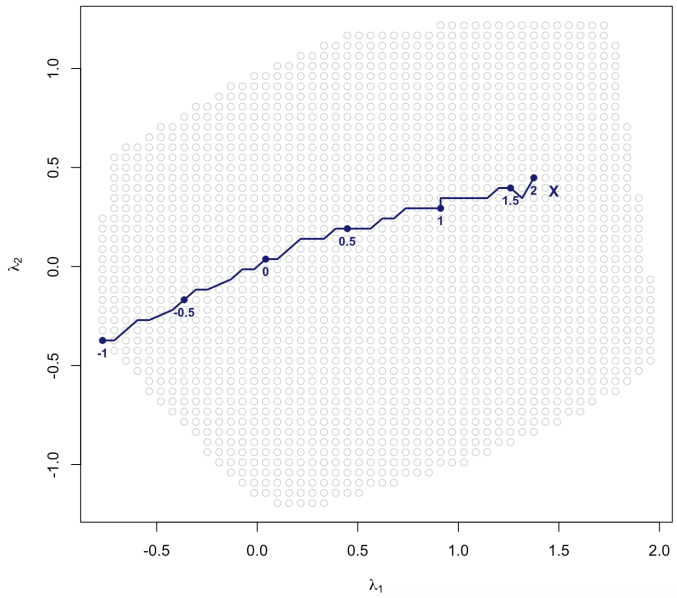

FIGURE 6.3: Interpolation axis for variable X in two-dimensions.

Groenen, Le Roux, and Gardner-Lubbe (2015) warn that interpolation axes cannot be used for prediction and vice versa; and thus seen from this section, the axes need to be constructed differently. Figure 7.8, which appears in the next chapter, highlights the differences in the construction of prediction and interpolation axes. The next section gives attention to creating intersection spaces analogous to the intersection spaces created in PCA and nonlinear biplots to construct prediction biplot axes that will be utilised for prediction.

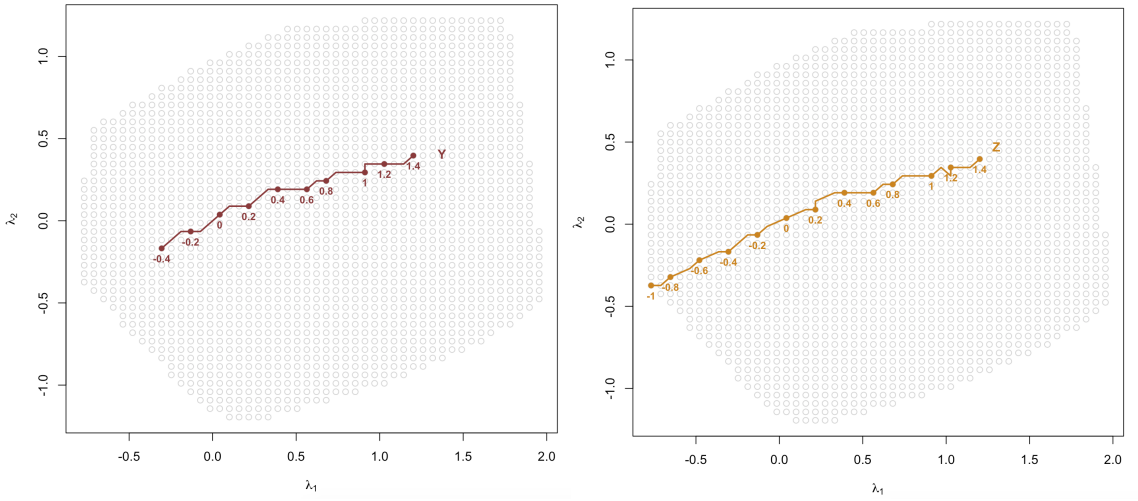

FIGURE 6.4: Interpolation axis for variables Y and Z, respectively.

6.4 Intersection spaces

In Chapters 3 and 4, hyperplanes $\mathcal{N}$ are constructed for different markers of a variable to create intersection spaces on the biplot. These intersection spaces are used to create a biplot axis representing a variable. The same analogy can be used in this case. A $(p-1)$-dimensional hyperplane $\mathcal{N}$ perpendicular to an axis is constructed at a specific marker value μ. The intersection will now be between the principal surface and the planes $\mathcal{N}$ at each marker point. All points on the intersection space for a marker μ of a variable, will predict the value μ for that variable.

Consider the blanket surface data in Figure 6.1 in the previous section. This data set will be used here to illustrate intersection spaces, as mentioned in Chapter 2, because of its simple nature and to effectively visualise the surface. Since $p = 3$ in this case, a two-dimensional plane $\mathcal{N}$ perpendicular to an axis is constructed at a specific marker point. In Figures 6.5, 6.6 and 6.7, five $\mathcal{N}$ planes are constructed at different marker values for each variable X, Y and Z, respectively. For illustration of the construction of marker points, the marker points for each variable here, are chosen to range from the minimum to the maximum value of the respective variable, with the number of markers set to five. In practice, convenient marker values in the original unit of measurement will be chosen and will be illustrated in an application from Section 6.4.1.

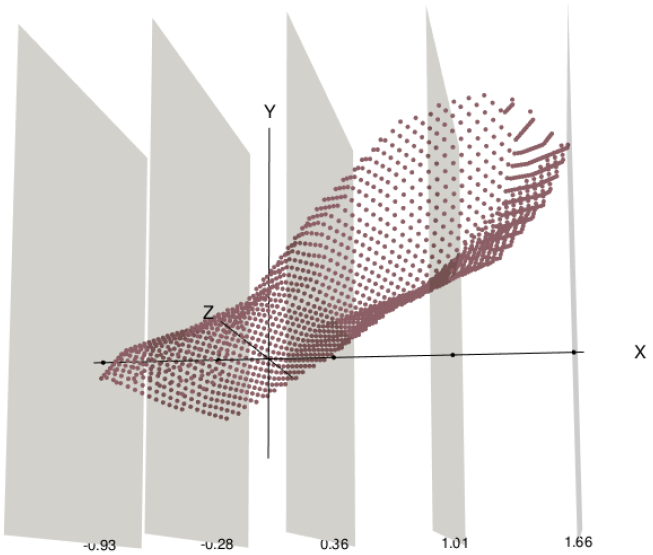

FIGURE 6.5: Five planes cutting through the principal surface for variable X.

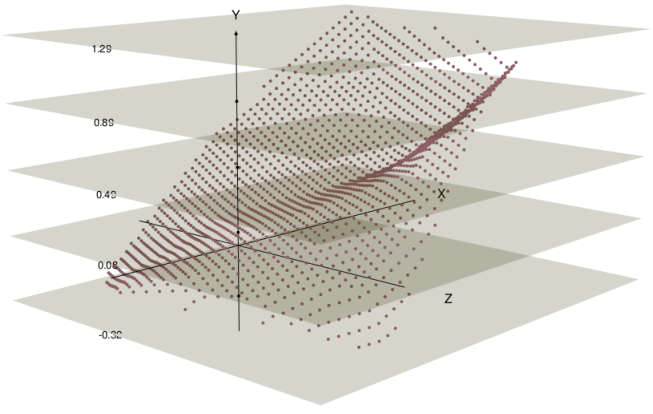

FIGURE 6.6: Five planes cutting through the principal surface for variable Y.

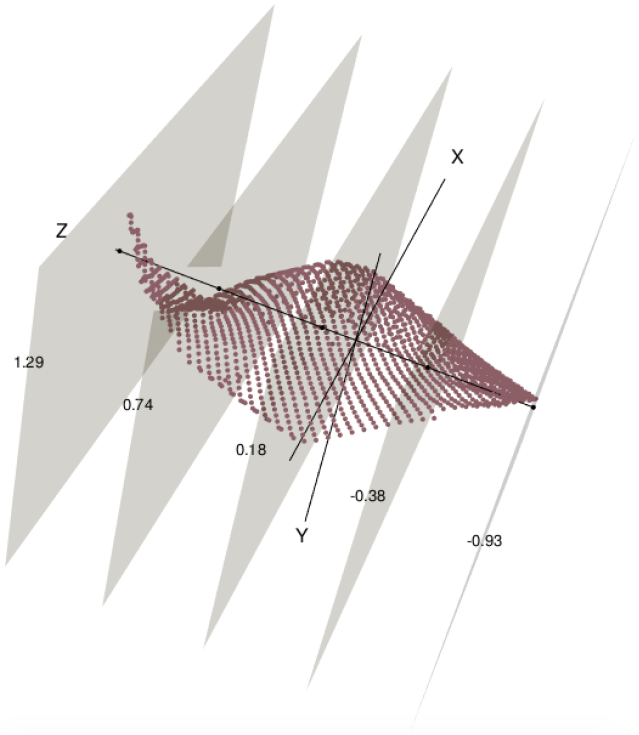

FIGURE 6.7: Five planes cutting through the principal surface for variable Z.

Since the surface cannot be expressed as a function, it is problematic to express the intersection line as a function (even with $p = 3$). As the number of dimensions increase, equating the surface and the $\mathcal{N}$ planes to find intersection lines involves many complicated systems of equations to solve. The full representation of the principal surface f_{grid} can help make this task easier in finding the intersection spaces. Figure 6.8 presents the λ_{grid} points for the principal surface f_{grid} using equation 6.1.

Using the grid of coordinates λ_{grid} that lie in two dimensions in Figure 6.8 and the surface f_{grid} in Figure 6.1, the intersections for a variable j in the two-dimensional space can be found as follows:

Consider a tile from the surface f_{grid} of three points f_a, f_b and f_c shown in Figure 6.9. The values of $[f_a]_j$, $[f_b]_j$ and $[f_c]_j$ are single values that are adjacent to each other on the surface, of the j-th variable. Each point has a corresponding grid coordinate, (λ_1, λ_2). The point $[f_a]_j$ has coordinates $(\lambda_{a1}, \lambda_{a2})$, $[f_b]_j$ has coordinates $(\lambda_{b1}, \lambda_{b2})$ and $[f_c]_j$ has coordinates $(\lambda_{c1}, \lambda_{c2})$. Although these coordinates are denoted by λ, they represent the grid coordinates from Figure 6.8.

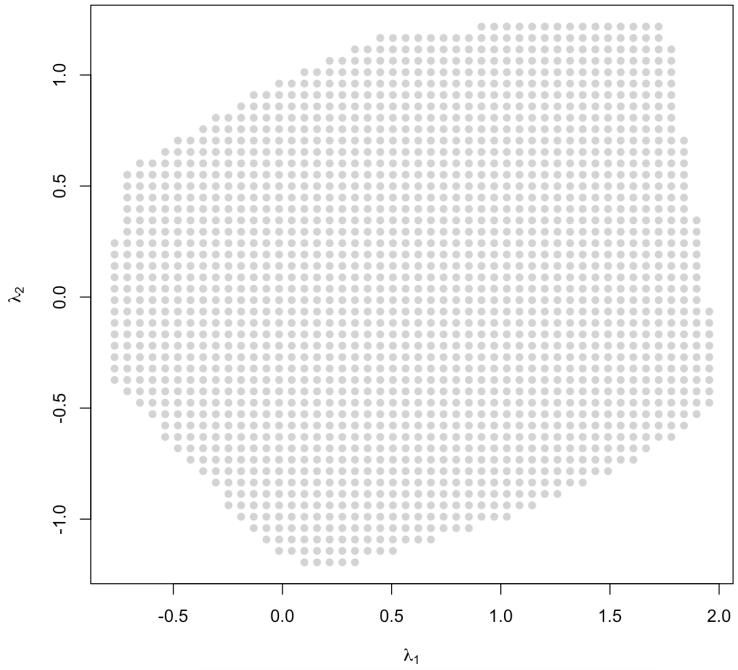

FIGURE 6.8: λ_{grid} points of blanket surface data.

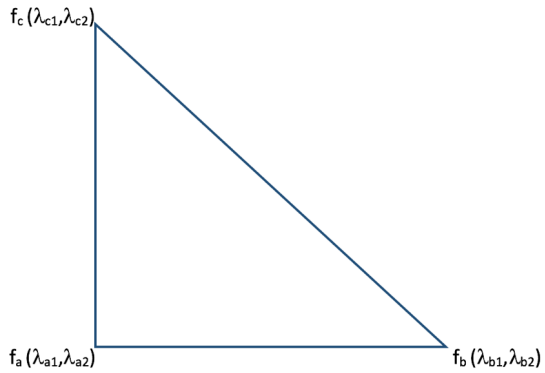

FIGURE 6.9: Three points on the principal surface f_{grid} labelled a, b and c each with λ_{grid} coordinates.

The first step is to check whether a plane $\mathcal{N}_\mu$ intersects with the tile. This is done by checking if one or two of the $[f]_j$'s $\leq \mu$ and two or one $[f]_j$'s $\geq \mu$, respectively. If this is true then the plane intersects with the tile, and the intersection could be any of three combinations with the line cutting between any two sides of the triangle, shown in Figure 6.10.

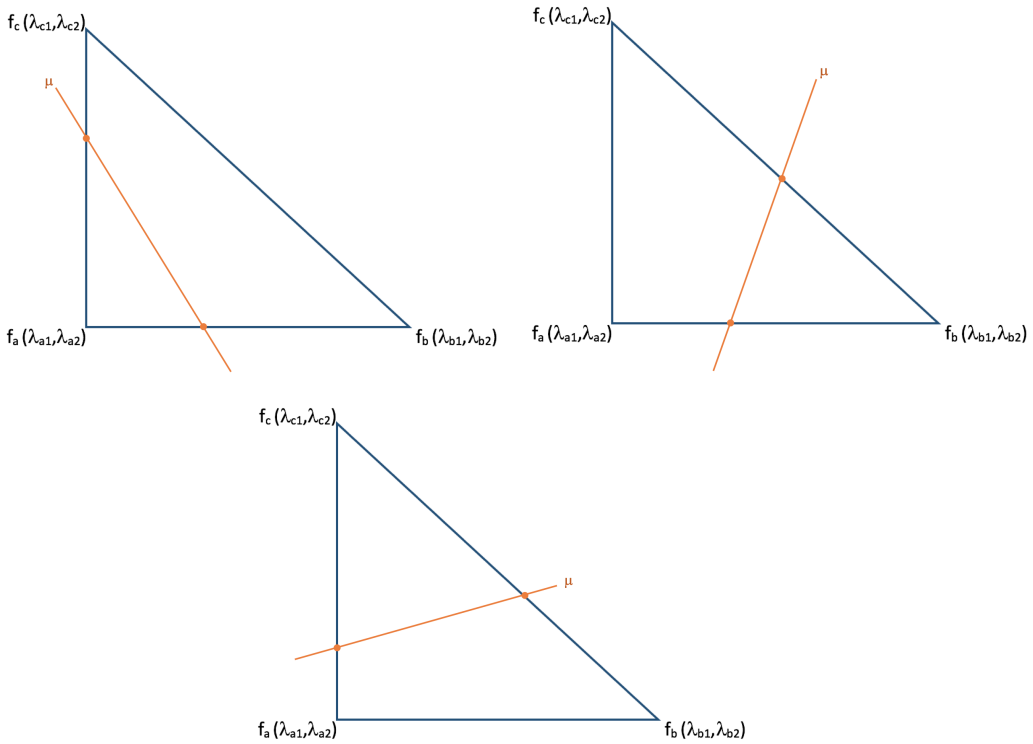

FIGURE 6.10: Three combinations of intersection lines between three points on the principal surface and $\mathcal{N}_\mu$.

Figure 6.10 illustrates that in summary there can be three points of intersection, one on each side of the tile. This is summarised in Figure 6.11 with labelled points *1*, *2* and *3*.

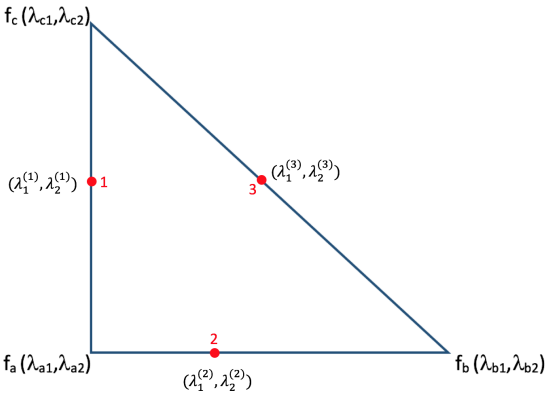

FIGURE 6.11: Summary of three intersection points from the three figures in Figure 6.10.

To calculate the λ coordinates in the two-dimensional space of the three points (*1*, *2* and *3*) in Figure 6.11, there will be three sets of equations, one for each side of

the triangle:

$$\begin{aligned}\lambda_1^{(1)} &= \lambda_{a1} + \frac{\mu - [f_a]_j}{[f_c]_j - [f_a]_j}(\lambda_{c1} - \lambda_{a1}) \\ \lambda_2^{(1)} &= \lambda_{a2} + \frac{\mu - [f_a]_j}{[f_c]_j - [f_a]_j}(\lambda_{c2} - \lambda_{a2})\end{aligned} \tag{6.2}$$

which are the coordinates of the intersection point between f_a and f_c;

$$\begin{aligned}\lambda_1^{(2)} &= \lambda_{a1} + \frac{\mu - [f_a]_j}{[f_b]_j - [f_a]_j}(\lambda_{b1} - \lambda_{a1}) \\ \lambda_2^{(2)} &= \lambda_{a2} + \frac{\mu - [f_a]_j}{[f_b]_j - [f_a]_j}(\lambda_{b2} - \lambda_{a2})\end{aligned} \tag{6.3}$$

which are the coordinates of the intersection point between f_a and f_b;

$$\begin{aligned}\lambda_1^{(3)} &= \lambda_{c1} + \frac{\mu - [f_c]_j}{[f_b]_j - [f_c]_j}(\lambda_{b1} - \lambda_{c1}) \\ \lambda_2^{(3)} &= \lambda_{b2} + \frac{\mu - [f_b]_j}{[f_c]_j - [f_b]_j}(\lambda_{c2} - \lambda_{b2})\end{aligned} \tag{6.4}$$

which are the coordinates of the intersection point between f_b and f_c.

The solution of the two points at intersection will depend where the plane cuts the principal surface:

- If $[f_a]_j$ and $[f_c]_j \geq \mu$ and $[f_b]_j < \mu$, or vice versa, then the solutions will be equations 6.3 and 6.4.
- If $[f_b]_j$ and $[f_c]_j \geq \mu$ and $[f_a]_j < \mu$, or vice versa, then the solutions will be equations 6.2 and 6.3.
- If $[f_a]_j$ and $[f_b]_j \geq \mu$ and $[f_c]_j < \mu$, or vice versa, then the solutions will be equations 6.2 and 6.4.

Consider finding the intersection points for variable X when $\mu = 0.36$ in Figure 6.5 from the simulated blanket surface data. To illustrate the process of finding the intersection points in the two-dimensional space, consider four random tiles chosen in the λ_{grid} plot. The surface points of these tiles are given in Table 6.1:

TABLE 6.1: The points f_a, f_b and f_c of the X variable in each tile.

X	f_a	f_b	f_c
First tile	-0.66	-0.62	-0.63
Second tile	0.32	0.37	0.34
Third tile	0.36	0.42	0.39
Fourth tile	0.35	0.40	0.37

The *first* tile given in Figure 6.12 does not intersect with the plane $\mathcal{N}_{0.36}$. This is because all points f_a, f_b and $f_c < 0.36$.

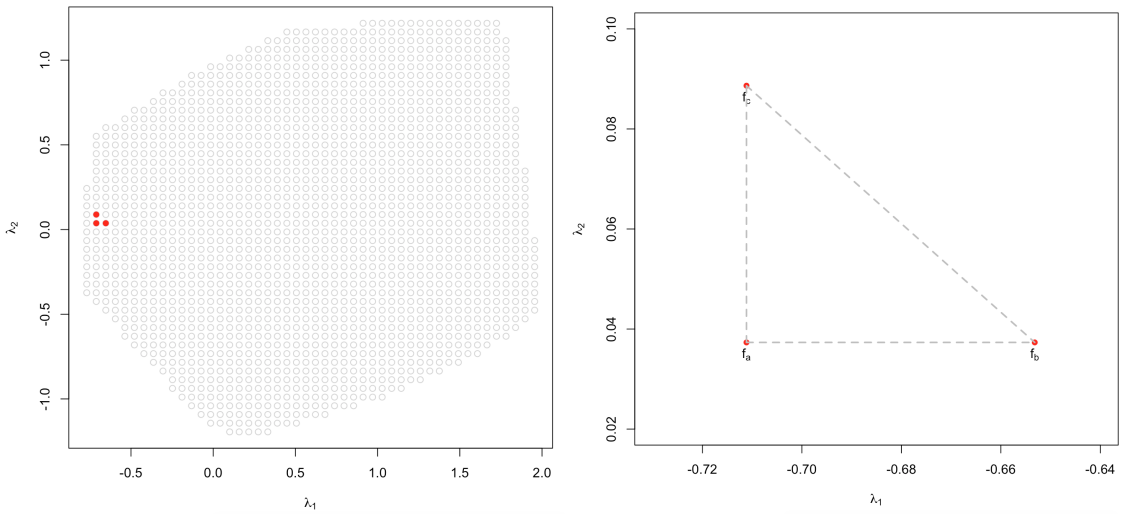

FIGURE 6.12: First tile in the λ_{grid} plot.

The second, third and fourth tiles all intersect with the plane $\mathcal{N}_{0.36}$ and therefore, give solutions of λ as shown by the purple points connected by a line in Figures 6.13, 6.14 and 6.15. Figure 6.16 displays the intersection of all four tiles in three dimensions.

The solution for each tile intersection can be explained as follows:

- The *second* tile intersects with the plane, as f_a, $f_c < 0.36$ and $f_b > 0.36$, thus giving the solutions of the two λ points: $\lambda^{(2)}$ and $\lambda^{(3)}$ from equations 6.3 and 6.4, respectively.
- The *third* tile intersects with the plane, as f_b, $f_c \geq 0.36$ and $f_a < 0.36$, thus giving the solutions of the two λ points: $\lambda^{(1)}$ and $\lambda^{(2)}$ from equations 6.2 and 6.3, respectively.
- The *fourth* tile intersects with the plane, as f_b, $f_c > 0.36$ and $f_a < 0.36$, thus giving the solutions of the two λ points: $\lambda^{(1)}$ and $\lambda^{(2)}$ from equations 6.2 and 6.3, respectively.

The solutions are found for all adjacent tiles on the surface intersecting with the plane $\mathcal{N}_{0.36}$ for the X variable. The final result will be a set of points in two dimensions that can be joined by a line to represent the intersection, shown in Figure 6.17. The process is done for different marker points μ, and all tiles in the grid. The result will be a set of lines, which will be referred to as *contour lines*. All points on a contour line will predict the same value μ for variable j.

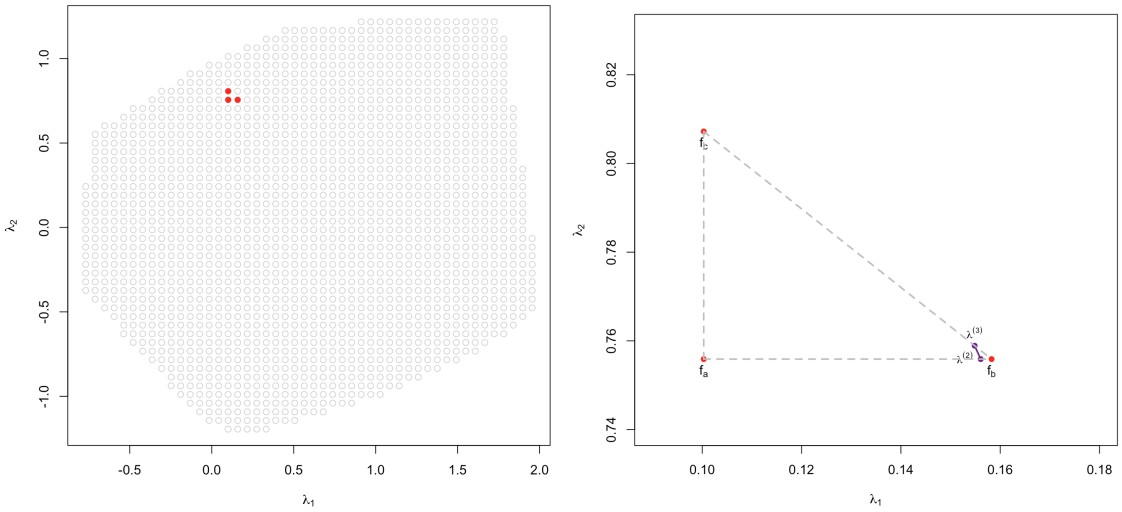

FIGURE 6.13: Second tile in the λ_{grid} plot.

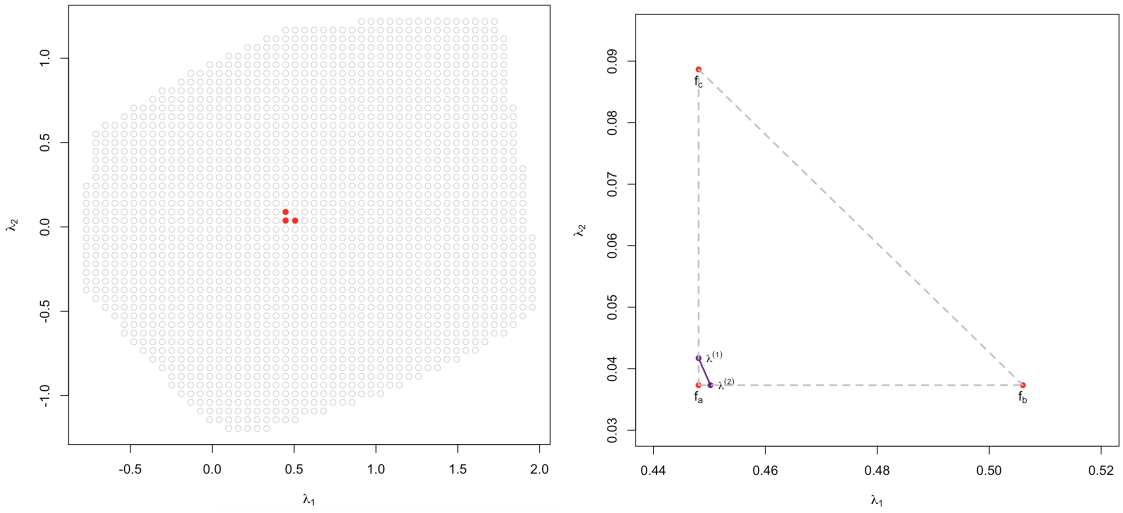

FIGURE 6.14: Third tile in the λ_{grid} plot.

Now consider computing all the contour lines for each variable in the blanket surface data. The number of markers for each variable is increased to 10. The higher the number of markers, the more contour lines there will be in the grid plot. These contours lines for a set of markers on the variables X, Y and Z are shown in Figures 6.18, 6.19 and 6.20, respectively.

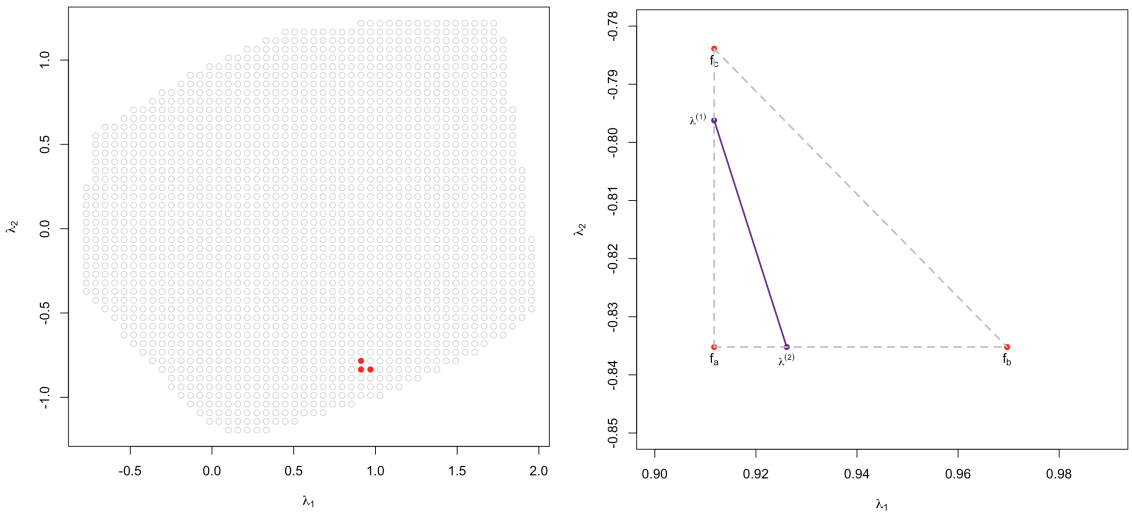

FIGURE 6.15: Fourth tile in the λ_{grid} plot.

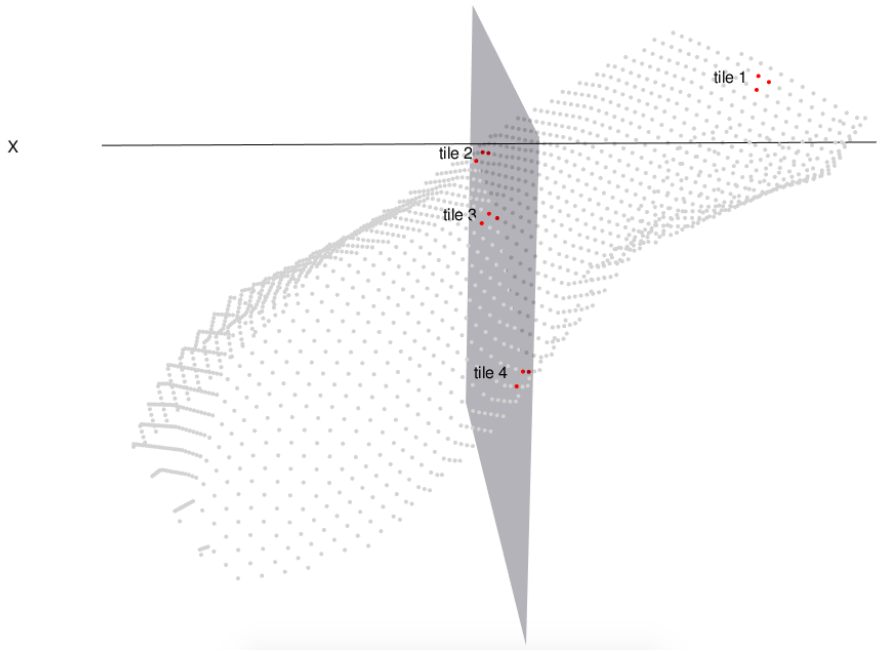

FIGURE 6.16: The intersection of $\mathcal{N}_{0.36}$ plane with four tiles in three-dimensions.

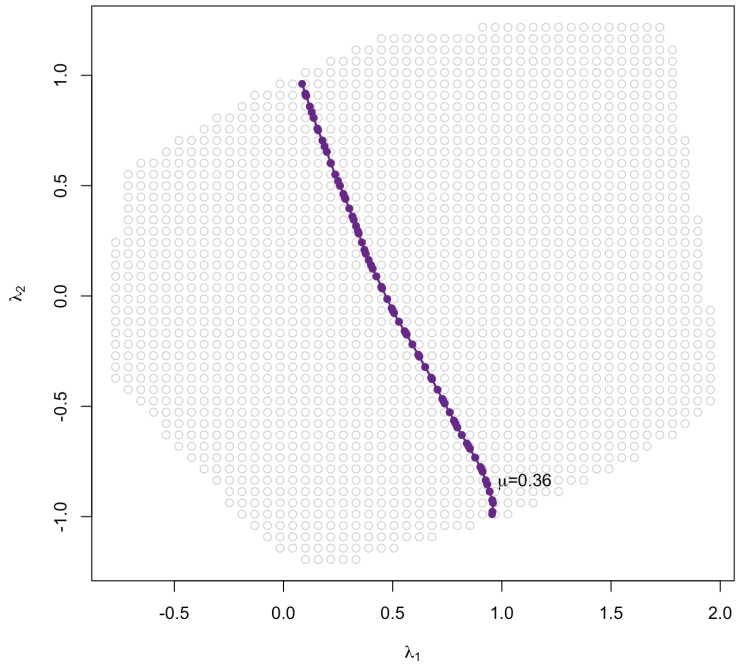

FIGURE 6.17: Intersection points from the intersection of the plane $\mathscr{N}_{0.36}$ with all tiles in the grid.

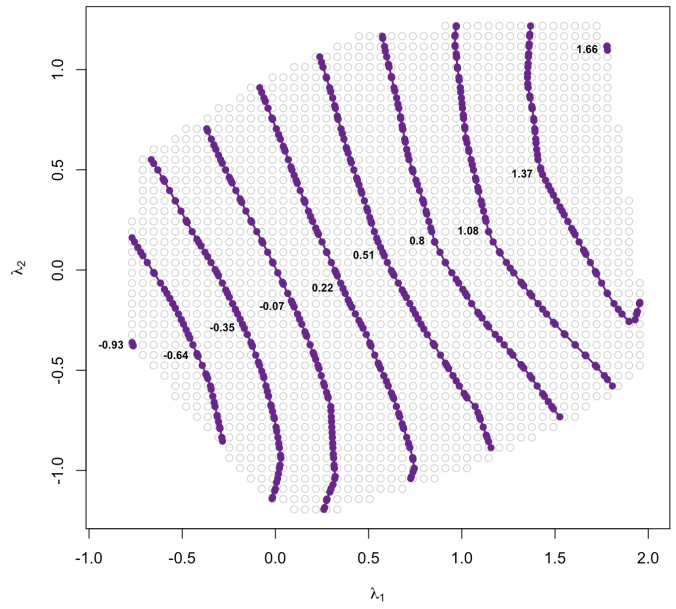

FIGURE 6.18: Contour lines of the variables X of the blanket surface data.

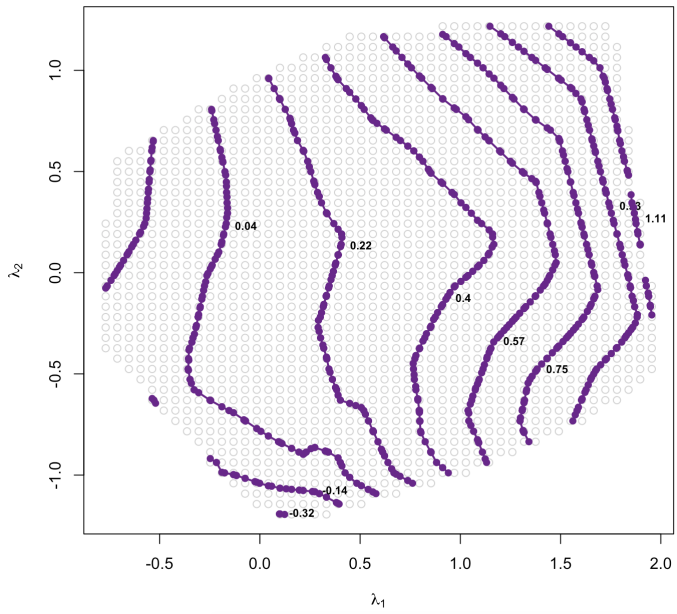

FIGURE 6.19: Contour lines of the variable Y of the blanket surface data.

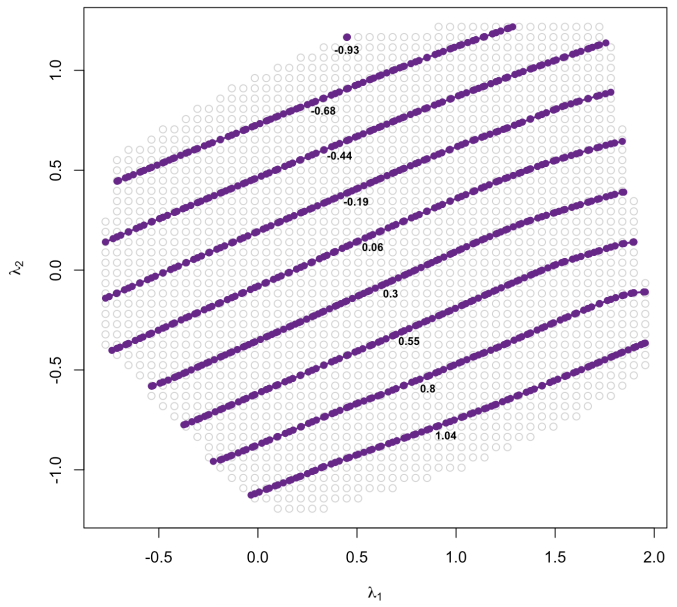

FIGURE 6.20: Contour lines of the variables Z of the blanket surface data.

6.4.1 Breast Tissue data

The contour lines are explained using a simulated data set. The following is an illustration of contour lines on a real data set. The Breast Tissue data is extracted from Dheeru and Taniskidou (2017) and consists of 106 observations on five variables:

- **IO**: Impedivity (ohm) at zero frequency.
- **PA500**: Phase angle at 500 KHz.
- **HFS**: High-frequency slope of phase angle.
- **DA**: Impedance distance between spectral ends.
- **IP**: Maximum of the spectrum.

A principal surface is fitted to the 106 data points in the Breast Tissue data. Figure 6.21 presents the representation of samples in two dimensions using $\lambda = (\lambda_1, \lambda_2)$ points, with five points labelled *1*, *2*, *3*, *4* and *5*. The f_{grid} and λ_{grid} are computed from the fitted principal surface.

Contour lines are constructed for each variable with marker points on λ_{grid} plots. These *contour plots* are illustrated in Figure 6.22 to Figure 6.26. The legend in each figure represent the marker points of the variable. Some variables have relatively linear contour lines, while others are highly nonlinear, and some of which form closed loops. Although one cannot visualise the five-dimensional data and principal surface, the sets of contour lines give an idea of what the surface will look like.

Consider predicting the five marked points representing observations in the data shown in the contour plots. The actual values for each observation under each variable are given in in Table 6.2. The contour lines can be used to predict samples, based on which region between the contour lines the sample points lie. For example, point *1* lies in the contours marked between 2200 and 2400 for variable IO (Figure 6.22), and its actual value is 2350. Point 3 lies in the contours marked between 0.05 and 0.10, but almost on the line of 0.10 for variable PA500 (Figure 6.23), and its actual value is 0.0885. This shows that the contour lines predict observations closely in the correct region, and will be the case for any variable and any set of data. This statement holds true because the planes are constructed at marker points on the original axis in the p-dimensional space to create contour lines on the surface, and a point on a surface must lie between two sets of contour lines representing marker values that come from the original axis.

When the number of variables is not significantly large, the contour plots provide a means of predicting observations in the data. However, contour plots need to be constructed for each variable. When there are many variables, the ideal situation would be to have a single plot, i.e. a biplot, with axes representing variables and points representing samples in the data.

The next chapter presents the idea of constructing a biplot axis from each set of contour lines representing a variable which is then presented in a single plot for the use of prediction.

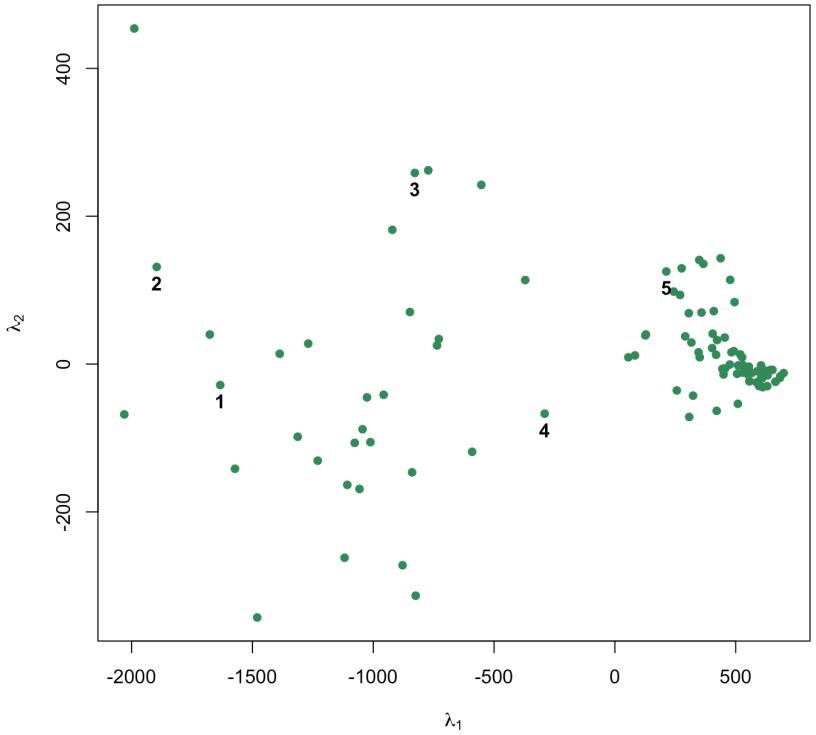

FIGURE 6.21: Representation of 106 observation by λ.

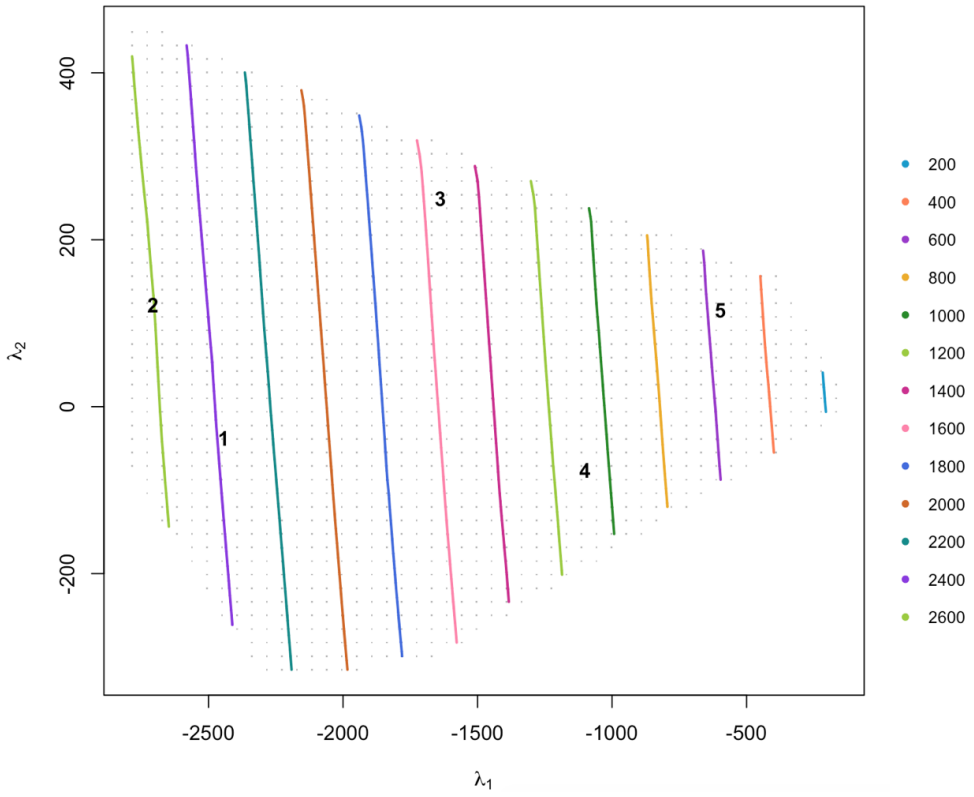

FIGURE 6.22: Contour lines for variable **IO**.

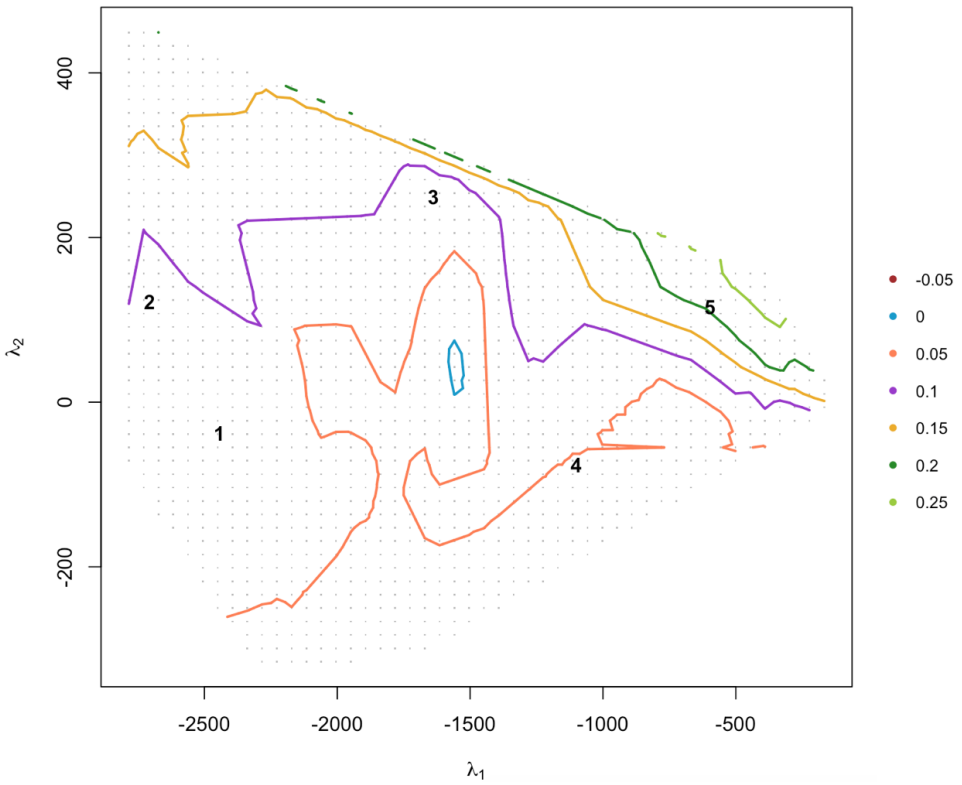

FIGURE 6.23: Contour lines for variable **PA500**.

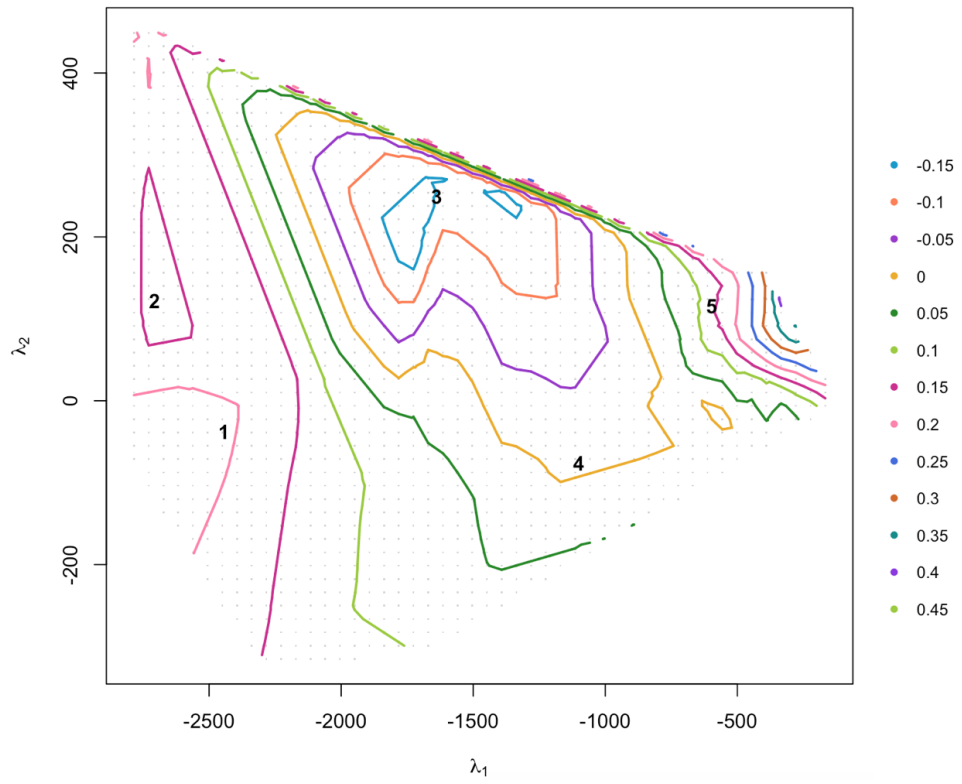

FIGURE 6.24: Contour lines for variable **HFS**.

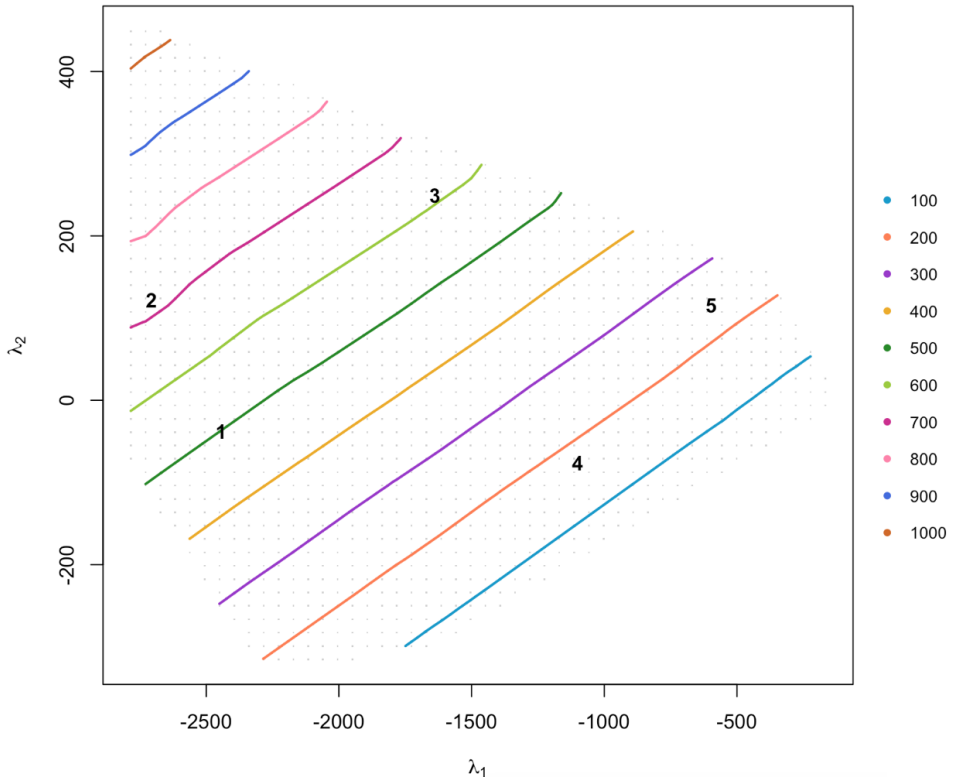

FIGURE 6.25: Contour lines for variable **DA**.

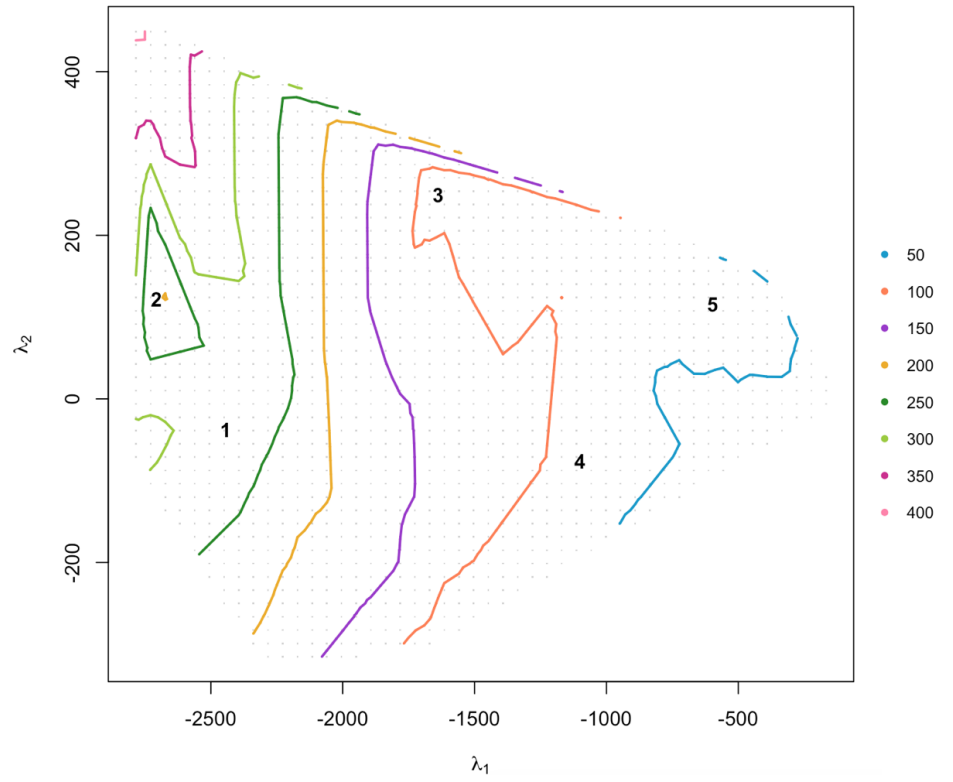

FIGURE 6.26: Contour lines for variable **IP**.

TABLE 6.2: Actual values for observations *1*, *2*, *3*, *4* and *5* in the Breast Tissue data.

Sample no.	I0	PA500	HFS	DA	IP
1	2350.00	0.0815	0.2716	515.2895	289.5691
2	2600.00	0.0700	0.0489	745.4744	154.1226
3	1535.85	0.0885	0.0042	637.3497	96.6104
4	1084.25	0.0735	0.0000	191.8980	66.5636
5	551.88	0.2325	0.0635	264.8049	77.7933

6.5 Summary

This chapter began by introducing the idea of a principal surface biplot, and what is required for the construction of the biplot, i.e. sample representation and biplot axes. The samples are represented by the λ coordinates of the principal surface. The construction of the biplot axes is shown through interpolation axes. However, it will be further highlighted in the next chapter that prediction biplot axes need to rather be constructed. The intersection spaces, or now referred to as contour lines, presented in this chapter, will be used to construct the biplot axes in the next chapter.

Chapter 7

Constructing Principal Surface Prediction Biplot Axes

7.1 Introduction

Chapter 6 introduces the concept of biplots based on principal surfaces. Samples are represented by λ coordinates from a fitted principal surface. Contour lines are constructed using the intersection of $\mathscr{N}_\mu$ planes and the principal surface, which are represented in a two-dimensional plot. This chapter explores three different designs of constructing a set of prediction biplot axes representing variables by using the sets of contour lines. The theory of these designs: circular projection, normal projection and back-projection originate from the literature of nonlinear biplots mentioned in Chapter 4. The chapters ends by selecting the best design for constructing principal surface biplot axes that will be utilised for the prediction of samples.

7.2 The design of circular projection

In Chapter 4 on nonlinear biplots, circular projection is briefly introduced. It is one of three different methods to obtain biplot prediction trajectories. Circular projection defines the marker μ on the intersection space as the orthogonal projection of the origin O, for a variable. The intersection spaces refer to the intersection between the biplot plane $\mathscr{L}$ and $\mathscr{N}_\mu$.

Consider from Chapter 4 Figures 4.3 and 4.4, that contain the intersection spaces $\mathscr{L} \cap \mathscr{N}$ at marker points $\mu = 2, 3$ and 4 for the variables X_1 and X_2 from data matrix 4.4, respectively. In circular projection, the origin O is orthogonally projected onto each intersection line, to represent the markers of each variable. This is illustrated in two dimensions in Figure 7.1 for the intersection spaces where $\mu = 2, 3$ and 4 for the variables X_1 and X_2. The projection is done for all marker points to form an axis representing the variable.

Figure 7.2 presents the nonlinear biplot axes constructed through circular projection for variables X_1 and X_2. In order to predict a sample, say a, a circle is constructed such that aO is the diameter. The diameter subtends a right angle at a point on each biplot axis where the circle intersects the axis. The predicted value for a corresponds to that point of intersection on each biplot axis. If the axis intersects the circle more than once, the intersection point closest to O is selected. Point a predicts a value of 4.5 for X_1 and 3.4 for X_2.

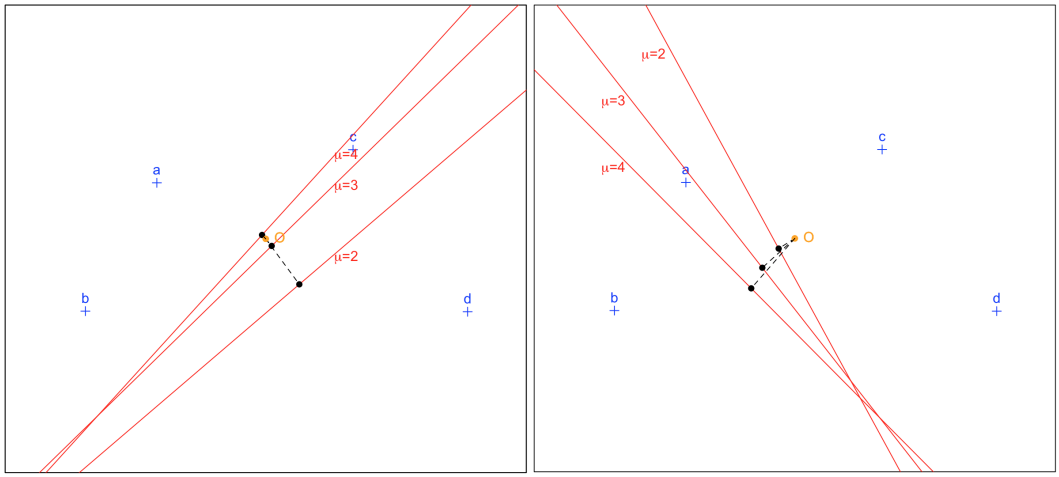

FIGURE 7.1: Orthogonal projection of point O onto each intersection space for $\mu = 2, 3$ and 4 for variables X_1 and X_2, respectively.

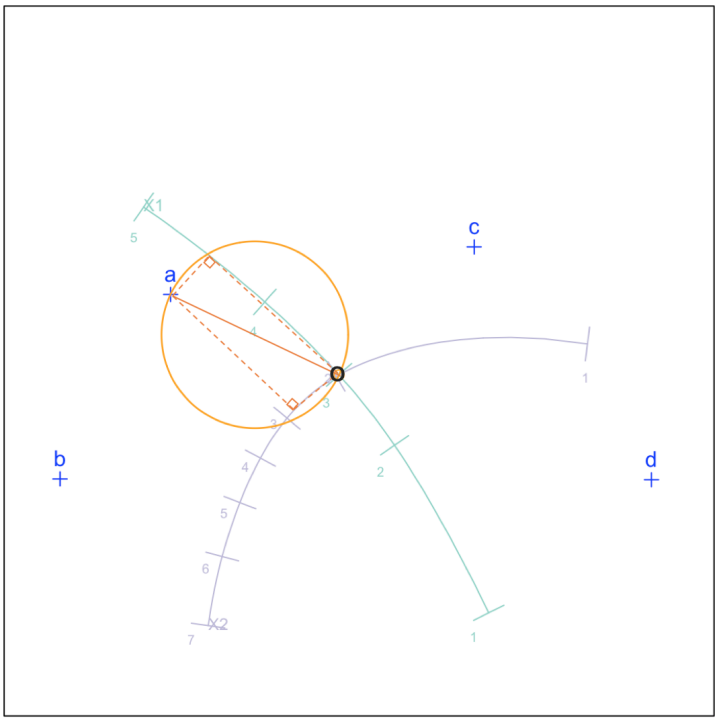

FIGURE 7.2: Circular projection of the point *a*, with diameter *aO* of the circle from *a* to the origin *O* and with dotted lines leading to the point of intersection between the circle and axes.

Now consider using this design of circular projection together with the contour lines. This will be illustrated using the simulated blanket surface data in Figure 6.1. First, the origin is projected onto each contour line. Since the values of λ in the contour plots are not centered around $\mathbf{0}$, the origin O will have different coordinates to that of $(0, 0)$. The origin O is chosen to be the average of the columns of the grid values λ_{grid}, so that O lies in the center of the plot. Figure 7.3 presents the projection of O onto each contour line.

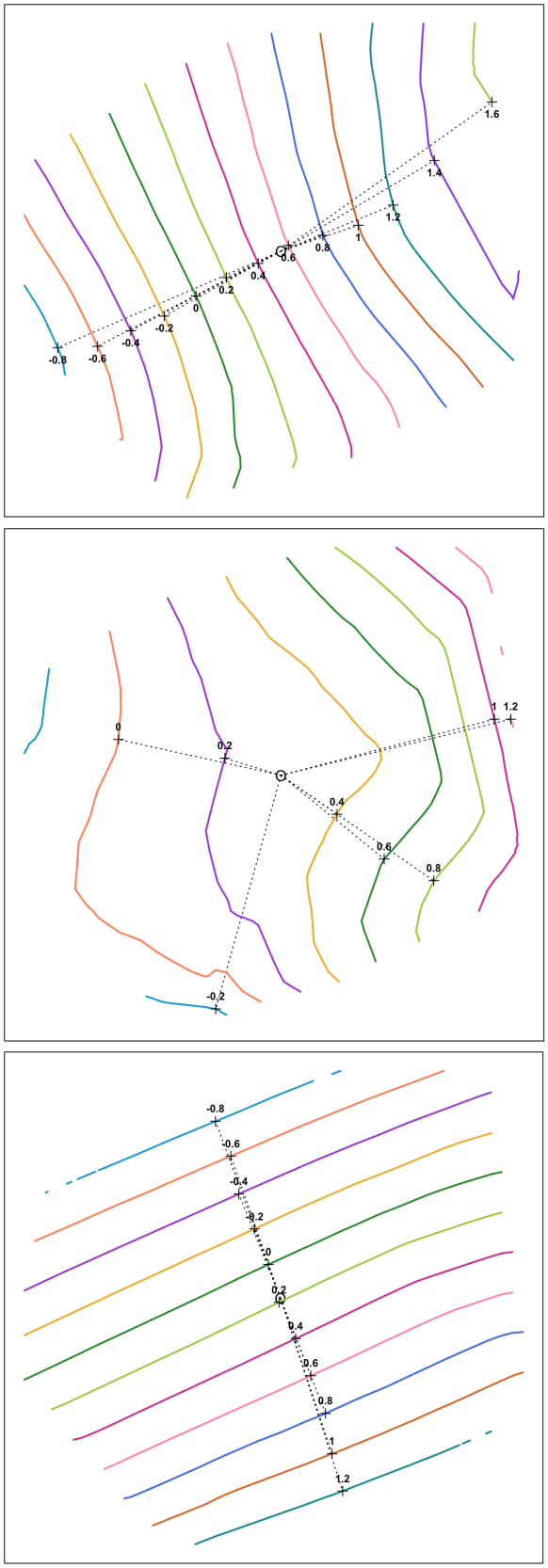

FIGURE 7.3: Projection of O onto each contour line for variables X, Y and Z in the blanket surface data, respectively. The projected points are labelled by the marker point μ for the respective variable.

Second, the projected points on the contour lines are joined to form a biplot axis. This is shown in one plot for all variables in Figure 7.4. The axes do not have marker points at the ends, because the points at the end of the axes are not necessarily a convenient marker point. Included in the figure is also the representation of the 100 sample points of the blanket surface data.

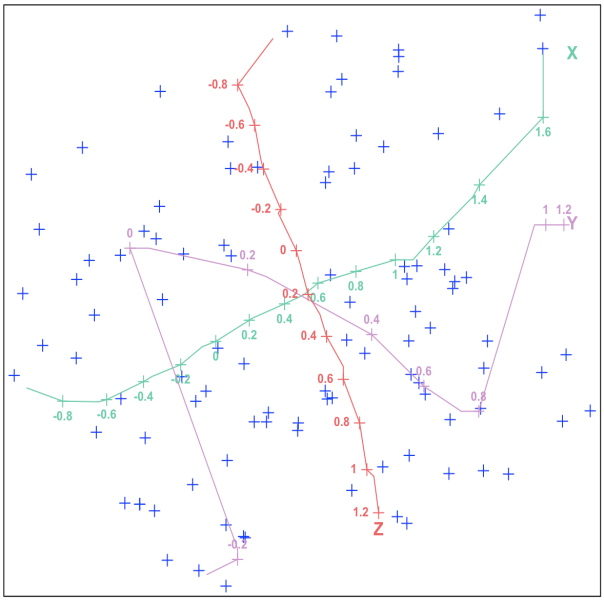

FIGURE 7.4: Prediction axes constructed using circular projection, for variables X, Y and Z in the blanket surface data.

Lastly, samples can be predicted by finding the intersection between the circles created at each point and the axes. Consider five points labelled a, b, c, d and e in Figures 7.5 and 7.6. The figures show the circles drawn with the diameter equal to the distance between the point to be predicted and O. The prediction of the points is found by the intersection of the circles and each axis. Table 7.1 summarises the predictions of the points a to e. The table provides the actual values of the points, the predicted values and a tick or cross mark to indicate whether the point is predicted in the correct region or not.

TABLE 7.1: Summary of predictions using circular projection for points a to e on circular projection axes. Tick marks (✓) indicate prediction in the correct region and cross marks (✗) indicate prediction in the incorrect region. The regions refer to the segment between the marker points on the biplot axis.

	Actual			Predicted			Accuracy		
	X	Y	Z	X	Y	Z	X	Y	Z
a	-0.768	-0.096	-0.113	-0.800	0.300	-0.180	✓	✗	✓
b	0.388	0.285	-0.582	0.600	0.300	-0.580	✗	✓	✓
c	1.310	0.445	0.189	1.250	0.300	0.200	✓	✗	✓
d	0.949	0.763	1.002	1.100	0.300	0.900	✓	✗	✓
e	-0.064	-0.084	1.045	-0.250	0.250	1.080	✓	✗	✓

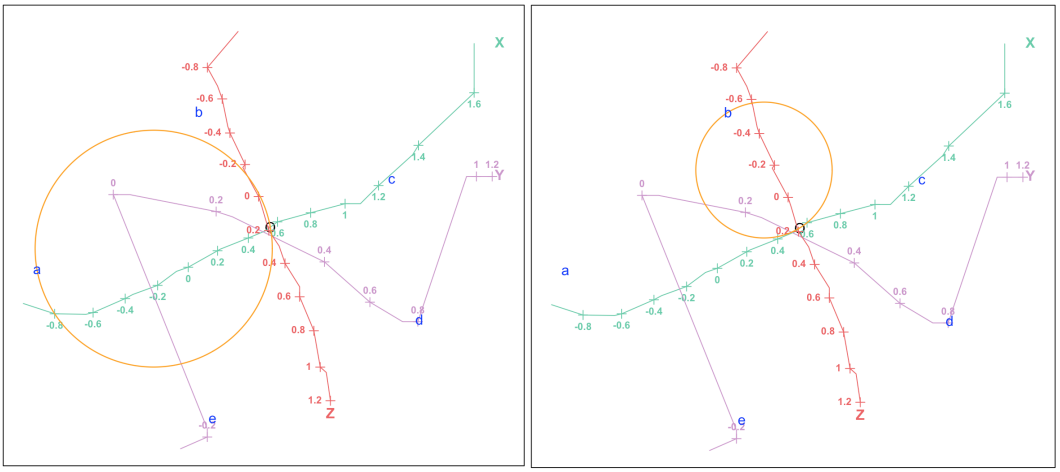

FIGURE 7.5: Circular projection for points a and b, respectively.

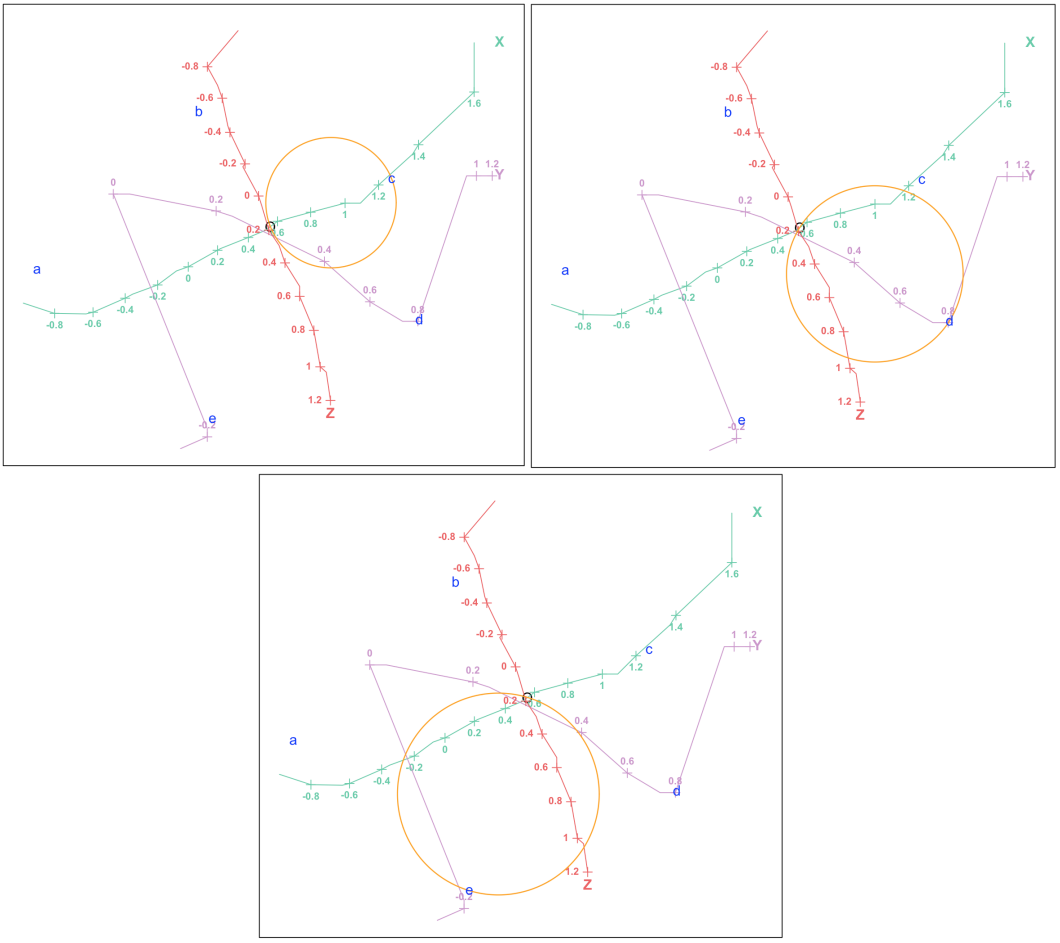

FIGURE 7.6: Circular projection for points c, d and e, respectively.

Although only five points are illustrated here, they provide an understanding of how other sample points are predicted. When predictions are computed on all the samples, the following conclusions are made:

- Axis Y predicted incorrectly far more than it did correctly. This could be due

to the nonlinear contour lines. As X and Z have more linear contour lines, their circular projection axes predicted better than Y.

- Axis X did better in predicting in the correct region, which could be due to the axes covering both ends of the biplot in Figure 7.4. Axis Z failed to predict in the correct region to the points lying at the bottom of the biplot, as the axis does not extend that far.

These conclusions are made using the prediction of points in the blanket surface data. When predictions through the circular projection biplot axes are made on sample points from the half-sphere data, the same conclusions are made.

7.3 The design of normal projection

One of the other methods to obtain biplot prediction trajectories for nonlinear biplots is normal projection. In normal projection, a trajectory is found such that it is normal to all intersection spaces for a variable. Again, the intersection spaces here refer to the intersection between the biplot plane $\mathscr{L}$ and $\mathscr{N}_\mu$. The mathematical explanation and detail of normal projection can be seen in Gower, Lubbe, and Le Roux (2011).

The analogy of normal projection can be applied to finding a trajectory from the contour lines in principal surfaces. The process will first start by finding the closest point on the r-th contour line from the origin, where r can be any value from 1 to the total number of contour lines. The closest point found will represent the marker point on the r-th contour line. Two sets of projections occur. First, the marker point on the r-th contour line is orthogonally projected onto the $(r+1)$-th contour line. That projected point on the $(r+1)$-th contour line is then projected onto the $(r+2)$-th contour line. This step is repeated until the last contour line. Second, the marker point on the r-th contour line is orthogonally projected onto the $(r-1)$-th contour line. That projected point on the $(r-1)$-th contour line is then projected onto the $(r-2)$-th contour line. This step is repeated until the first contour line. These sets of projections ensure that all marker points are represented in the two-dimensional space.

Consider the figures in Figure 7.7, that presents the construction of biplot axes using the design of normal projection for the blanket surface data. The marker points 0.60, 0.20 and 0.20 on the r-th contour lines for variables X, Y and Z are shown by the green points.

Labelled points *a*, *b*, *c*, *d* and *e* in Figure 7.7 are predicted by orthogonally projecting them onto the biplot axes and read off from the marker points. Table 7.2 summarises the predictions of the points *a* to *e*.

The normal projection method performs better on all variables X, Y and Z than it does in circular projection.

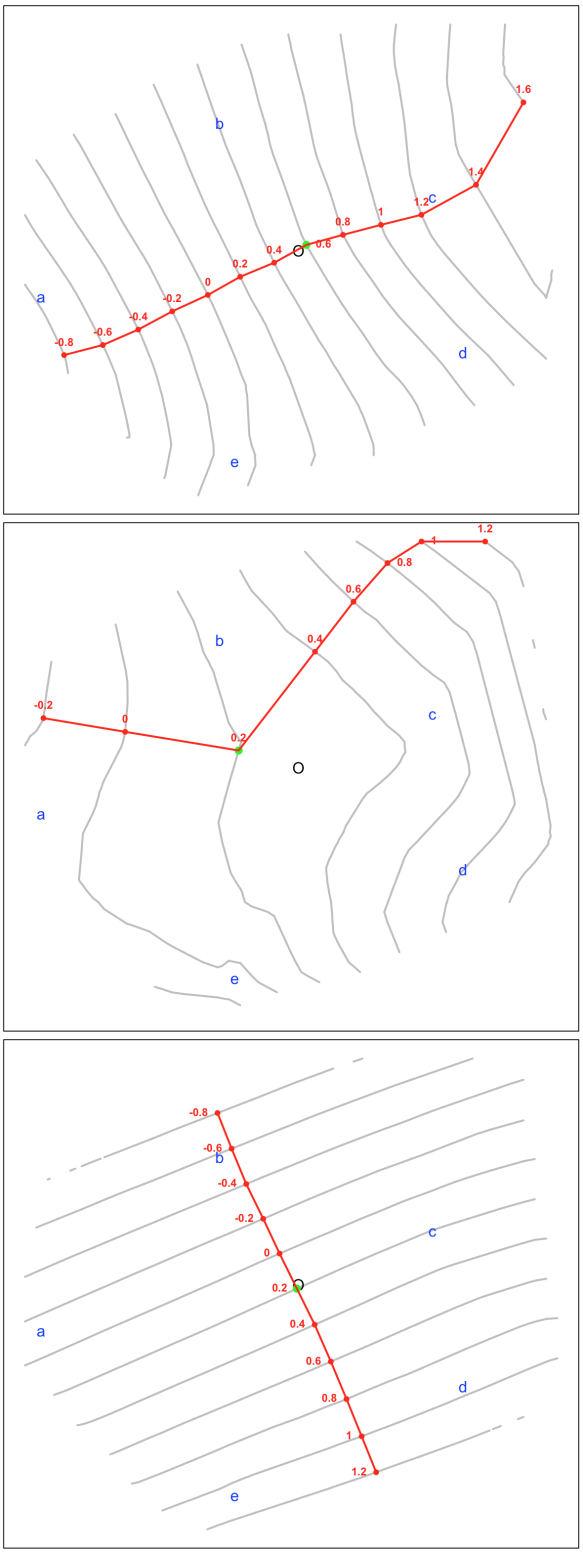

FIGURE 7.7: Normal projection biplot axes for variables X, Y and Z, respectively from the blanket surface data.

TABLE 7.2: Summary of predictions using normal projection for points a to e on normal projection axes. Tick marks (✓) indicate prediction in the correct region and cross marks (✗) indicate prediction in the incorrect region. The regions refer to the segment between the marker points on the biplot axis.

	Actual			Predicted			Accuracy		
	X	Y	Z	X	Y	Z	X	Y	Z
a	-0.768	-0.096	-0.113	-0.800	-0.100	-0.100	✓	✓	✓
b	0.388	0.285	-0.582	0.400	0.300	-0.570	✓	✓	✓
c	1.310	0.445	0.189	1.220	0.500	0.200	✓	✓	✓
d	0.949	0.763	1.002	1.100	0.250	0.950	✓	✗	✓
e	-0.064	-0.084	1.045	-0.320	0.200	1.100	✓	✗	✓

7.4 The design of back-projection

The third and final method to obtain biplot prediction trajectories for nonlinear biplots is back-projection. From Chapter 4, it is based on finding the point on the intersection $\mathscr{L} \cap \mathscr{N}$, nearest to the marker μ on the embedded original variable axis (Gower, Lubbe, and Le Roux, 2011). This analogy can be applied to the principal surfaces.

The method can be explained as follows:

1. Find a point on a contour line of variable j, that is closest to the respective marker point μ in the p-dimensional space.
2. The point that lies on the contour line that is closest to μ can be represented in the two-dimensional plot, to represent the marker μ for the j-th variable.
3. When all points on each contour line in the two-dimensional space are joined, they form a biplot prediction axis for variable j.

Creating prediction axes in this way can seem quite similar to that of creating interpolation axes in Section 6.3. However, the main difference is where μ is represented on the surface. Figure 7.8 summarises this difference. In interpolation, μ is orthogonally projected onto the principal surface shown by the red dotted line. In the case of prediction, the representation of μ is the point that lies on the contour line of the principal surface that is closest to μ of the j-th variable. This is shown by the orange dotted line.

7.4.1 Approximation

Currently, the contour lines calculated in the previous chapter are defined in the two-dimensional space. The contour lines will need to be found in the p-dimensional space, in order to find the marker point μ onto the respective contour line, as shown in Figure 7.8. In Chapter 2, the values of λ in the two dimensions are given by the principal surface $f(\lambda)$ in p-dimensions multiplied by ϑ_2, where ϑ_2 consists of the first two columns of the right singular vectors of $f(\lambda)$. This is given by the equation $\lambda = f(\lambda)\vartheta_2$. The same applies to the grid values, λ_{grid} and f_{grid} from equation 6.1.

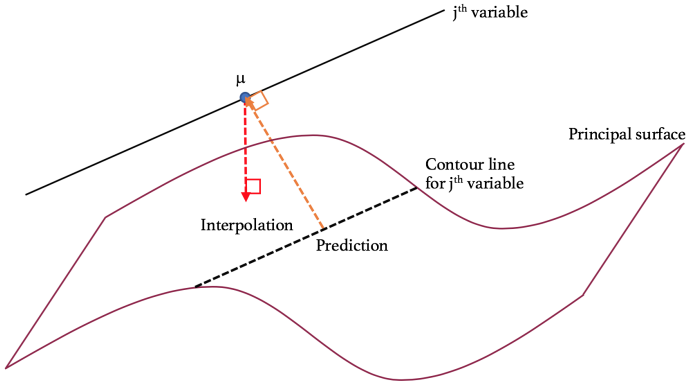

FIGURE 7.8: Difference in projection of μ on the principal surface between interpolation and back-projection (for prediction).

In the case of finding the contour points on the principal surface in p-dimensions, it therefore, seems intuitive to use the equation of λ, by rearranging the formula: $f(\lambda) = \lambda\vartheta_2'$. The contour points in p-dimensions will be given by the contour points in two dimensions multiplied by ϑ_2. However, calculating it this way will result in the contour points to lie on a flat plane, instead of contour points on the surface $f(\lambda)$. This is true because of the linear transformation $f(\lambda) = \lambda\vartheta_2'$ as shown in Chapter 3, $\mathbf{XV}_2$ (equation 3.6) are used to represent the projected sample points onto a principal component plane (i.e a flat plane), or also known as the biplot plane in terms of PCA biplots. The right singular vectors of a matrix forces the outcome to lie on a linear plane. This is illustrated in Figure 7.9.

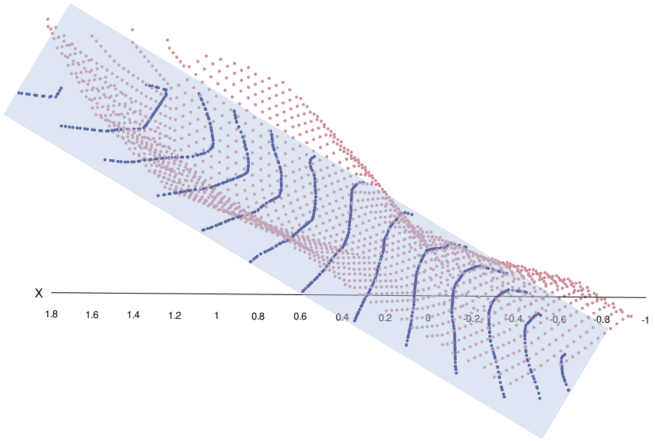

FIGURE 7.9: Contour points for variable X shown in blue are calculated by $f(\lambda) = \lambda\vartheta_2'$ in p-dimensions. The contour points are shown not to lie on the principal surface, but rather lie on a flat surface, as shown by the blue plane.

Given this shortfall, an approximation of the contour lines in the two-dimensional space will be used to find the contour lines in the p-dimensional space. This is done by finding the closest λ_{grid} point to each contour point for a marker μ of a variable. The coordinates of the closest λ_{grid} points will be used to represent the

contour points of the same coordinates in f_{grid}. Figure 7.10 presents the approximation of the contour lines for variable X in the blanket surface data. The approximated contour points are close to the calculated contour points, as intended. Figure 7.11 presents approximated contour points for variables Y and Z, respectively.

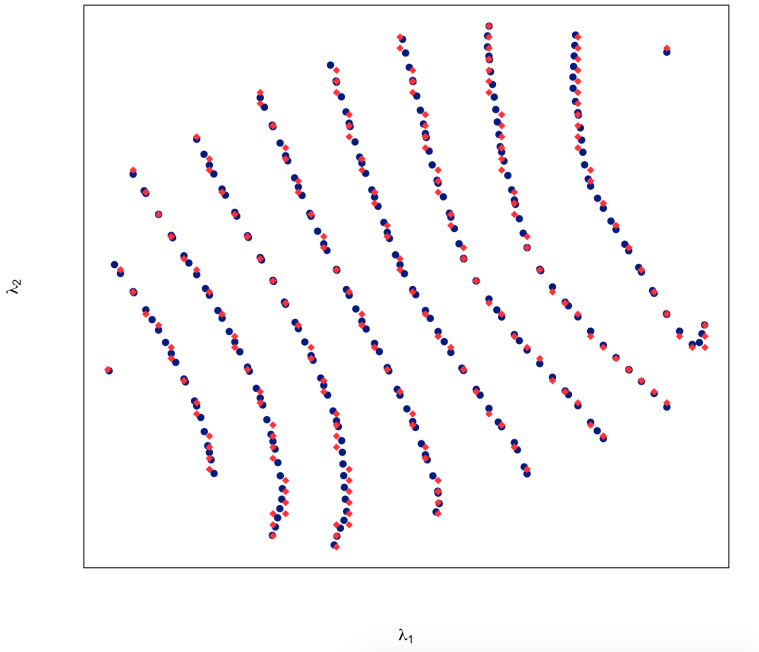

FIGURE 7.10: Approximation of contour points are shown in red, whereas blue points represent calculated contour points; for variable X in the blanket surface data.

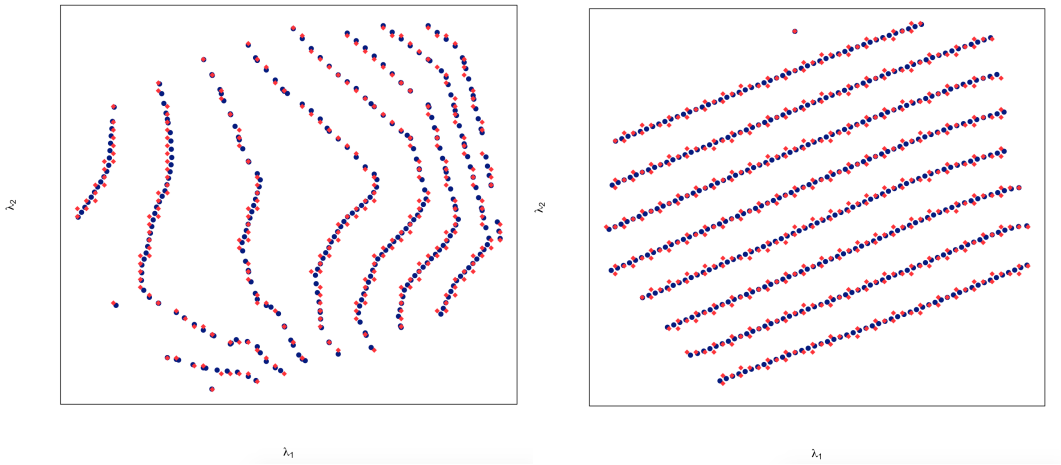

FIGURE 7.11: Approximation of contour points are shown in red, whereas blue points represent calculated contour points; for variables Y and Z in the blanket surface data, respectively.

The approximation of contour points that have λ_{grid} coordinates, can now be represented in the p-dimensional space by their respective f_{grid} coordinates. Closest

points on each contour line to the respective marker points on the j-th axis are found in p-dimensions. The points which lie on the principal surface will form the j-th biplot axis for variable j. The points in the two-dimensional space are joined by segments to form the j-th axis in the biplot. This is illustrated in Figures 7.12 and 7.13 for variable X.

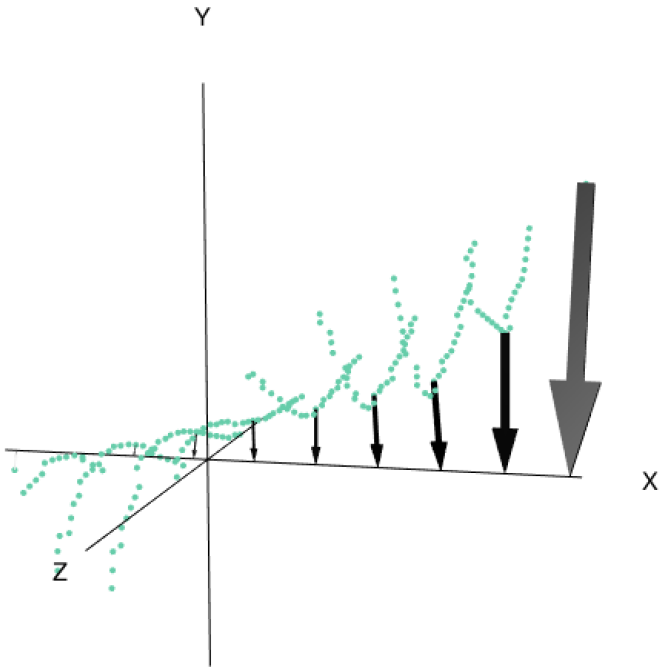

FIGURE 7.12: Projection of marker points on X onto respective contour lines, indicated by arrows. The size of the arrows is proportional to the distance. Blue line on principal surface represents biplot axis in three-dimensions.

Figure 7.14 presents the back-projection axes for variables Y and Z, respectively. Some marker points in the two-dimensional plot tend to be further away from other marker points. For example, the marker points 0.90 and 1.00 for axis Y are at the far end of the plot. This tends to happen as the surface is not linear, and the point chosen on the contour line is the orthogonal projection of the point 0.90 on the original axis. The marker point does not depend on where the previous marker point (0.80) lies, whether it is furthest or closest to it.

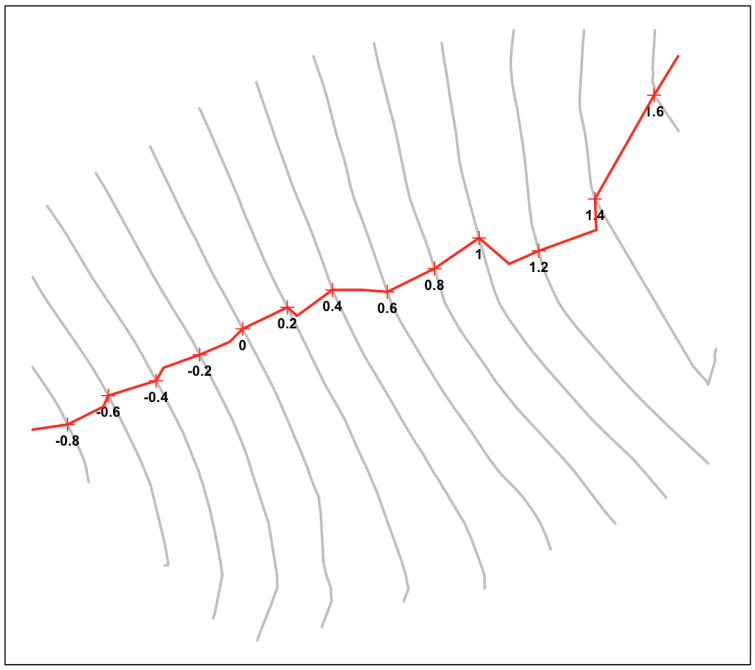

FIGURE 7.13: Constructed biplot axes for variable X with labelled marker points. The grey lines are approximated contour lines.

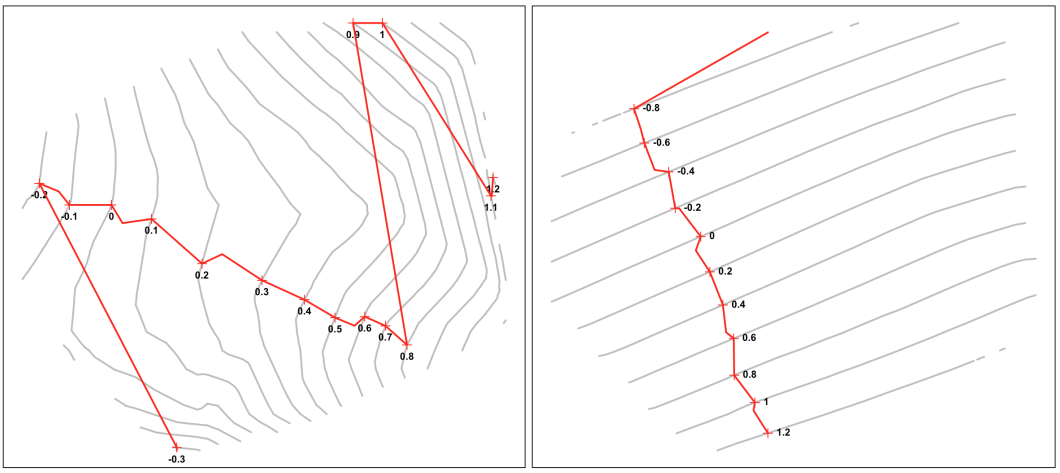

FIGURE 7.14: Constructed biplot axes for variables Y and Z with labelled marker points. The grey lines are approximated contour lines.

All axes are then plotted in a single plot with sample points in Figure 7.15. Samples a to e from previously, displayed in Figure 7.16 with back-projection axes can be predicted using normal projection. A summary of the predictions are given in Table 7.3.

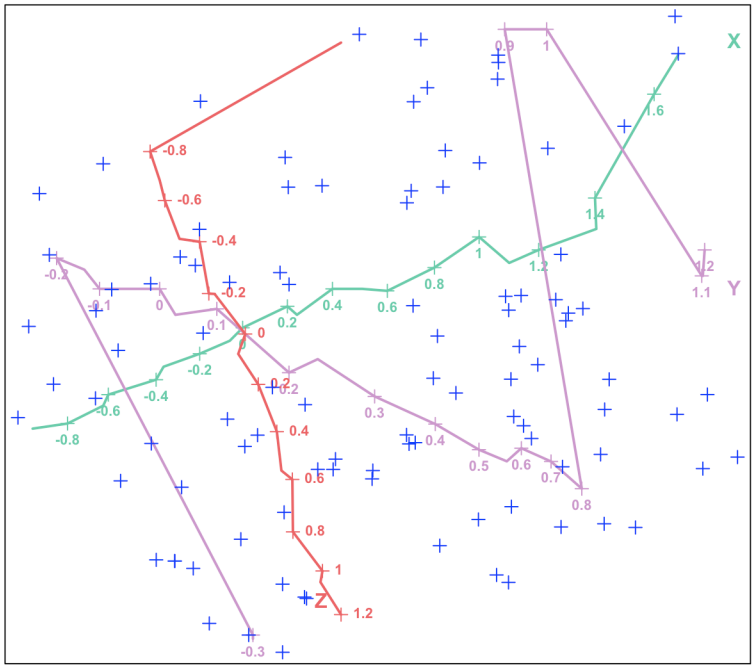

FIGURE 7.15: Prediction axes constructed using back-projection for variables X, Y and Z in the blanket surface data.

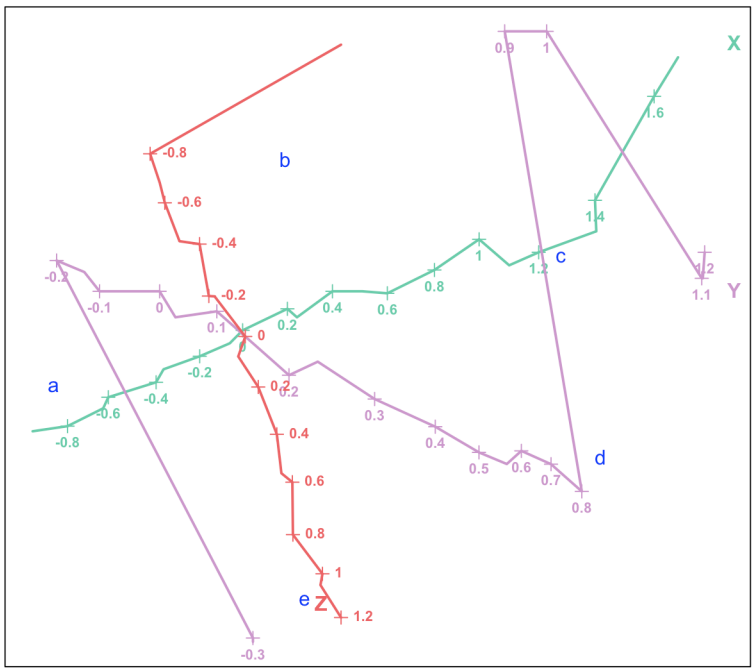

FIGURE 7.16: Normal projection of points a to e in the blanket surface data, with back-projection axes.

TABLE 7.3: Summary of predictions using normal projection for points a to e on back-projection axes. Tick marks (✓) indicate prediction in the correct region and cross marks (✗) indicate prediction in the incorrect region. The regions refer to the segment between the marker points on the biplot axis.

	Actual			Predicted			Accuracy		
	X	Y	Z	X	Y	Z	X	Y	Z
a	-0.768	-0.096	-0.113	-0.790	-0.225	-0.020	✓	✗	✓
b	0.388	0.285	-0.582	0.400	0.100	-0.650	✓	✓	✓
c	1.310	0.445	0.189	0.850	0.850	0.180	✓	✗	✓
d	0.949	0.763	1.002	0.810	0.810	0.800	✓	✓	✓
e	-0.064	-0.084	1.045	-0.300	-0.300	1.100	✓	✓	✓

When comparing the three different designs, samples a to e predicted the poorest through circular projection axes. The tie for these samples lies between the normal and back-projection axes. The normal projection axes does worse for samples d and e, and the back-projection axes does worse for a and c, both for variable Y, indicated by the ✗'s in Table 7.2 and 7.3, respectively. The test is to determine how the prediction of all samples perform under these designs.

When all points are predicted with back-projection axes, they have shown to overall predict better than the design of axes through circular projection and normal projection. There are however, points in the data that predict in the incorrect region, as seen for some of the points in Table 7.3. An accurate predictability score that will take into account the actual predictions will be shown in the next chapter. Although only shown here that the back-projection axes predict samples better on the simulated blanket surface data, it is the case for the half-sphere data in Chapter 2 and other sets of data that will be presented later in Chapter 9, where the principal surface biplot will be applied.

7.5 Summary

The focus of this chapter was on constructing prediction biplot axes for the principal surface biplot through three different designs. In practice, although for a limited number of data sets, it was found that the biplot axes constructed through back-projection predict samples the best. Now that samples can be represented and biplot axes constructed, the principal surface biplot will be defined in the next chapter. The next chapter will also look at a predictability score that will look at actual predictions of samples, and will demonstrate the impact of centering the data has on predictability.

Chapter 8

Prediction Accuracy in a Principal Surface Biplot

In this chapter, the principal surface biplot will be first officially defined. The focus of this chapter will be on the prediction accuracy of samples in such a biplot compared to the simplest type of biplot; the PCA biplot. The PCA biplot is chosen for comparison, as it proved to perform the best prediction of samples in the data sets used in this chapter, when compared to the other biplots from previous chapters. The comparison will be made by taking into account the actual values of samples in the data and the predicted values from the biplot. Furthermore, a detailed motivation for centering the principal surface before constructing the biplot, will be illustrated using the prediction accuracy.

8.1 Introduction

A principal surface biplot will display (1) the biplot axes constructed through the design of back-projection for all variables and (2) the sample points of the data represented by λ from the principal surface. The projection of a sample point on the biplot is done through normal projection. This follows the same technique of prediction in spline-based nonlinear biplots and normal projection in nonlinear biplots.

Figure 8.1 presents the principal surface biplot with biplot axes for variables X, Y and Z of the simulated blanket surface data in Chapter 7. The number of markers are increased to create finer marker points on the biplot axes. The biplot axis for variable Y tends to be more nonlinear, due its nonlinear contours, when compared to the axes of X and Z.

8.2 Squared standardised prediction errors

The principal surface biplot can be utilised in the same manner as the other biplots discussed in the previous chapters, in terms of predicting sample points in a data set. The squared standardised prediction errors will be used to compare the prediction accuracy between the PCA and the principal surface biplot when constructed on the blanket surface data.

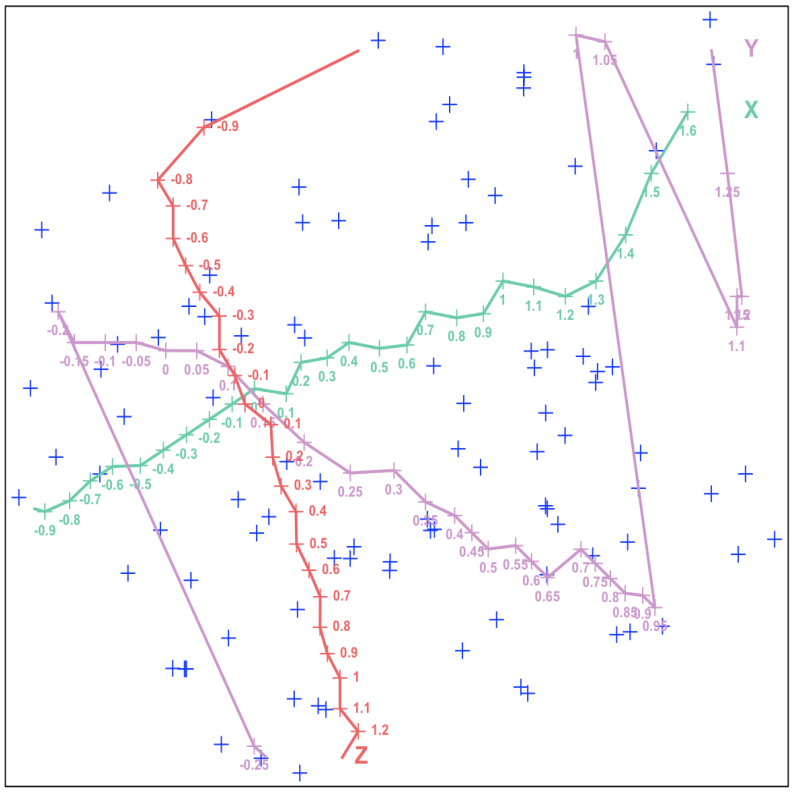

FIGURE 8.1: Principal surface biplot on blanket surface data.

The squared standardised prediction error (SSPE) is given by the sum of the squared differences between the actual observations and the predictions from the respective biplot for a variable divided by the standard deviation of the actual observations,

$$\text{Squared standardised prediction error (SSPE)} = \sum_{i=1}^{n} \left(\frac{x_i - \hat{x}_i}{sd} \right)^2. \tag{8.1}$$

Table 8.1 presents the SSPE values for variables X, Y and Z on the PCA and principal surface biplot. The PCA biplot has a lower SSPE than that of the principal surface biplot, indicating better prediction accuracy of samples on all variables.

TABLE 8.1: SSPE from PCA and principal surface (PS) biplots, for variables X, Y and Z from the blanket surface data.

	PCA	PS
X	1.688	7.645
Y	21.729	64.213
Z	0.055	19.346
$\sum$	23.472	91.204

Typically before constructing any biplot for a data set, variables are centered. Centering of data can be also done when constructing a principal surface biplot. Therefore, in an attempt to improve on the prediction accuracy of the principal surface biplot, the principal surface is first centered before constructing the biplot. This is motivated in the next section.

8.3 Centering the surface

To motivate the idea of centering the surface in order to improve the prediction accuracy of the principal surface biplot, data from Chapter 3 described in Figure 3.1 and Table 3.1 will be used. A principal surface is fitted on the original data, shown in Figure 8.2. The 25 sample points are represented by the λ coordinates of the principal surface, shown in Figure 8.3.

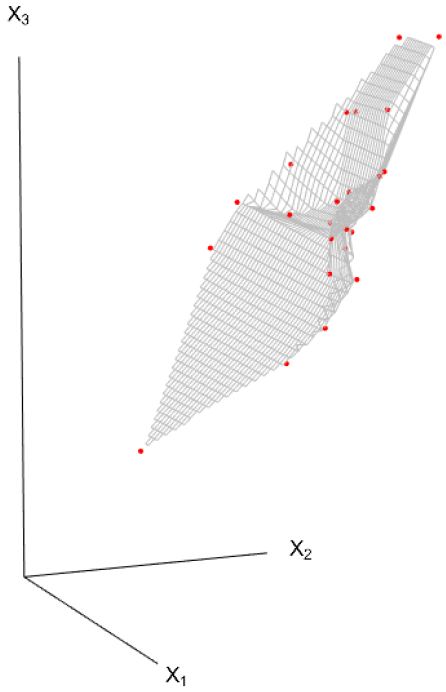

FIGURE 8.2: Principal surface on data in Table 3.1.$f(\lambda)$ given by red points.

The principal surface biplot of the 25 samples is shown in Figure 8.4. As one can see, the biplot axes for variable X_2 and X_3 lie on the edges of the biplot. This is because, if one could try to imagine for any variable from Figure 8.2, first the set of contour lines on the principal surface. Then the points chosen on the contour lines that are closest to the marker points on the variable axis, will lie on the edge of the surface. This therefore, creates a biplot axis that will lie on the edge of the biplot.

When the f_{grid} values of the principal surface are centered, as shown in Figure 8.5, the representation of sample points given by λ coordinates and the contour lines for each variable will remain unchanged. However, because of the centered axes, the values of the marker points will be different. That means, the points

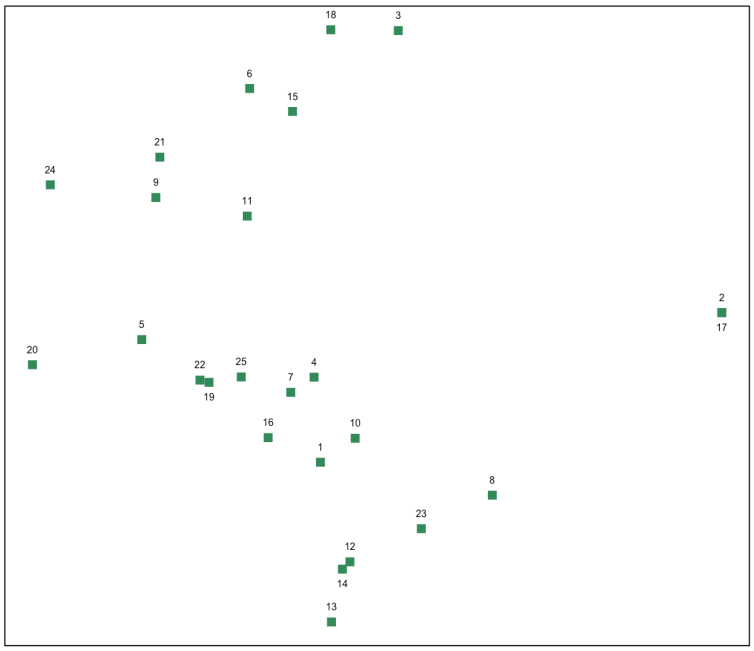

FIGURE 8.3: Representation of 25 sample points from the data in Table 3.1, given by the λ coordinates of fitted principal surface.

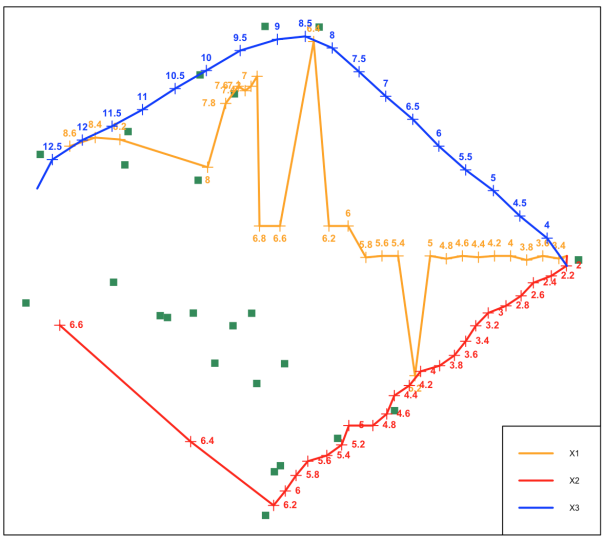

FIGURE 8.4: Principal surface biplot of the data in Table 3.1.

found on the contour lines to represent marker points on a biplot axis will have different values. Using the biplot axis to predict sample points in the data, the original marker points on the axis need to be used. Therefore, once the biplot axis is formed from the points on the contour lines, its marker points are adjusted according to the centering factors of the centered principal surface. The principal surface biplot when the principal surface is centered is shown in Figure 8.6.

The biplot axes in the principal surface biplot when the surface is centered in Figure 8.6 looks distinctly different to the principal surface biplot axes when the surface is not centered. Samples are predicted in each type of biplot for variables X_1,

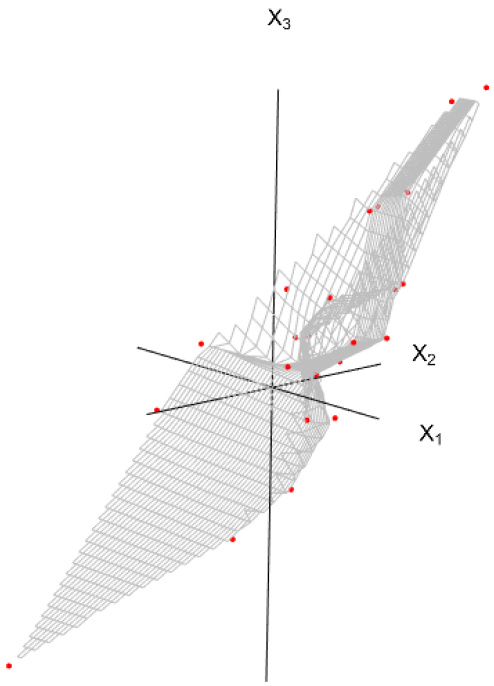

FIGURE 8.5: Centered principal surface on the data in Table 3.1. $f(\lambda)$ centered according to f_{grid} centering, shown by red points.

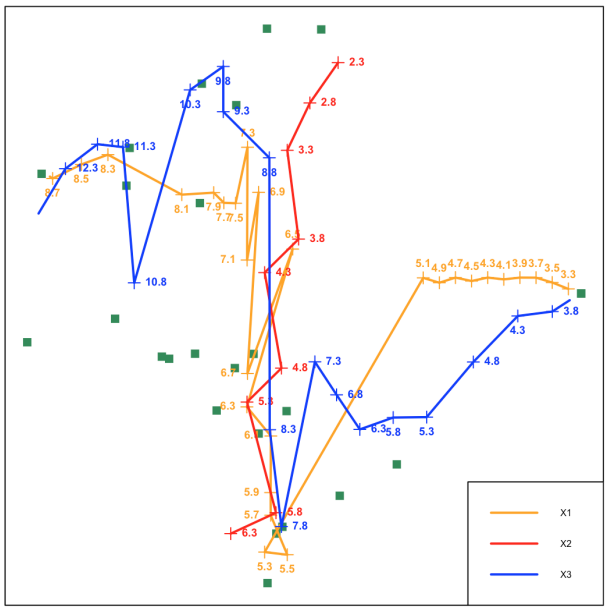

FIGURE 8.6: Principal surface biplot of Table 3.1, when surface is centered.

X_2 and X_3 and their predictions are shown in Tables 8.2, 8.3 and 8.4, respectively. Included in the tables are the SSPE values for each variable in each biplot. The principal surface biplot (when the surface is centered) in all three tables predict samples better than the principal surface biplot (when the surface is not centered).

This is especially true for variable X_2, thus showing the impact on prediction accuracy in the biplot when the surface is centered.

TABLE 8.2: Individual sample predictions and the SSPE for variable X_1 in Table 3.1 for PCA, principal surface (PS) (uncentered) and principal surface (PS) (centered) biplots.

	X_1: actual values	PCA	PS (uncentered)	PS (centered)
1	5.419	6.221	6.170	6.134
2	3.130	3.634	3.300	3.590
3	6.128	6.605	6.460	6.460
4	6.781	6.691	7.890	6.669
5	7.347	7.836	8.288	8.289
6	7.208	7.554	7.904	7.907
7	7.039	6.652	7.890	6.741
8	5.466	4.945	5.367	5.055
9	7.723	8.141	8.182	8.183
10	7.110	6.320	6.170	6.179
11	8.136	7.392	7.903	7.896
12	6.287	5.831	6.170	5.269
13	4.649	5.625	6.170	5.311
14	5.799	5.988	6.170	5.604
15	8.085	7.407	7.602	7.320
16	6.158	6.874	7.890	6.740
17	3.557	3.934	3.300	3.590
18	6.964	7.311	7.294	7.320
19	7.706	7.408	7.963	7.923
20	8.348	8.709	8.460	8.750
21	8.191	8.389	8.157	8.158
22	7.999	7.364	7.987	7.940
23	6.211	5.424	5.897	5.137
24	8.776	9.056	8.460	8.750
25	8.079	6.986	7.906	7.600
	SSPE	3.829	5.274	3.608

TABLE 8.3: Individual sample predictions and SSPE for variable X_2 in Table 3.1 for PCA, principal surface (PS) (uncentered) and principal surface (PS) (centered) biplots.

	X_2: actual values	PCA	PS (uncentered)	PS (centered)
1	5.054	5.216	6.169	5.443
2	1.783	1.885	1.920	2.160
3	2.173	2.269	4.900	2.450
4	4.753	4.735	6.199	4.679
5	5.893	5.992	6.390	6.390
6	3.744	3.814	6.390	3.150
7	5.214	5.135	6.259	5.400
8	4.493	4.388	4.408	2.160
9	4.708	4.792	6.390	6.390
10	4.988	4.828	5.787	4.900
11	4.392	4.242	6.390	4.650
12	5.909	5.817	5.893	5.890
13	7.198	7.395	6.142	5.903
14	6.120	6.158	5.966	5.890
15	3.235	3.098	6.365	3.150
16	5.743	5.888	6.326	5.400
17	2.026	2.102	1.920	2.160
18	2.470	2.539	5.400	3.058
19	5.401	5.341	6.390	6.390
20	6.697	6.770	6.390	6.390
21	4.255	4.294	6.390	4.650
22	5.925	5.796	6.390	6.390
23	5.105	4.947	5.158	5.652
24	5.350	5.406	6.390	6.390
25	5.668	5.447	6.390	5.400
	SSPE	0.159	25.579	6.864

TABLE 8.4: Individual sample predictions and SSPE for variable X_3 in Table 3.1 for PCA, principal surface (PS) (uncentered) and principal surface (PS) (centered) biplots.

	X_3: actual values	PCA	PS (uncentered)	PS (centered)
1	8.711	8.202	9.630	7.920
2	3.386	3.066	3.590	3.590
3	8.174	7.871	8.172	8.890
4	8.732	8.789	9.616	7.922
5	11.303	10.993	11.743	10.810
6	10.076	9.856	10.182	9.919
7	8.609	8.855	9.850	7.920
8	5.597	5.927	6.043	5.654
9	11.357	11.092	11.344	10.991
10	7.733	8.234	8.890	7.589
11	9.265	9.737	10.355	9.850
12	7.488	7.778	8.890	7.456
13	8.574	7.954	9.671	7.566
14	8.255	8.134	8.890	7.514
15	8.967	9.397	9.600	9.426
16	9.899	9.444	10.359	7.920
17	3.848	3.608	3.590	3.590
18	9.288	9.068	9.093	9.244
19	9.922	10.111	10.928	10.782
20	12.844	12.616	12.740	12.740
21	11.450	11.324	11.306	11.154
22	9.783	10.186	11.046	10.810
23	6.360	6.859	6.970	6.482
24	12.906	12.729	12.537	12.576
25	8.787	9.481	10.584	10.609
	SSPE	0.603	3.05	2.573

Figure 8.7 presents the principal surface biplot when the surface is centered on the blanket surface data. The SSPE values are summarised in Table 8.5, which are lower compared to the errors in Table 8.1 in the previous section. Although the interest in this chapter is on the overall SSPE, the behaviour of individual samples gets attention in Chapter 9.

TABLE 8.5: SSPE from PCA and principal surface (PS) biplots (when surface is centered), for variables X, Y and Z from the blanket surface data.

	PCA	PS
X	1.688	6.625
Y	21.729	32.327
Z	0.055	2.131
$\sum$	23.472	41.083

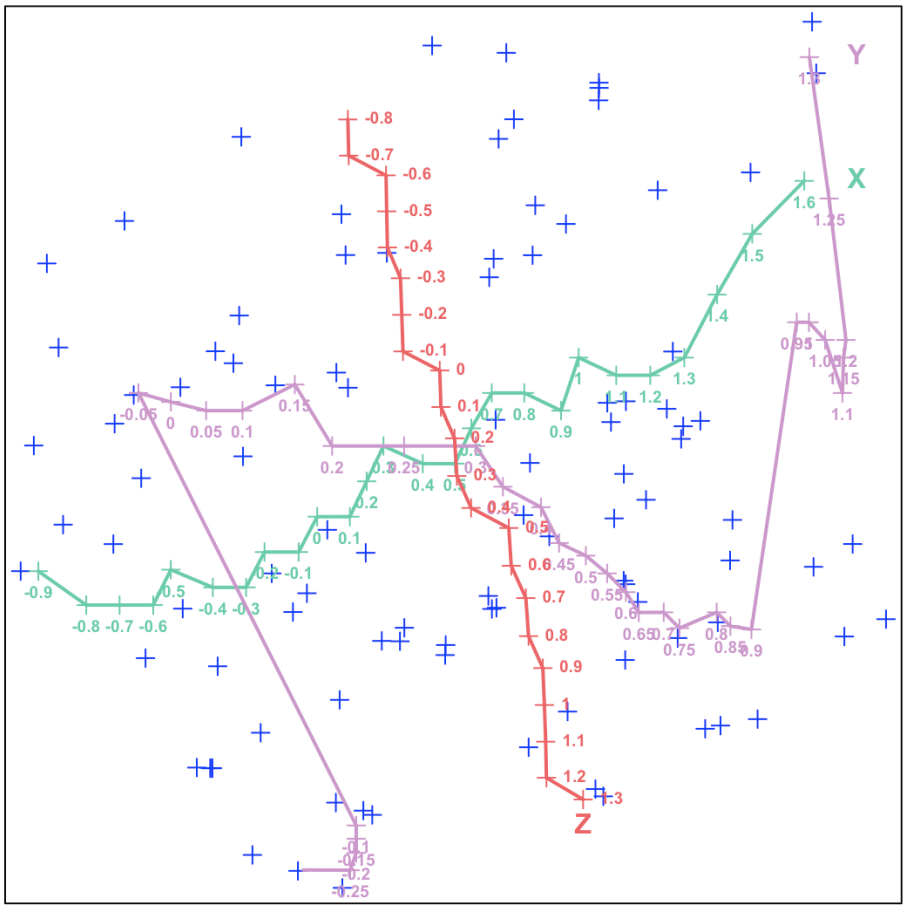

FIGURE 8.7: Principal surface biplot when surface is centered on blanket surface data.

The question now arises whether centering the principal surface will have an impact on prediction accuracy in the design of normal and circular projection discussed in Chapter 7. The principal surface biplot has been defined through the design of back-projection axes and it has been shown that when the surface is centered, sample prediction accuracy in the biplot improves. However, it is important to note that centering the surface, will have no effect on the circular and normal projection designs. These designs only use information from the two-dimensional representation to construct the biplot axes, i.e. the contour lines in two dimensions, whereas the back-projection design uses the original axes in p dimensions to construct the biplot axes.

8.4 Summary

The approach to the principal surface biplot is followed similar to that of PCA and nonlinear biplots from previous chapters. The main difference lies in the detail of using the principal surface as a nonlinear biplot plane, instead of a linear one. Samples are represented by using the principal surface coordinates $\lambda = (\lambda_1, \lambda_2)$ and biplot axes are constructed through the design of back-projection.

This chapter defined a measure of prediction accuracy given by the SSPE. It then provided motivation to center the principal surface to improve the prediction of samples in a principal surface biplot. From here onwards, for any data set, the principal surface is centered before a principal surface biplot can be constructed.

It was shown that the predictions were overall better in the PCA biplot for the blanket surface data and data from Chapter 3. It can be seen that prediction biplot axes are often not very useful in the principal surface biplot, and this is due to the highly nonlinear contours. Although a single axis for each variable is convenient, alternative representations of variable information can be used. The next chapter will illustrate the usefulness of the principal surface biplot since it allows for nonlinearity compared to linear restrictions of traditional biplot surfaces. The next chapter will also provide applications of the principal surface biplot, and the advantageous contributions it can add.

Chapter 9

Applications

9.1 Introduction

This chapter contains contributions the principal surface biplot can make on different types of data. The principal surface biplot has been defined to display prediction biplot axes constructed through back-projection. The axes are not always accurate for prediction in the principal surface biplot due to the nonlinear contours. To deal with this, the prediction of points is rather found using the contour lines, as done on the Breast Tissue data in Chapter 6. This gives better and more accurate predictions. Gower, Lubbe, and Le Roux (2011) outline the advantages of working with prediction regions for categorical data which can be brought into this context of predicting samples through contours.

The principal surface biplot utilised with sets of contours is still referred to as a biplot even though it is shown for each variable separately; as generalised biplots in Gower (1992), Gower and Hand (1996) and Gower, Lubbe, and Le Roux (2011) are called biplots even though only one variable can be represented at a time by prediction regions.

The principal surface biplot extends the toolbox of exploratory visualisation techniques for users in applied fields. It provides the user with another option, specifically when the data is highly nonlinear. Here, a few examples where the principal surface biplot provides added value is given. This chapter outlines four different contributions that the principal surface biplot can make to multivariate visualisation. First, the nonlinearity of the contours lines and biplot axes show the extent of linearity that exists in data. Second, the prediction of edge points in data that could potentially be outliers in the data are predicted better in a principal surface biplot. Third, separation of classes in nonlinear data can be better seen in a principal surface biplot. Lastly, to present nonlinear prediction in a large data set with an added feature of presenting the extent of nonlinearity in the data, which other biplots does not show.

For each contribution, a different data set is used to illustrate the value proposition of the principal surface biplot. It is beyond the scope of this thesis to perform a complete investigation into different types and shapes of data sets to establish the detailed properties of principal surface biplots. In general, the advantage of the principal surface biplot is to follow the data, nonlinearly if need be, compared to the restriction of linear biplot spaces in other types of biplots. This allows certain aspects of certain data sets to be more accurately displayed by principal surface biplots. The choice of data sets used here, are for illustration purposes. Although not reported here, the principal surface biplot has been tested on various data sets, real and simulated.

9.2 Nonlinearity of axes

A principal surface biplot is constructed by taking into account the nonlinearity that exists in the data. This is shown through the sets of contour lines and biplot axes. To demonstrate that this is what a principal surface biplot does, consider constructing one on data that lies on the flat plane, shown in Figure 9.1 that is defined by three variables X, Y and Z. Since the fitted principal surface will be linear, the sets of contour lines for each variable will be linear as shown in Figure 9.2. The linear contours result in linear biplot axes shown in Figure 9.3. The PCA biplot of the data is shown in Figure 9.4.

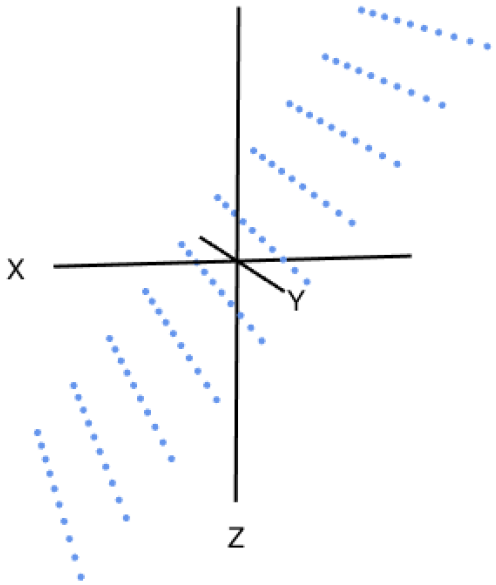

FIGURE 9.1: Data that lies on a flat plane.

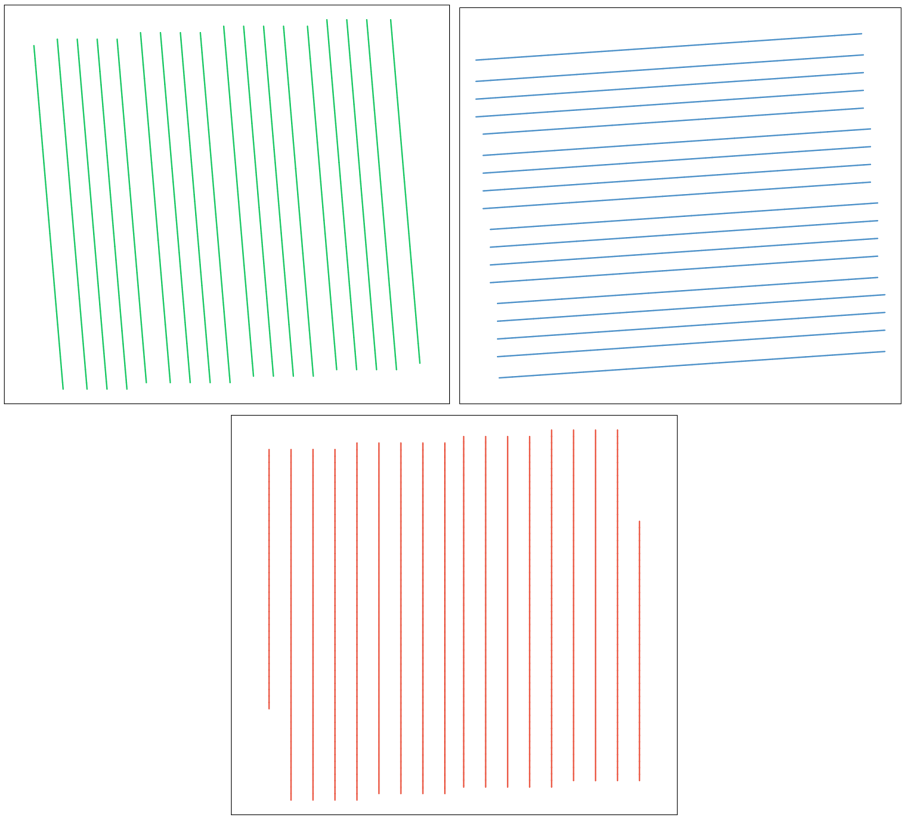

FIGURE 9.2: Sets of contour lines for variables X, Y and Z, respectively; from data that lies in a flat plane.

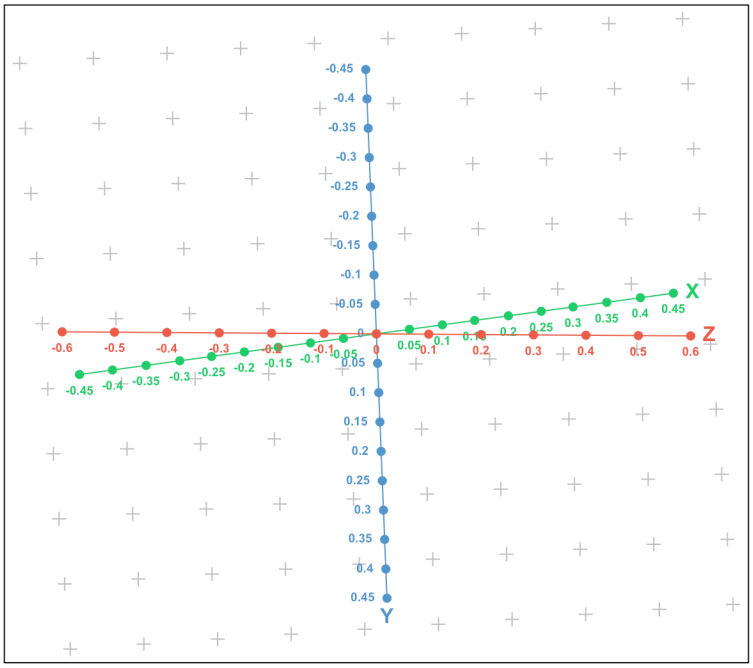

FIGURE 9.3: Principal surface biplot of data in Figure 9.1.

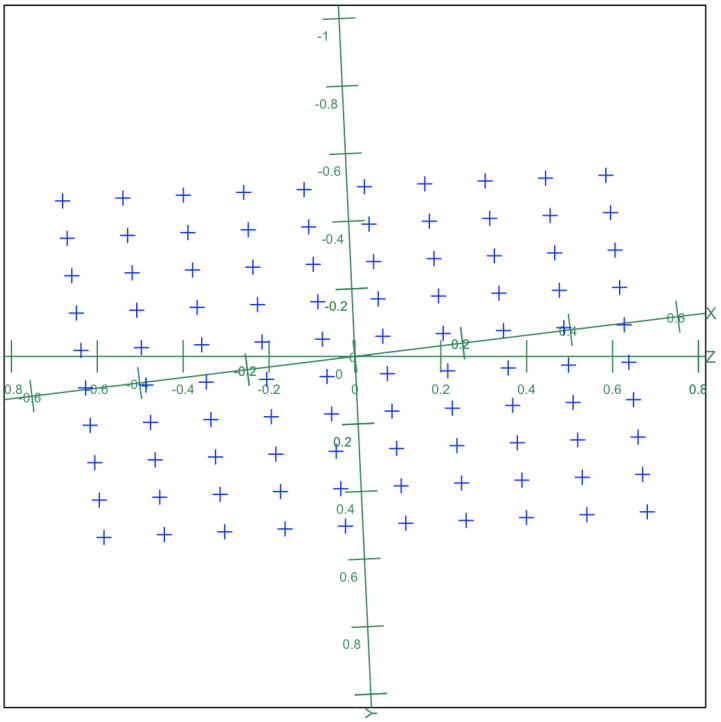

FIGURE 9.4: PCA biplot of data in Figure 9.1.

By seeing the similarities between the two different biplots, it can be reasoned that the principal surface biplot reduces to a PCA biplot when the data is linear. Therefore, for any data that produces a linear principal surface, the principal surface biplot results in a PCA biplot. This also shows that the principal surface

biplot demonstrates the nonlinearity, if any, through the contours. For example, the principal surface biplot of the blanket surface data in Figure 8.7 shows that the axes X and Z (Figures 6.18 and 6.20) are more linear than axis Y (Figure 6.19), and that the nonlinearity of the data comes from variable Y. It should be noted that the nonlinearity in a principal surface biplot will be different to the nonlinear axes in nonlinear biplots from Chapter 4.

9.3 Prediction of edge points

For this section, data from the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets.html) is used. The Madelon data, which was part of the NIPS 2003 feature selection challenge is a simulated data set consisting of 2000 observations and 20 variables labelled X_1 to X_{20}. It was purposefully simulated to be a highly nonlinear set of data, and therefore, is well suited for the application of the principal surface biplot. A random sample of 100 observations from this data is used to fit a principal surface biplot. Of these 100 observations, sample points that lie on the edge in the principal surface biplot are chosen. Other sample points in the data, can also be chosen, however, the edge points are chosen as they are not always accurately predicted by traditional biplot axes. For example, the PCA biplot goes through the centroid of the data where the plane is close to the central data points and orthogonal projection onto the linear plane is quite accurate. However, at the edges, the linear PCA plane remains in the same direction throughout, while the principal surface can change with the shape of the data. Edge points are also crucial as they are potential outliers that can be identified in the data.

The first panel in Figure 9.5 presents the sample representation in the principal surface biplot with eight edge points (**100**, **7**, **17**, **40**, **38**, **95**, **31** and **98**). These points are also highlighted in the sample representation of the PCA biplot in the second panel of Figure 9.5.

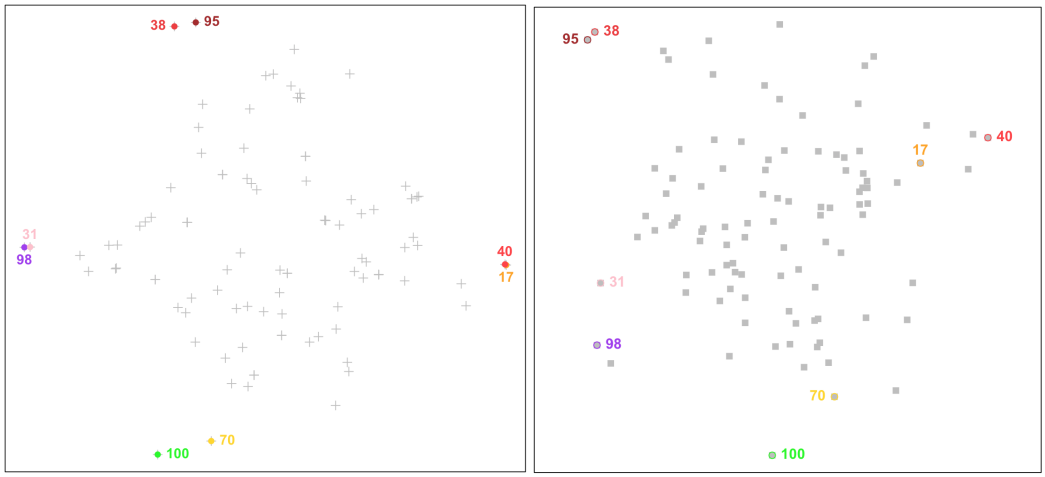

FIGURE 9.5: Eight edge points are identified from the principal surface biplot shown in the principal surface biplot (first panel) and PCA biplot (second panel) respectively.

The purpose of this section is to show that the prediction through the contours in the principal surface biplot cater better for the edge points than in a PCA biplot shown in Figure 9.6. The sets of contour lines for each variable X_1 to X_{20} are shown in Figure 9.7. One can see that the sets of contour lines for these variables are highly nonlinear, with some contour lines forming closed loops for many variables. For example, variable X_1 to X_6, X_8 to X_{10}, X_{12} to X_{18} and X_{20}. Each contour represents a marker point for a variable. Samples are predicted by interpolating between contour lines in the region where the sample point lies.

Table 9.1 presents the original value of the edge points for all variables, whereas Tables 9.2 and 9.3 present the predicted value from the PCA and principal surface biplot, respectively. The green predicted points in Table 9.3 indicate that the principal surface biplot predicted the edge points better than the PCA biplot. What is more of interest is the magnitude of differences between the original data and each respective biplot. This is presented in Table 9.4. The difference is calculated as the absolute difference between the original data value and the predicted value from the biplot.

From these tables mentioned, it can be seen that the principal surface biplot predicted the edge points overall much better than the PCA biplot. The magnitude of differences in Table 9.4 also indicate that where for some samples the principal surface did not perform better, the differences are not far off from the PCA biplot. For example, for sample **70** predicted for variable X_2, the difference under the PCA biplot is 0.33 and 0.35 for the principal surface biplot. On the other hand, when the principal surface biplot performed better, the differences were larger compared to the PCA biplot. For example, again sample **70** predicted for variable X_7, the differences are 0.92 and 0.14 for the PCA and principal surface biplots, respectively.

It is expected that since the principal surface biplot predicted the edge points more accurately, the prediction will be better for all sample points. This is shown in Table 9.5, by the SSPE values for each variable in each biplot, when all 100 samples are predicted.

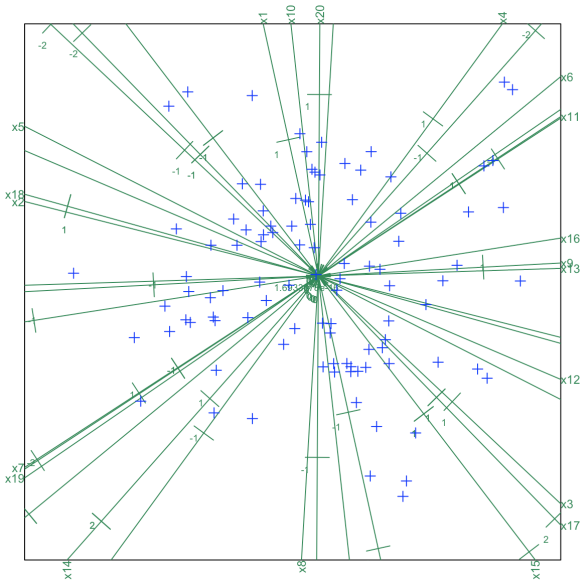

FIGURE 9.6: PCA Biplot of Madelon data

TABLE 9.1: Original values of edge points for all variables in the data.

Data	**70**	**100**	**98**	**31**	**95**	**38**	**17**	**40**
X_1	0.373	1.334	1.174	1.174	0.853	1.654	-1.548	-1.548
X_2	0.122	0.024	1.390	1.651	-0.463	-1.472	-0.659	-1.537
X_3	-1.430	0.177	-0.967	-2.744	-0.553	-0.286	0.225	1.150
X_4	-0.604	-0.604	0.042	1.548	0.580	0.472	0.257	-0.173
X_5	-0.625	-0.021	0.155	1.112	-1.103	-0.373	1.414	0.734
X_6	-2.171	-0.909	2.121	0.101	1.363	1.237	0.227	-1.161
X_7	1.800	2.578	1.297	1.022	-3.483	-0.144	0.268	0.107
X_8	-1.193	0.950	-1.907	0.236	-1.907	-1.907	0.236	1.664
X_9	-1.695	-1.302	0.339	0.010	0.404	0.535	0.995	-0.318
X_{10}	0.730	-0.139	-0.719	0.585	1.020	-1.009	0.585	-0.139
X_{11}	-1.396	-0.717	0.796	-1.004	-0.300	1.266	1.109	0.274
X_{12}	1.444	1.057	-1.404	0.213	-0.420	-0.139	-1.369	-0.631
X_{13}	-0.780	-0.354	0.041	1.532	-0.202	1.501	0.680	0.041
X_{14}	0.604	0.420	-0.710	-2.365	-1.393	-1.235	1.051	1.681
X_{15}	0.961	-1.579	-1.071	-1.071	1.808	-0.478	1.723	1.554
X_{16}	-0.599	-1.236	-1.432	-1.089	1.996	2.388	-1.530	1.017
X_{17}	-1.069	-1.313	-1.313	-0.337	-0.011	-0.418	1.372	0.965
X_{18}	-0.493	1.040	1.998	-1.611	0.529	-2.441	1.391	-1.419
X_{19}	-0.966	1.512	1.042	0.316	-1.393	-0.752	-0.752	-0.624
X_{20}	-0.845	-0.161	0.637	0.067	2.234	0.979	-1.643	-0.845

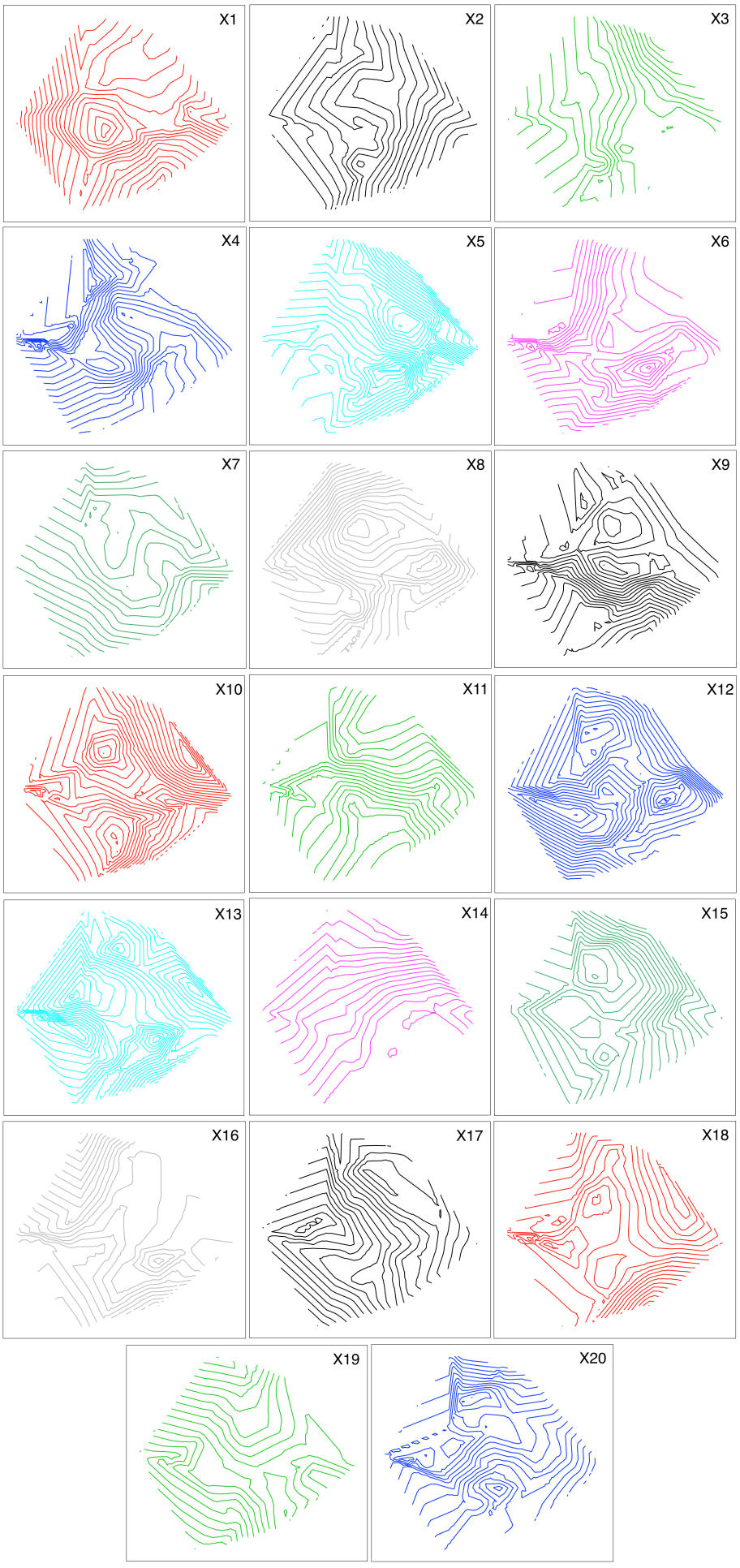

FIGURE 9.7: Sets of contour lines for variables X_1 to X_{20}, respectively.

TABLE 9.2: Predicted values of edge points from the PCA biplot.

PCA	**70**	**100**	**98**	**31**	**95**	**38**	**17**	**40**
X_1	-0.149	0.395	1.489	1.364	1.071	1.006	-1.144	-1.673
X_2	0.453	0.661	0.477	0.301	-0.368	-0.395	-0.259	-0.378
X_3	-0.500	-0.981	-1.205	-0.935	0.015	0.075	0.792	1.157
X_4	-0.814	-0.731	0.369	0.549	1.369	1.362	-0.449	-0.659
X_5	0.316	0.510	0.465	0.328	-0.181	-0.205	-0.280	-0.408
X_6	-0.484	-0.500	0.043	0.169	0.712	0.716	-0.135	-0.199
X_7	0.877	0.950	0.042	-0.201	-1.221	-1.234	0.156	0.231
X_8	0.165	0.027	-0.401	-0.401	-0.466	-0.450	0.334	0.489
X_9	-1.134	-1.481	-0.731	-0.342	1.188	1.235	0.303	0.440
X_{10}	-0.062	0.044	0.294	0.278	0.257	0.245	-0.233	-0.341
X_{11}	-1.118	-1.209	-0.048	0.261	1.559	1.575	-0.203	-0.301
X_{12}	-0.236	-0.368	-0.313	-0.214	0.154	0.171	0.183	0.267
X_{13}	-0.460	-0.606	-0.309	-0.150	0.475	0.495	0.132	0.192
X_{14}	1.027	0.978	-0.313	-0.557	-1.639	-1.637	0.453	0.666
X_{15}	-0.361	-0.869	-1.299	-1.057	-0.237	-0.174	0.893	1.305
X_{16}	-0.665	-0.839	-0.349	-0.130	0.742	0.766	0.119	0.172
X_{17}	-0.497	-1.019	-1.316	-1.036	-0.055	0.011	0.877	1.281
X_{18}	0.618	0.909	0.673	0.431	-0.489	-0.526	-0.370	-0.540
X_{19}	0.448	0.480	0.008	-0.114	-0.631	-0.637	0.090	0.132
X_{20}	-0.346	0.007	1.010	0.992	1.075	1.033	-0.828	-1.212

TABLE 9.3: Predicted values of edge points from the principal surface (PS) biplot. Green points indicate better predicted value in the principal surface biplot compared to the PCA biplot.

PS	**70**	**100**	**98**	**31**	**95**	**38**	**17**	**40**
X_1	0.519	1.170	1.000	1.000	0.853	1.170	-1.440	-1.440
X_2	0.470	0.180	1.696	1.651	-0.260	-0.260	-0.990	-0.990
X_3	-0.290	-0.480	-1.590	-1.590	-0.391	-0.480	0.613	0.615
X_4	-0.620	-0.618	0.909	0.913	0.390	0.700	-0.010	-0.010
X_5	-0.176	-0.130	0.420	0.420	-0.579	-0.690	0.545	0.557
X_6	-1.249	-1.480	0.640	0.640	0.931	1.070	-0.366	-0.377
X_7	1.663	2.260	0.735	0.709	1.590	-1.589	0.360	0.360
X_8	-0.036	-0.020	-0.927	-0.900	-1.339	-1.516	0.850	0.850
X_9	-1.038	-0.992	-0.072	-0.067	0.350	0.356	0.478	0.480
X_{10}	-0.250	0.070	-0.030	-0.030	0.290	0.290	0.070	0.070
X_{11}	-0.980	-0.700	0.390	0.390	0.522	0.520	0.522	0.524
X_{12}	0.782	0.880	-0.710	-0.710	-0.381	-0.297	-0.696	-0.701
X_{13}	-0.440	-0.270	0.241	0.240	0.205	0.330	0.427	0.435
X_{14}	0.471	0.470	-0.950	-0.950	-1.578	-1.580	0.875	0.870
X_{15}	-0.530	-0.980	-1.130	-1.130	0.341	0.073	1.560	1.560
X_{16}	-1.143	-1.080	-0.620	-0.620	1.410	1.673	0.160	0.160
X_{17}	-1.130	-1.540	-0.800	-0.800	0.085	0.090	1.280	1.280
X_{18}	0.286	0.489	0.196	0.191	-0.554	-0.815	-0.080	-0.080
X_{19}	0.972	1.090	0.494	0.495	-1.200	-1.145	-0.120	-0.120
X_{20}	-0.651	-0.620	0.540	0.513	1.410	1.124	-1.050	-1.050

TABLE 9.4: Magnitude of differences from the PCA and principal surface biplot for the edge point samples. Differences are calculated as the absolute difference between the original data and predicted values in the respective biplot.

	70		100		98		31		95		38		17		40		$\sum$	
	PCA	PS	PCA	PS	PCA	PS	PCA	PS	PCA	PS	PCA	PS	PCA	PS	PCA	PS	PCA	PS
X_1	0.52	0.15	0.94	0.16	0.32	0.17	0.19	0.17	0.22	0.00	0.65	0.48	0.40	0.11	0.12	0.11	3.36	1.36
X_2	0.33	0.35	0.64	0.16	0.91	0.31	1.35	0.00	0.10	0.20	1.08	1.21	0.40	0.33	1.16	0.55	5.96	3.10
X_3	0.93	1.14	1.16	0.66	0.24	0.62	1.81	1.15	0.57	0.16	0.36	0.19	0.57	0.39	0.01	0.54	5.64	4.85
X_4	0.21	0.02	0.13	0.01	0.33	0.87	1.00	0.64	0.79	0.19	0.89	0.23	0.71	0.27	0.49	0.16	4.54	2.38
X_5	0.94	0.45	0.53	0.11	0.31	0.26	0.78	0.69	0.92	0.52	0.17	0.32	1.69	0.87	1.14	0.18	6.49	3.40
X_6	1.69	0.92	0.41	0.57	2.08	1.48	0.07	0.54	0.65	0.43	0.52	0.17	0.36	0.59	0.96	0.78	6.74	5.49
X_7	0.92	0.14	1.63	0.32	1.25	0.56	1.22	0.31	2.26	1.89	1.09	1.45	0.11	0.09	0.12	0.25	8.62	5.01
X_8	1.36	1.16	0.92	0.97	1.51	0.98	0.64	1.14	1.44	0.57	1.46	0.39	0.10	0.61	1.17	0.81	8.59	6.63
X_9	0.56	0.66	0.18	0.31	1.07	0.41	0.35	0.08	0.78	0.05	0.70	0.18	0.69	0.52	0.76	0.80	5.09	3.00
X_{10}	0.79	0.98	0.18	0.21	1.01	0.69	0.31	0.62	0.76	0.73	1.25	1.30	0.82	0.52	0.20	0.21	5.33	5.25
X_{11}	0.28	0.42	0.49	0.02	0.84	0.41	1.26	1.39	1.86	0.82	0.31	0.75	1.31	0.59	0.57	0.25	6.93	4.64
X_{12}	1.68	0.66	1.42	0.18	1.09	0.69	0.43	0.92	0.57	0.04	0.31	0.16	1.55	0.67	0.90	0.07	7.96	3.40
X_{13}	0.32	0.34	0.25	0.08	0.35	0.20	1.68	1.29	0.68	0.41	1.01	1.17	0.55	0.25	0.15	0.39	4.98	4.14
X_{14}	0.42	0.13	0.56	0.05	0.40	0.24	1.81	1.41	0.25	0.19	0.40	0.35	0.60	0.18	1.02	0.81	5.45	3.36
X_{15}	1.32	1.49	0.71	0.60	0.23	0.06	0.01	0.06	2.05	1.47	0.30	0.55	0.83	0.16	0.25	0.01	5.70	4.40
X_{16}	0.07	0.54	0.40	0.16	1.08	0.81	0.96	0.47	1.25	0.59	1.62	0.71	1.65	1.69	0.84	0.86	7.87	5.83
X_{17}	0.57	0.06	0.29	0.23	0.00	0.51	0.70	0.46	0.04	0.10	0.43	0.51	0.50	0.09	0.32	0.31	2.85	2.27
X_{18}	1.11	0.78	0.13	0.55	1.33	1.80	2.04	1.80	1.02	1.08	1.91	1.63	1.76	1.47	0.88	1.34	10.18	10.45
X_{19}	1.41	1.94	1.03	0.42	1.03	0.55	0.43	0.18	0.76	0.19	0.11	0.39	0.84	0.63	0.76	0.50	6.38	4.81
X_{20}	0.50	0.19	0.17	0.46	0.37	0.10	0.92	0.45	1.16	0.82	0.05	0.14	0.82	0.59	0.37	0.21	4.36	2.96

TABLE 9.5: SSPE values for the PCA biplot and principal surface (PS) biplot for variables X_1 to X_{20} in the sample of Madelon data. Values coloured in green indicate better prediction for that variable in the respective biplot.

Variable	PCA	PS	Variable	PCA	PS
X_1	56.500	56.804	X_{11}	68.288	50.174
X_2	92.210	63.811	X_{12}	96.684	67.058
X_3	73.615	59.151	X_{13}	93.579	76.148
X_4	76.642	63.629	X_{14}	67.278	59.283
X_5	94.288	60.566	X_{15}	70.905	53.256
X_6	92.878	70.892	X_{16}	88.348	75.265
X_7	80.127	56.766	X_{17}	69.250	59.854
X_8	94.727	64.605	X_{18}	86.016	76.055
X_9	66.523	66.834	X_{19}	94.017	79.741
X_{10}	97.162	62.174	X_{20}	73.951	53.726

9.4 Separation of classes

The Wine data, available in the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets.html), consists of results of a chemical analysis of wines grown in the same region in Italy. The values in the data consisted of 13 variables found in each of three types of wines with a total of 178 instances. Attention is given to the representation of sample points in a PCA and principal surface biplot. Figure 9.8 displays the two representations of sample points with only three variables (malic acid, flavanoids and proline) in the PCA and principal surface biplot, respectively.

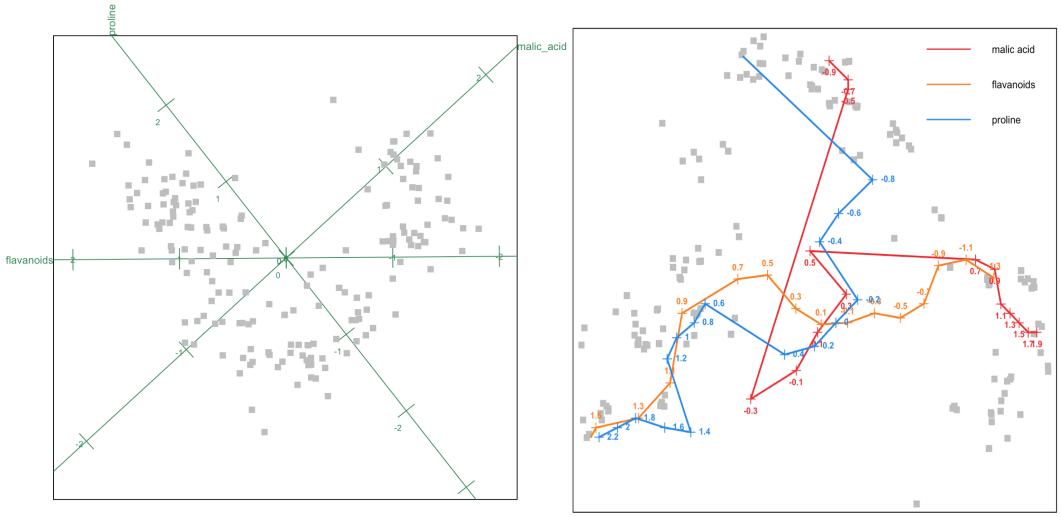

FIGURE 9.8: Representation of samples in a PCA (first panel) and principal surface biplot (second panel), respectively.

The purpose is to illustrate that in the principal surface representation there is a better view of class separation of the three types of wines than in the PCA representation. The approach is taken from an unsupervised point of view. In a supervised classification setting, a Canonical Variance Analysis (Gower, Lubbe, and Le Roux, 2011) biplot is better suited than a PCA biplot. However, here in the unsupervised setting, what is shown is that the principal surface biplot better represents the classes in the data, than a PCA biplot.

Figure 9.9 presents the same representation of samples as Figure 9.8, only with samples being coloured according to their wine type. If the samples were not coloured, it will be more difficult to pick up the class separation in the PCA biplot. Samples for each class in the principal surface biplot are closer together and there is a bigger class separation. Whereas in the PCA biplot, samples are not as close together and there is a smaller class separation. The principal surface biplot axes can be used to make better conclusions about the classes, especially if the class variable was not available. Class separation is shown here for the Wine data, but can be established to be the case for most data sets as well. For example, the famous Iris data from Fisher (1936) available in the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets.html), that contains three types of the iris plants of 50 instances each; and the Breast Tissue data from Chapter 6 that includes a class variable. These data sets also showed a better class separation in the principal surface biplot than in the PCA biplot.

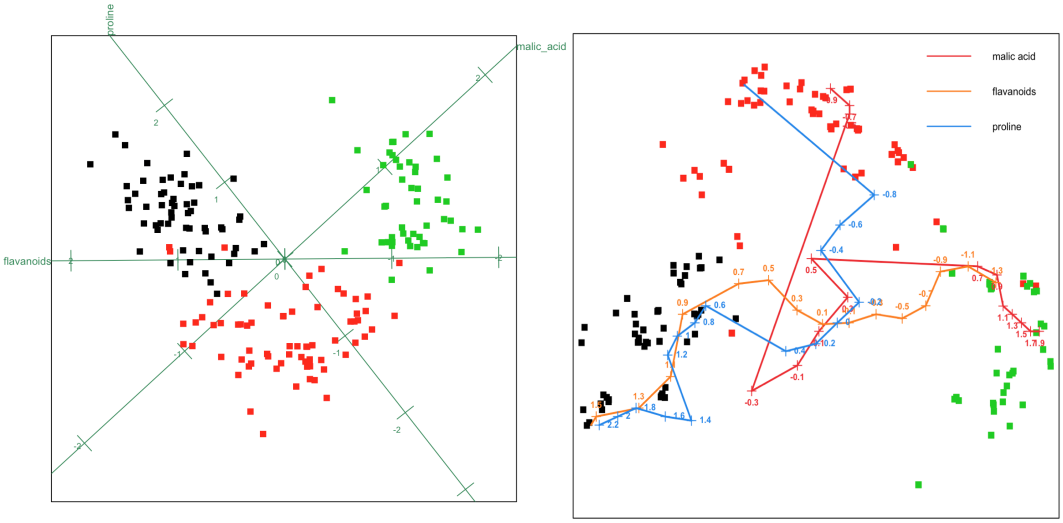

FIGURE 9.9: Representation of samples coloured according to the type of wine with three variables (malic acid, flavanoids, and proline), in a PCA (first panel) and principal surface biplot (second panel), respectively.

In addition, interpretations can be made on the three wine types in terms of its values of the variables in the principal surface biplot of Figure 9.9. The wine type coloured in green have high values in the variable malic acid, and low values in the variable flavanoids. The wine type coloured in red have low values for the variables malic acid and proline. The wine type coloured in black has high values in the variables flavanoids and proline. These conclusions are more easily distinguishable in the principal surface biplot than in the PCA biplot. This is because

of the unsupervised setting where the principal surface follows the data more closely, it has the flexibility to "change direction" for each group and therefore, shows groupings better than PCA.

Figures 9.10, 9.11 and 9.12 displays the sets of contour lines for the variables malic acid, flavanoids and proline, respectively. The contour lines can also be used to make better conclusions about the classes, as shown with the principal surface biplot axes. The SSPE values when samples in the Wine data are predicted from the PCA biplot and the contour lines in the principal surface biplot are shown in Table 9.6. The principal surface biplot has overall lower SSPE values for variables, and therefore, produce more accurate predictions than the PCA biplot. There are especially better predictions in the principal surface biplot for variables that have nonlinear sets of contour lines, such as the malic acid variable.

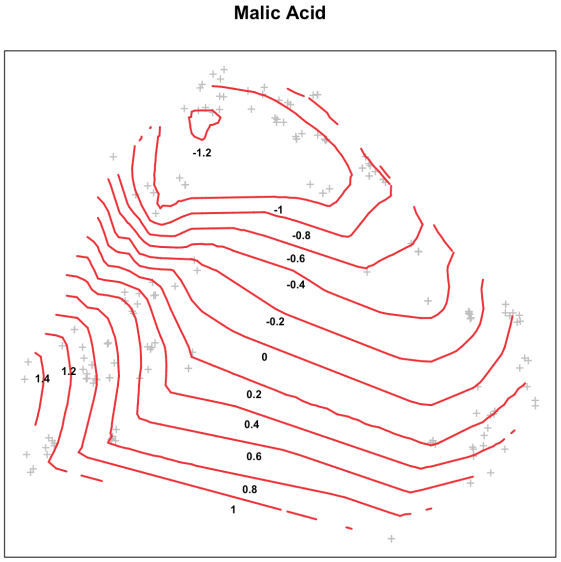

FIGURE 9.10: Sets of contour lines for variable malic acid in the Wine data.

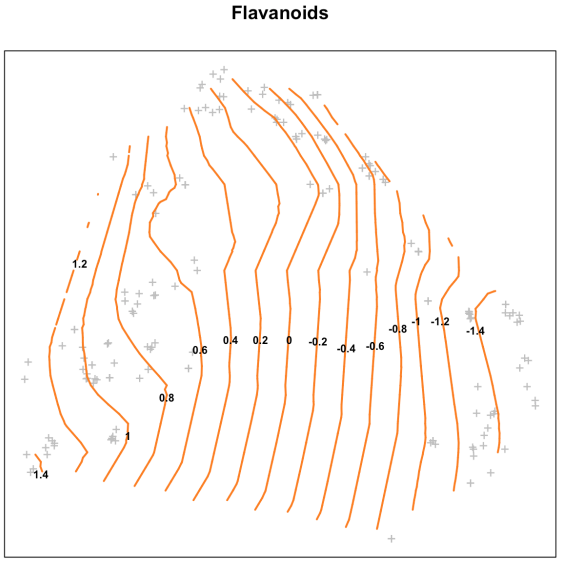

FIGURE 9.11: Sets of contour lines for variable flavanoids in the Wine data.

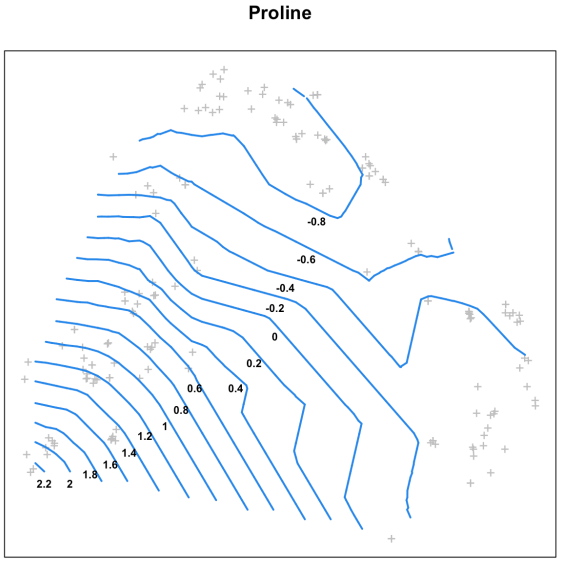

FIGURE 9.12: Sets of contour lines for variable proline in the Wine data.

9.5 Nonlinear prediction

In this section, the full Madelon data set of 2000 observations is used. For each variable, the sample points can be plotted according to a colour gradient. The colour gradient ranges from the colour yellow to blue, which represents the values of a particular variable in ascending order. The representations are shown in a PCA, spline-based nonlinear biplot using the SMACOF distance measure and a principal surface biplot. The principal surface biplot is shown with constructed biplot axes and in addition the set of contour lines for the respective variable coloured according to the colour gradient. The figures in Figure 9.13 demonstrate

TABLE 9.6: SSPE values for the PCA biplot and principal surface (PS) biplot for variables in the Wine data. Values coloured in green indicate better prediction for that variable in the respective biplot.

Variable	PCA	PS
alcohol	56.265	58.409
malic acid	104.566	78.688
ash	132.844	70.198
alcalinity	129.245	70.751
magnesium	120.527	41.538
total phenols	45.395	41.538
flavanoids	28.005	27.063
non-flavanoids	102.401	59.13
proanthocyanins	94.491	61.27
color intensity	46.313	26.996
hue	69.208	61.466
diluted	47.179	38.519
proline	49.661	34.281

the three constructed biplots on the Madelon data showing only the biplot axis for variable X_1 and the sets of contour lines from variable X_1. The samples and contour lines are coloured from yellow to blue according to the range of X_1.

From the PCA and the spline-based nonlinear biplot in Figure 9.13, it can be seen that there is no clear distinction between the yellow and blue points, or that the points have a trend of moving from yellow to blue along with the biplot axis. On the other hand, in the principal surface biplot the yellow and blue points are better grouped together. There also seems to be a trend of colour in the sample points that moves with the principal surface biplot axis. The trend can also be seen from the sets of contour lines. It is also surprising, that even though the biplot axis in the spline-based nonlinear biplot is nonlinear, it looks more similar to the PCA biplot axis, than it does to the principal surface biplot axis.

Figures 9.14 to 9.32 illustrate the same as Figure 9.13, for variables X_2 to X_{20}, respectively. The same conclusions can be drawn on the other variables in terms of trends of colour. There are for certain variables better trends of colour moving from yellow to blue in the PCA and spline-based nonlinear biplots. However, for these variables, these trends are more convincing in the principal surface biplot. The contour plots with the colour gradient also provide a better understanding on the patterns and trends of values for each variable in the data. This gives the view that the representation of samples presented in a principal surface biplot are meaningful.

For most variables, there are no visible marker points on the constructed PCA biplots. The marker points in the principal surface biplots, give clear values of the marker points on the biplot axes. Although reading off values from the principal surface biplot axes can be sometimes difficult, for example in Figure 9.20 for variable X_8 or in Figure 9.23 for variable X_{11}, it still shows that it can deal with larger data sets that are particularly nonlinear, better than PCA biplots do. In addition, there is no need to worry about marker points with contours as it is always easier to see the values.

It can be seen that there is no structure visible in the PCA or spline-based nonlinear biplot since it is mostly a ball with blue and yellow points mixed, while different patterns come to the fore in the samples and colour coding in the principal surface biplot. This is essentially a way of representing the variable, i.e. the "bi" in biplot of the principal surface biplot.

There are many plots presented in this section. Nonetheless, the purpose of this section is to illustrate that the principal surface biplot has the added feature of demonstrating the extent of nonlinearity in the data, which the PCA and spline-based nonlinear biplot does not show. It is also important to view this notion variable by variable. Since the Madelon data is simulated to be a highly nonlinear set of data, and also a very large one, it can be assumed that the same patterns will be seen on other types of data, even with an extreme case of nonlinearity. This is linked to the flexibility of the principal surface to follow nonlinearity in the data, thus the same conclusions will be made for other nonlinear data sets.

Samples in the full Madelon data set, predicted better with the sets of contours in the principal surface biplot than in the PCA and spline-based nonlinear biplot (SMACOF method). The SSPE values are given in Table 9.7.

TABLE 9.7: SSPE values in the PCA, spline-based nonlinear and principal surface (PS) biplot for variables X_1 to X_{20} in the full set of the Madelon data. Values coloured in green indicate better prediction for that variable in the respective biplot.

Variable	PCA	Spline	PS
X_1	1405	1557	1443
X_2	1975	2040	1627
X_3	1995	2030	1623
X_4	1575	1834	1551
X_5	1828	1919	1586
X_6	1545	1659	1635
X_7	1670	1904	1612
X_8	1568	1650	1563
X_9	1721	1778	1535
X_{10}	1925	1938	1615
X_{11}	1981	2002	1683
X_{12}	1463	1668	1543
X_{13}	1917	1977	1534
X_{14}	1727	1834	1660
X_{15}	1913	2022	1609
X_{16}	1750	1849	1519
X_{17}	1549	1732	1610
X_{18}	1937	2028	1302
X_{19}	1946	2036	1561
X_{20}	1971	1996	1650

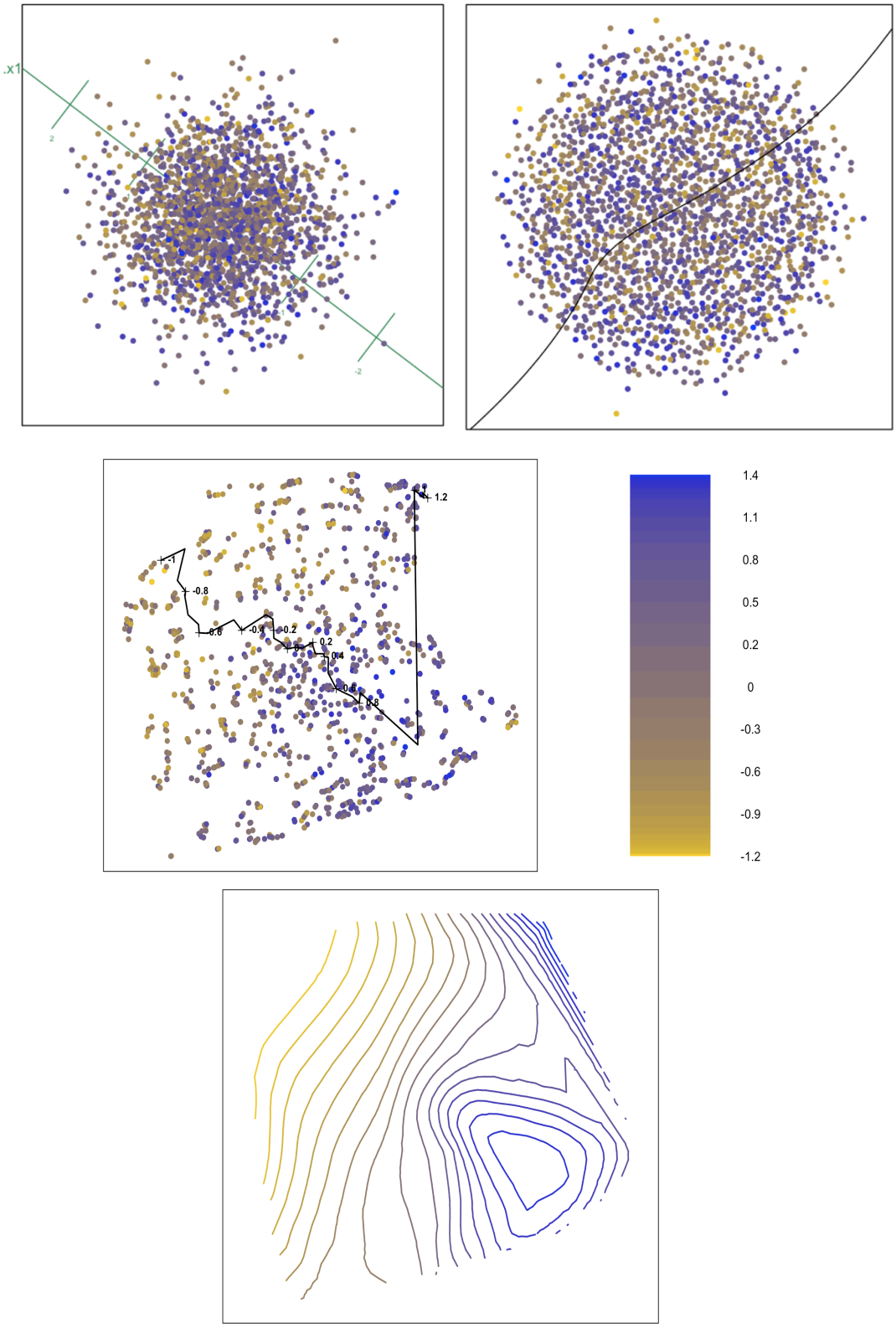

FIGURE 9.13: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_1 and sample points coloured according to the values of X_1.

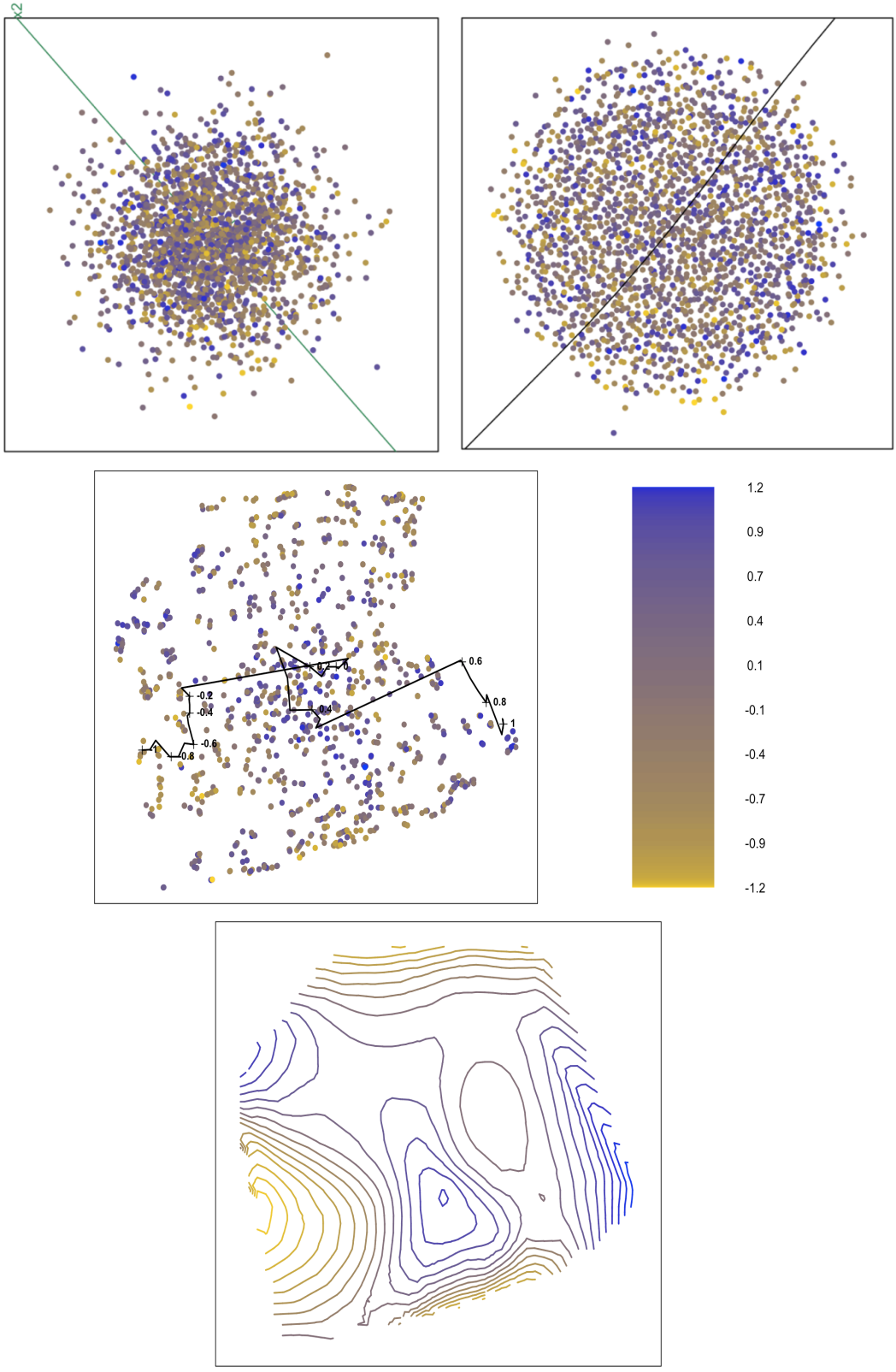

FIGURE 9.14: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_2 and sample points coloured according to the values of X_2.

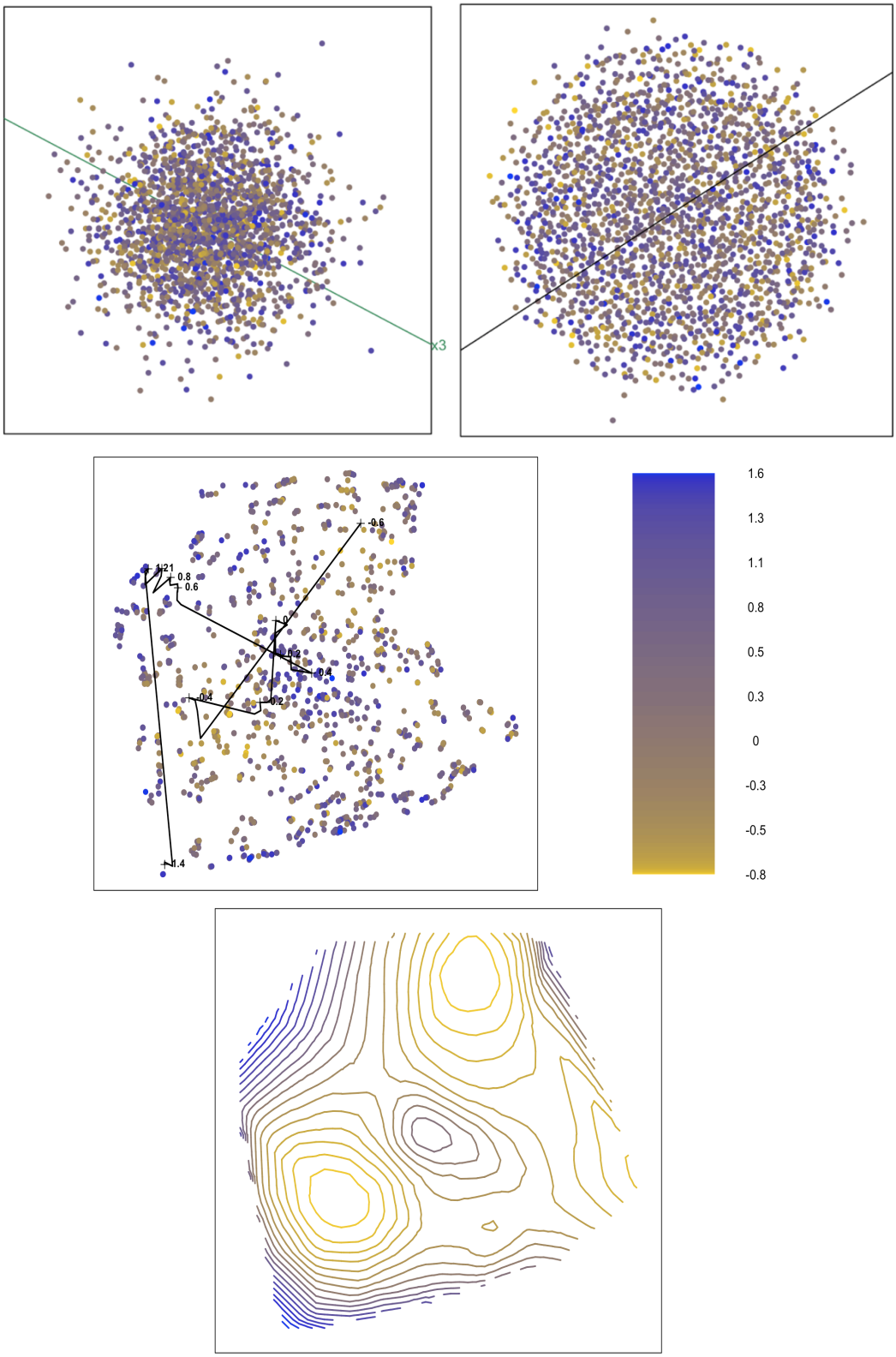

FIGURE 9.15: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_3 and sample points coloured according to the values of X_3.

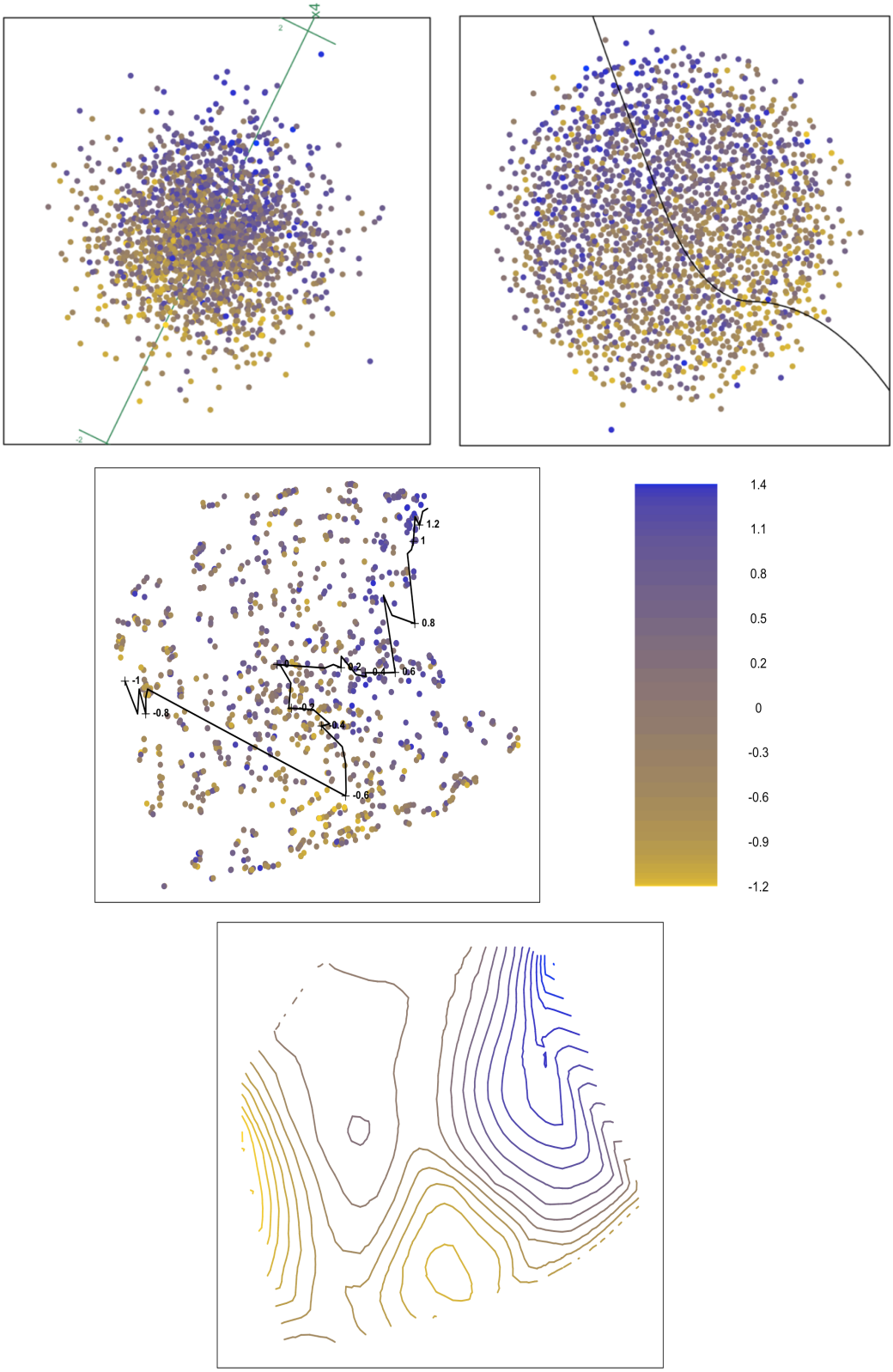

FIGURE 9.16: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_4 and sample points coloured according to the values of X_4.

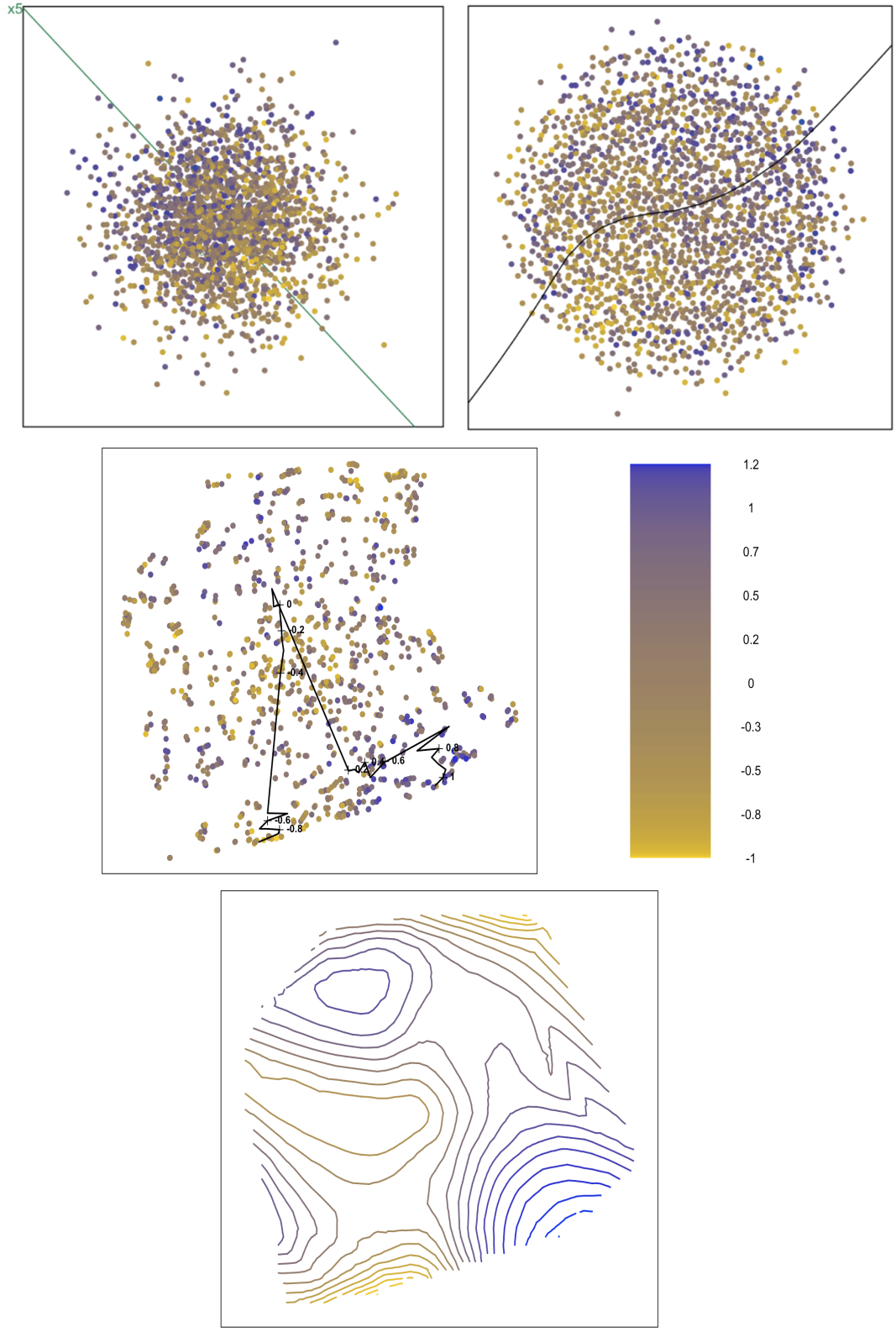

FIGURE 9.17: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_5 and sample points coloured according to the values of X_5.

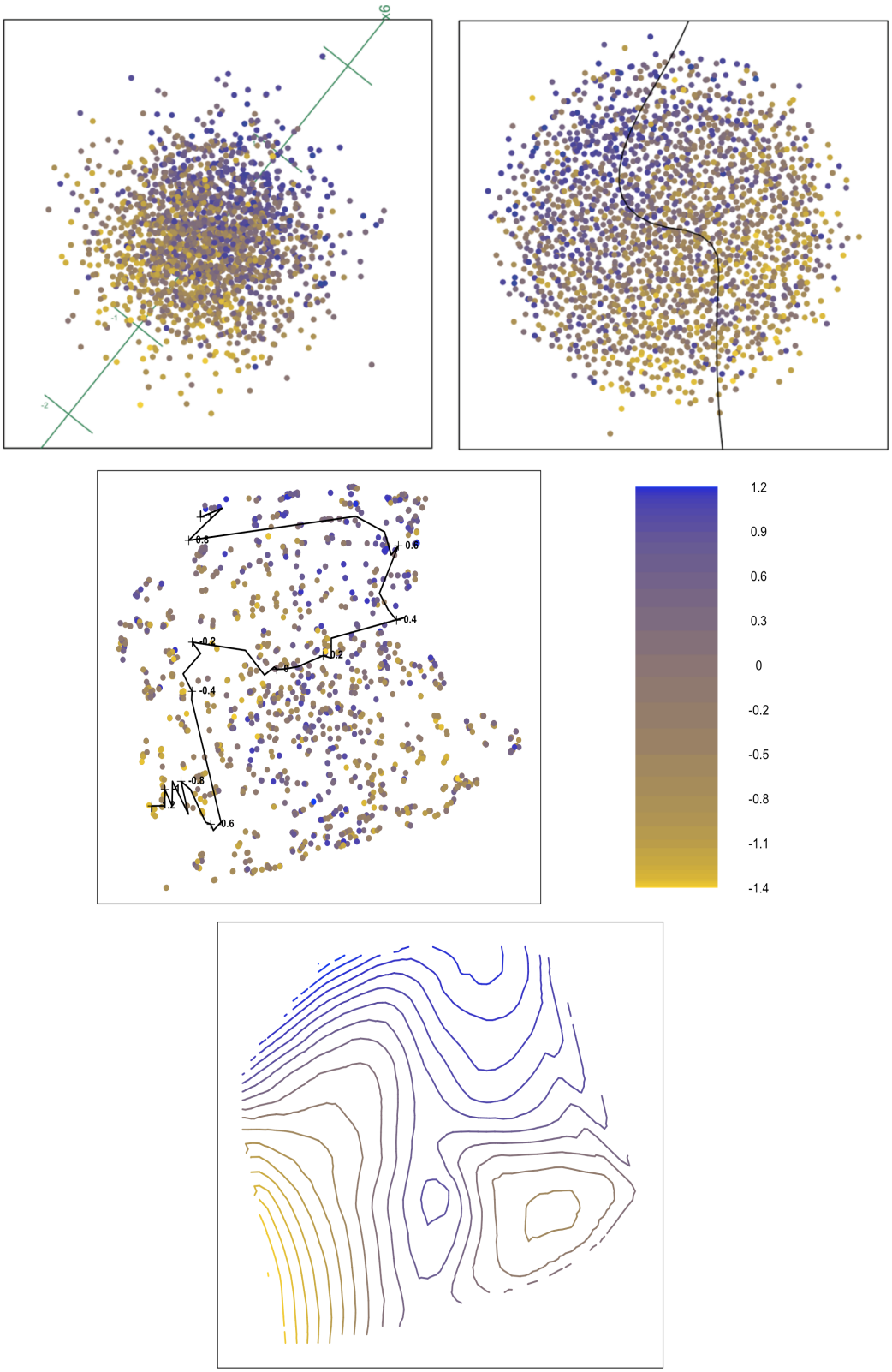

FIGURE 9.18: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_6 and sample points coloured according to the values of X_6.

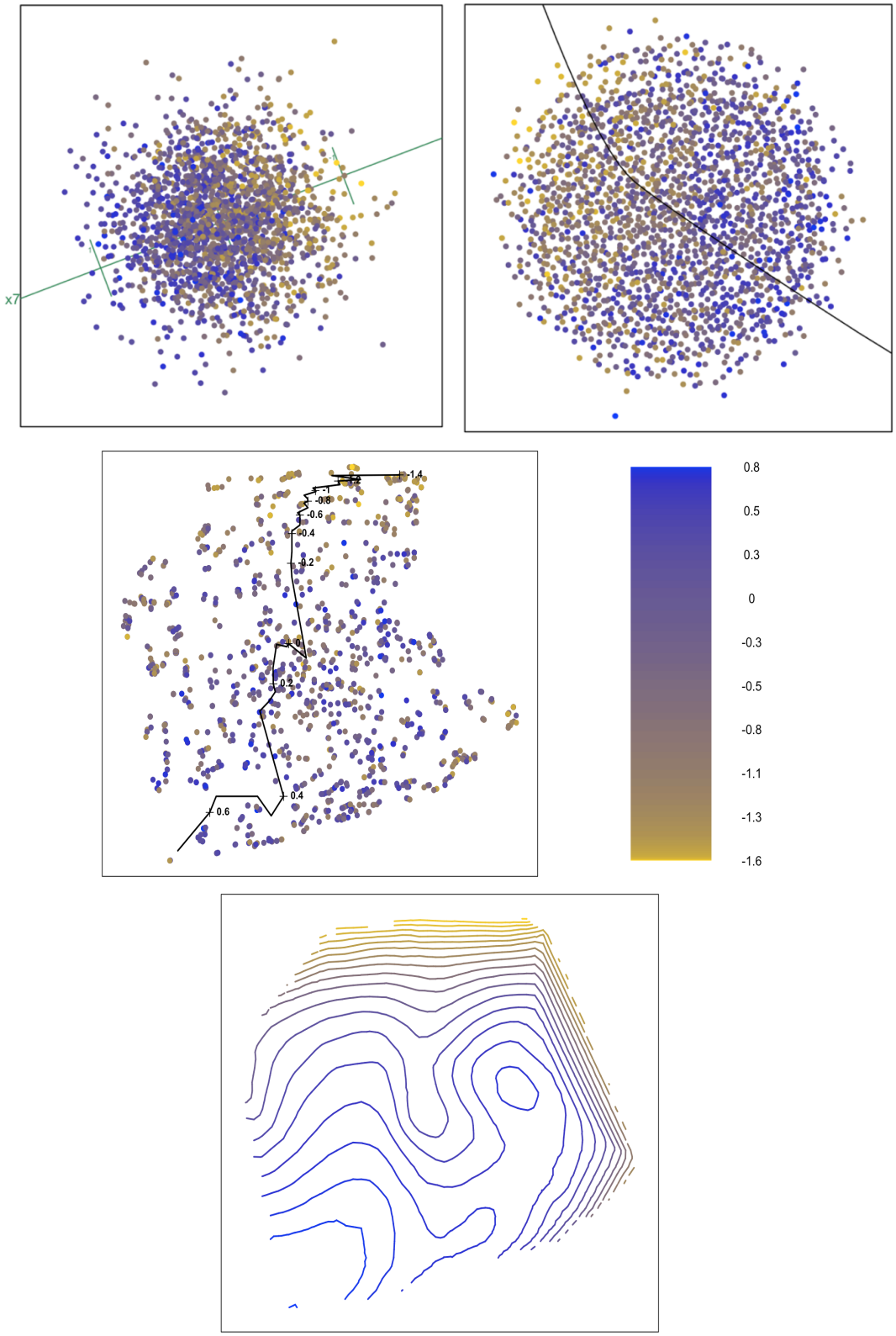

FIGURE 9.19: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_7 and sample points coloured according to the values of X_7.

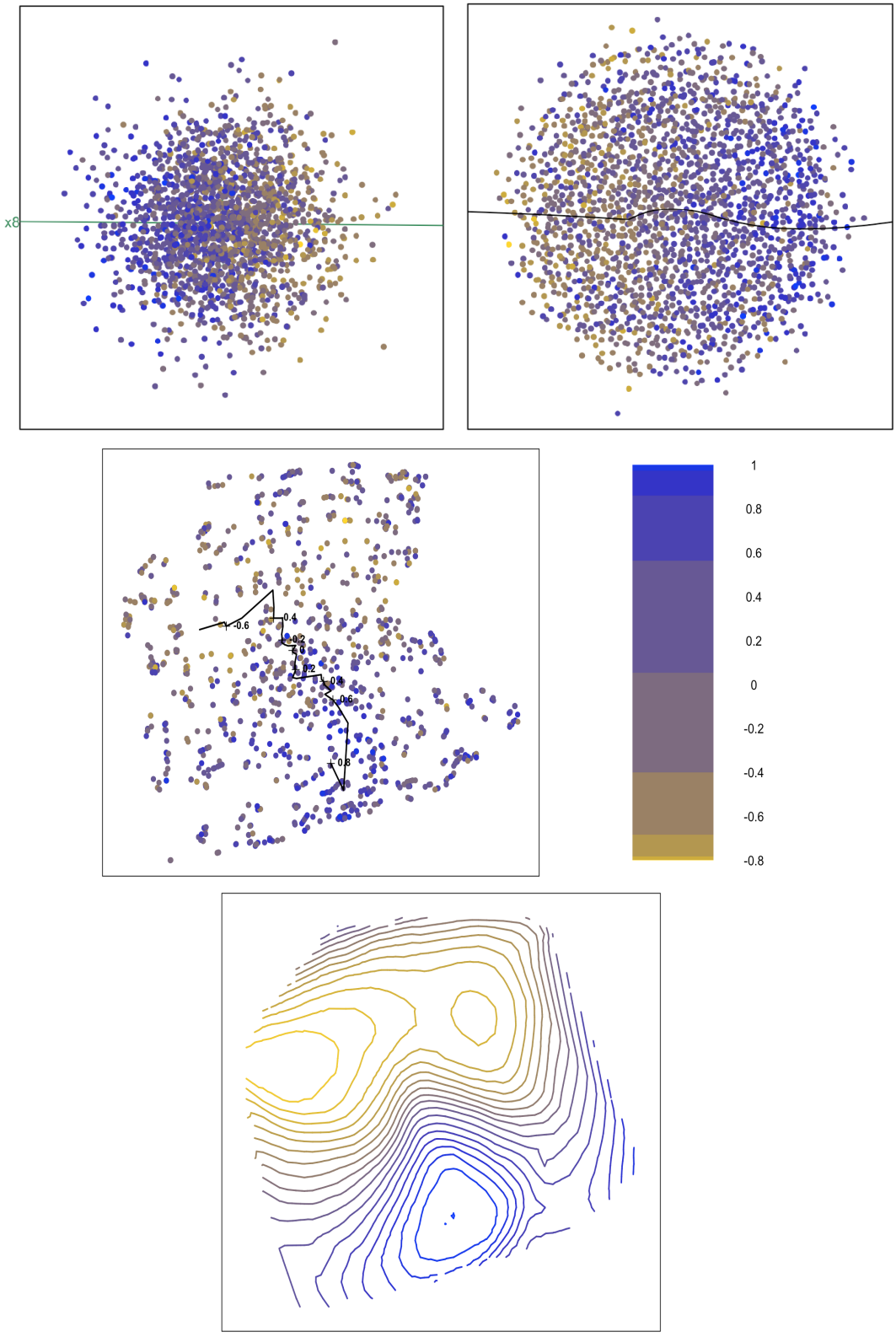

FIGURE 9.20: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_8 and sample points coloured according to the values of X_8.

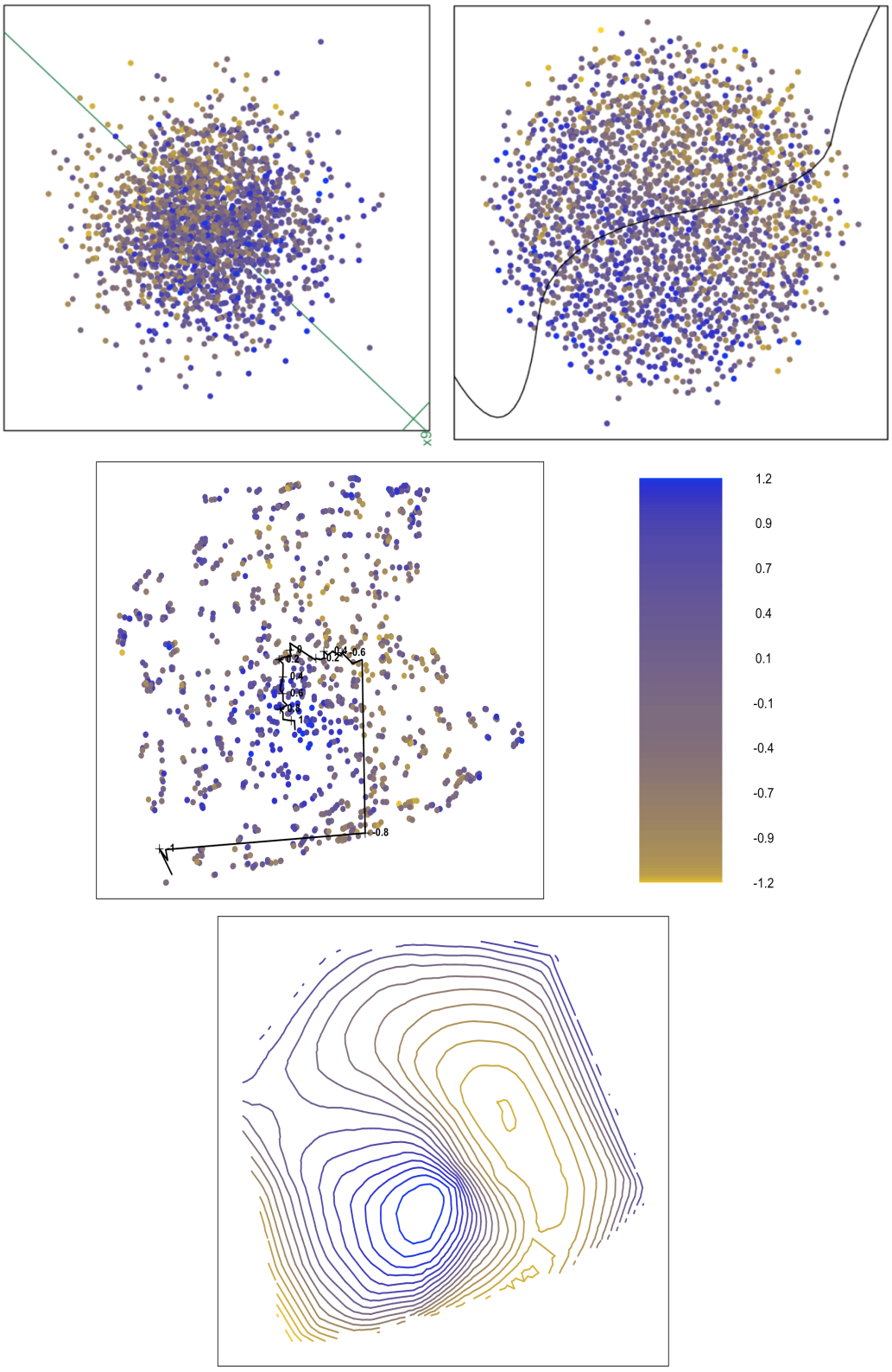

FIGURE 9.21: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_9 and sample points coloured according to the values of X_9.

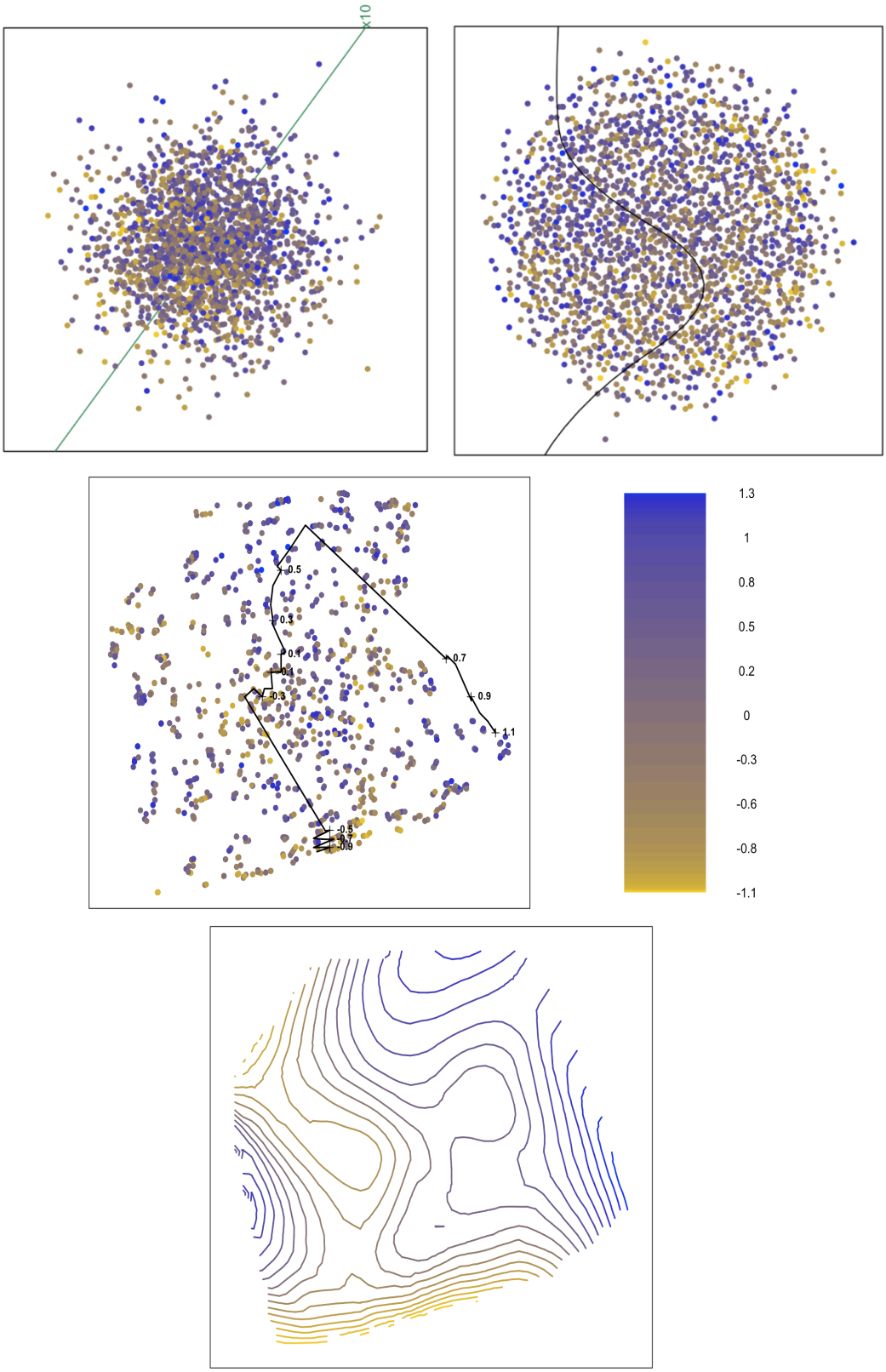

FIGURE 9.22: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_{10} and sample points coloured according to the values of X_{10}.

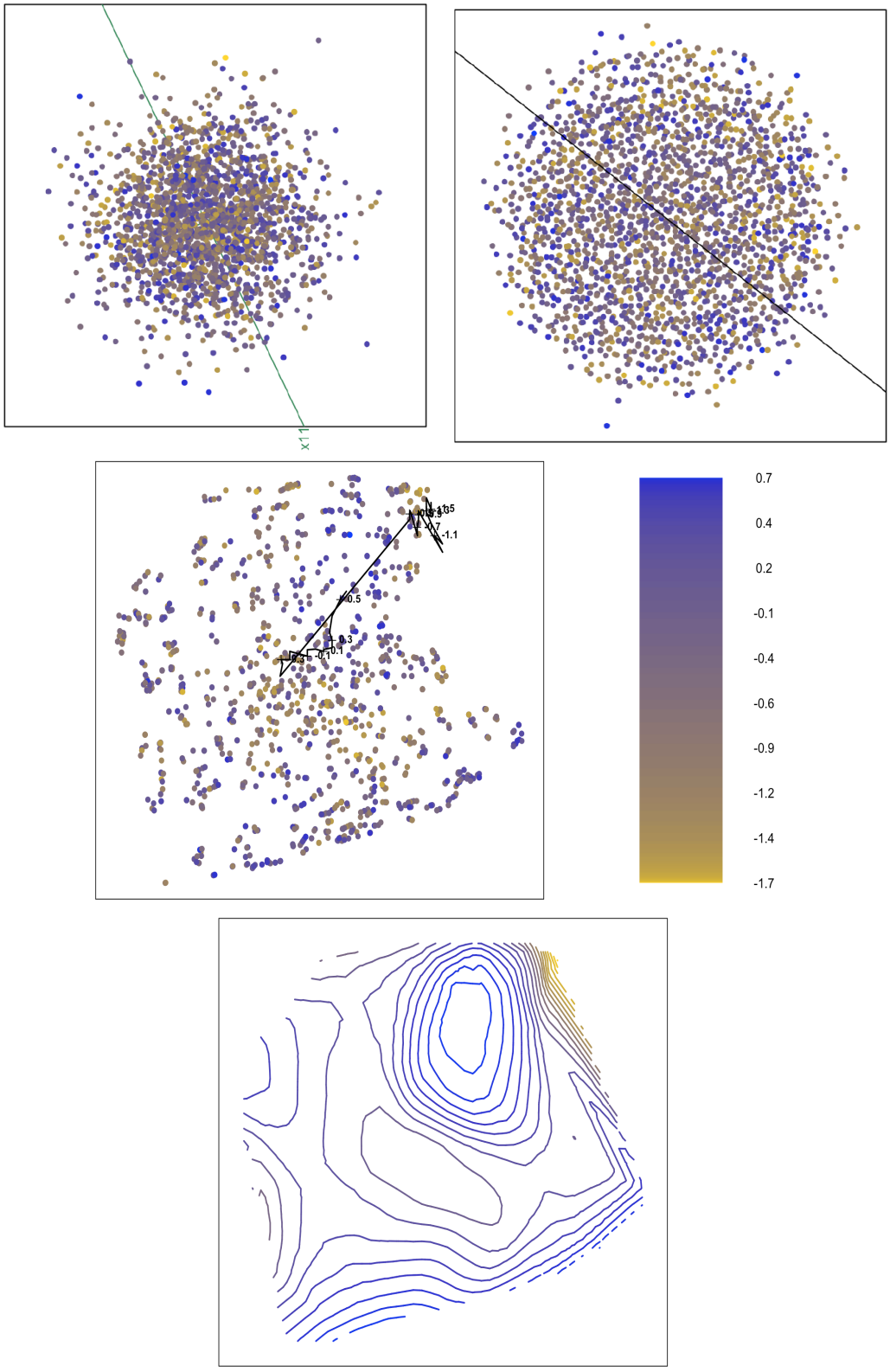

FIGURE 9.23: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_{11} and sample points coloured according to the values of X_{11}.

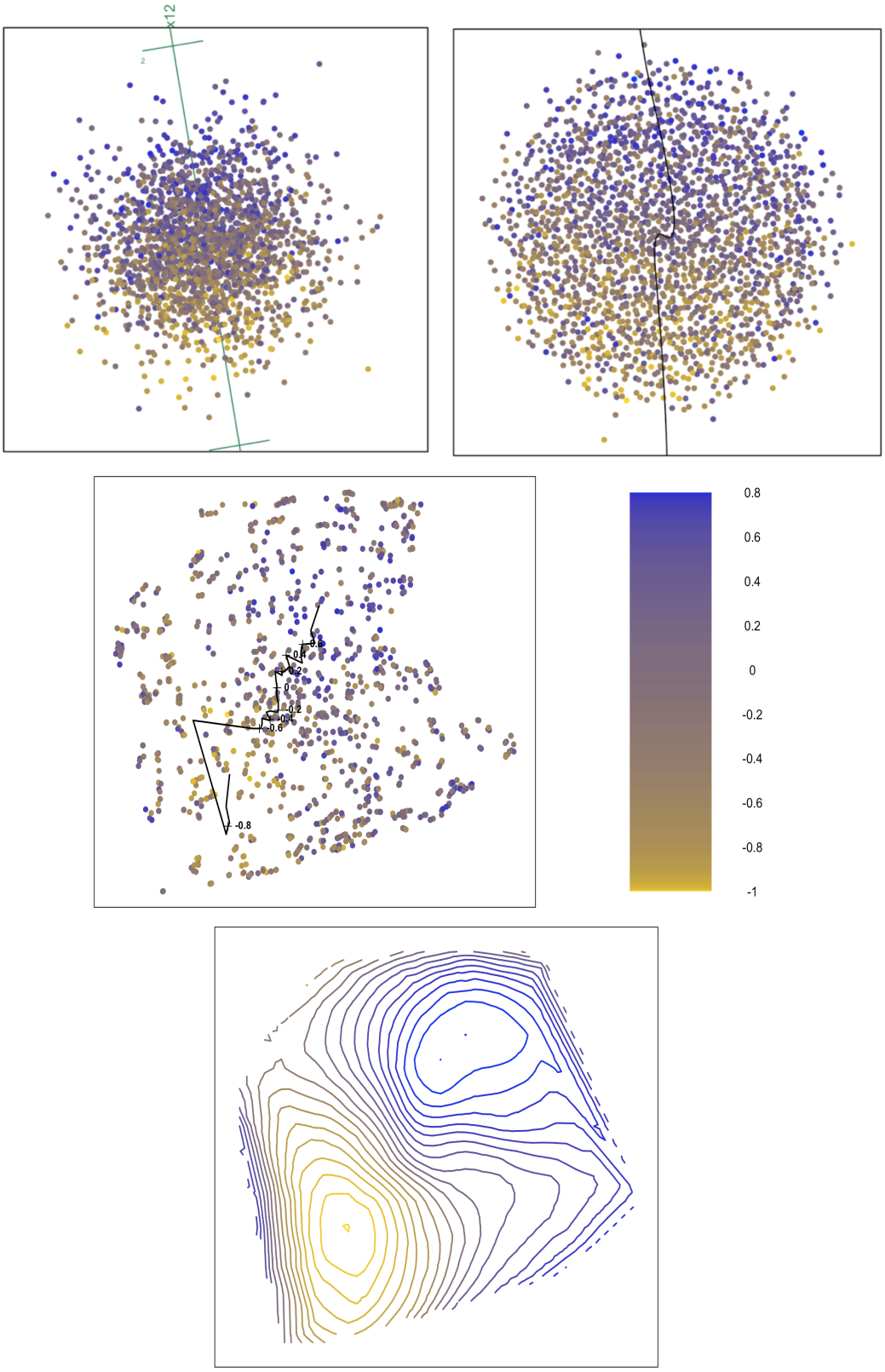

FIGURE 9.24: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_{12} and sample points coloured according to the values of X_{12}.

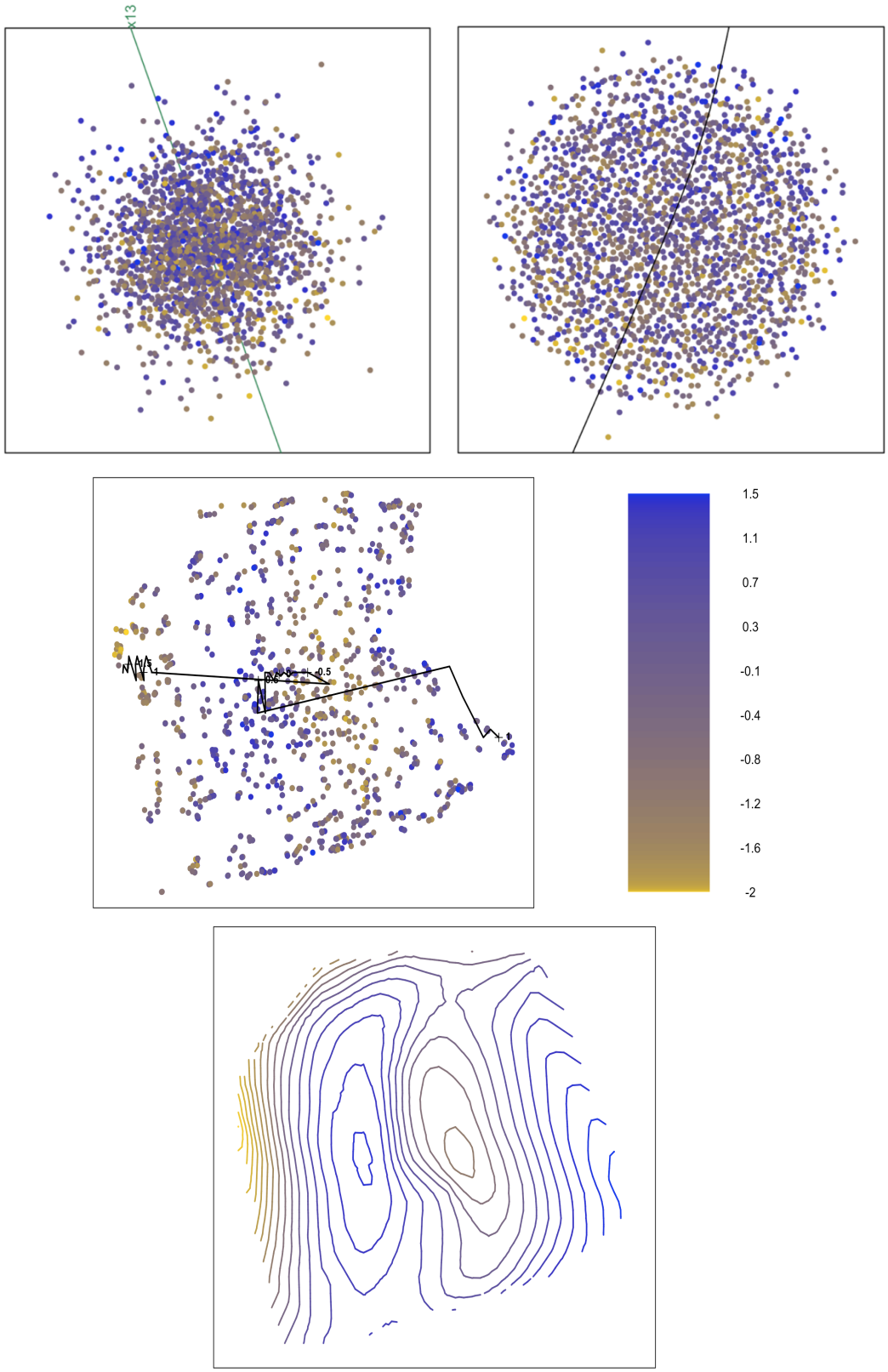

FIGURE 9.25: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_{13} and sample points coloured according to the values of X_{13}.

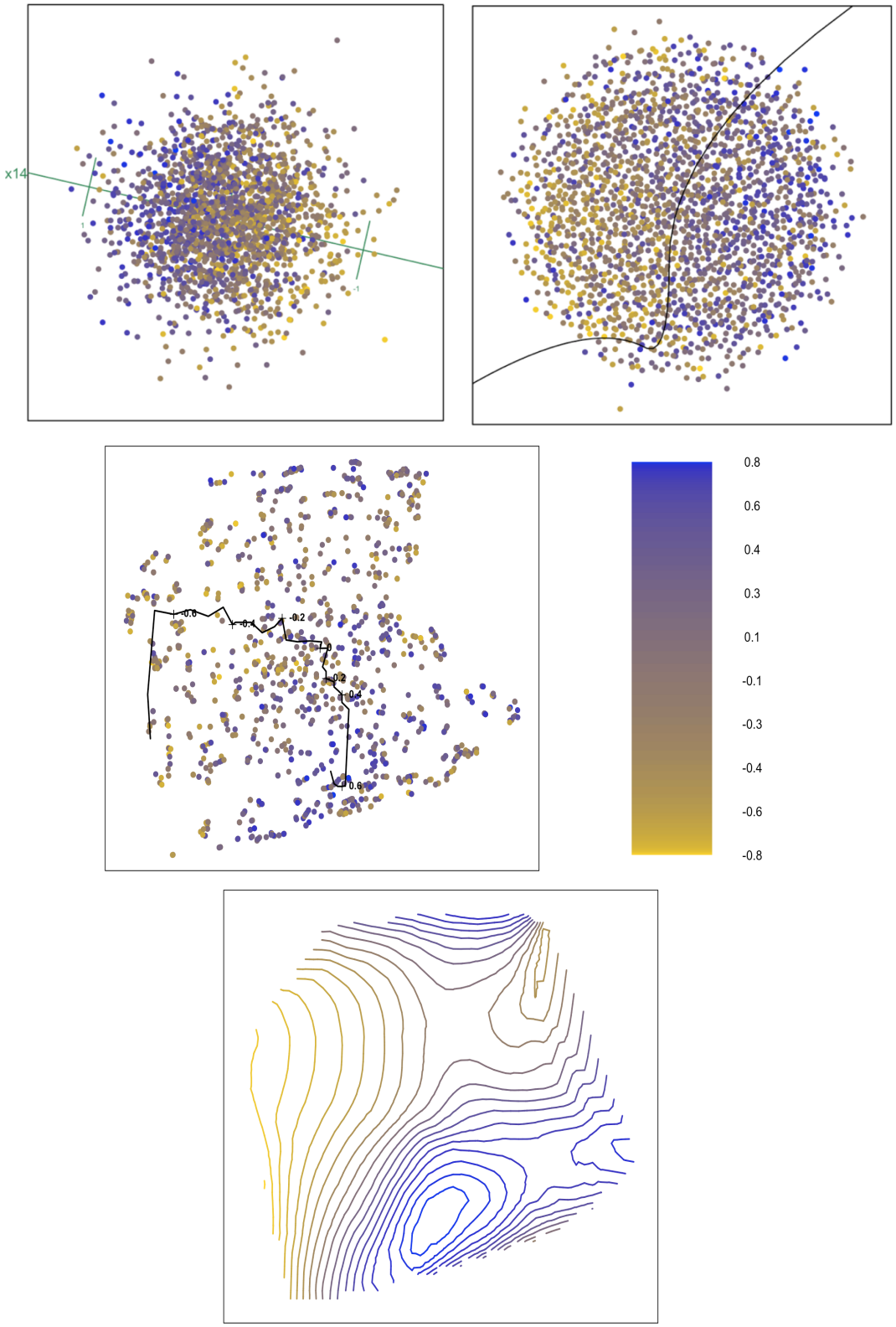

FIGURE 9.26: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_{14} and sample points coloured according to the values of X_{14}.

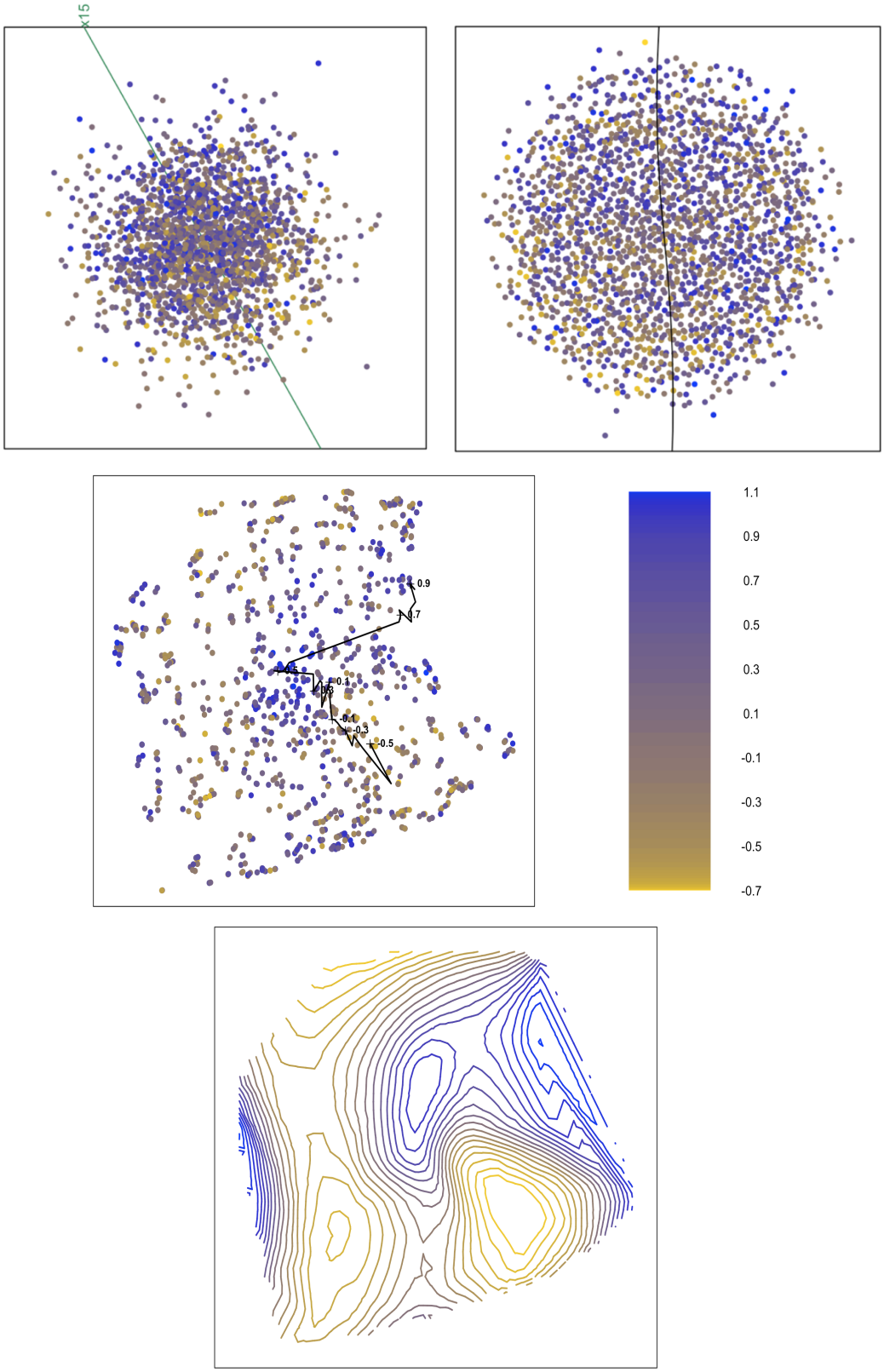

FIGURE 9.27: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_{15} and sample points coloured according to the values of X_{15}.

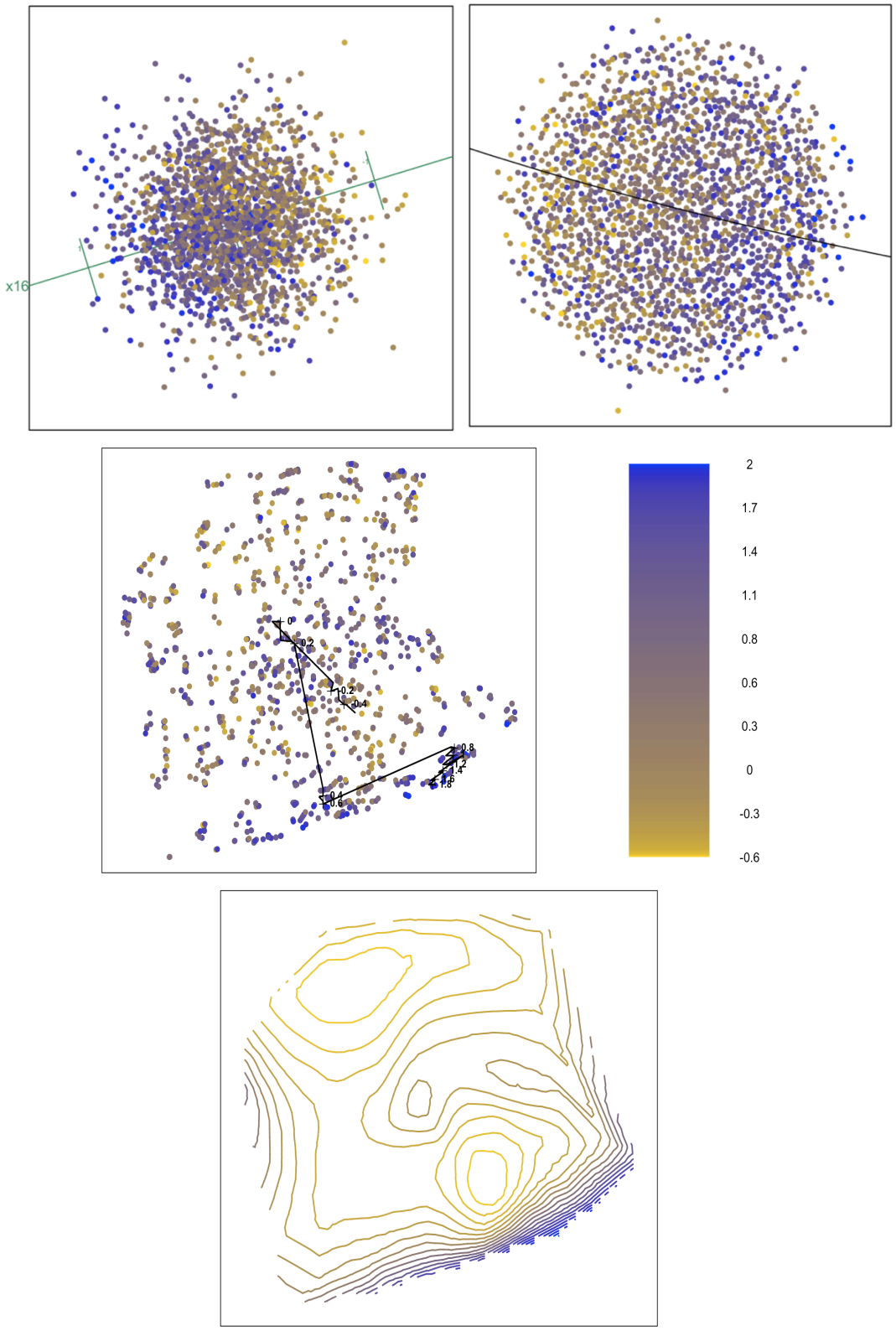

FIGURE 9.28: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_{16} and sample points coloured according to the values of X_{16}.

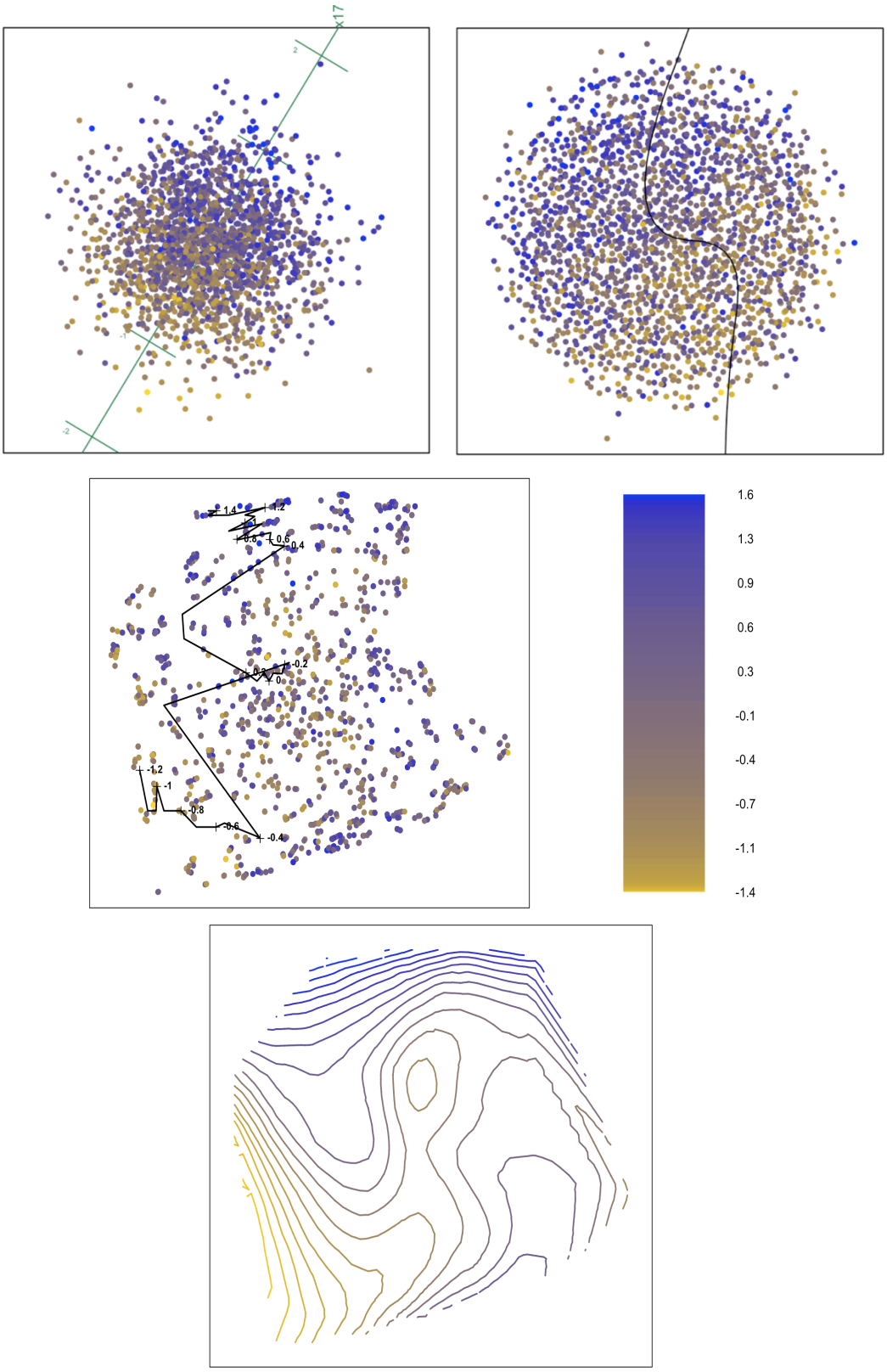

FIGURE 9.29: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_{17} and sample points coloured according to the values of X_{17}.

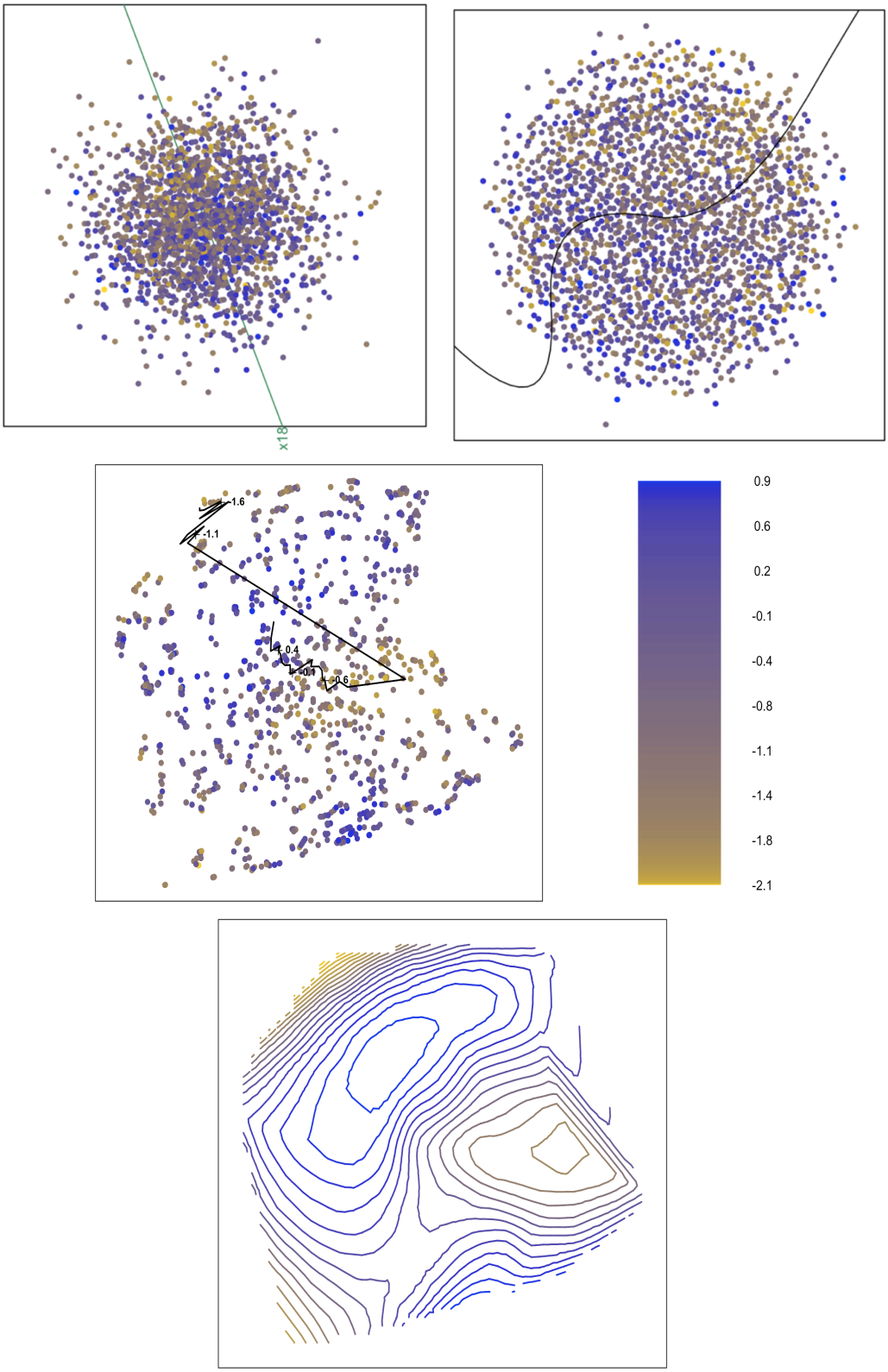

FIGURE 9.30: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_{18} and sample points coloured according to the values of X_{18}.

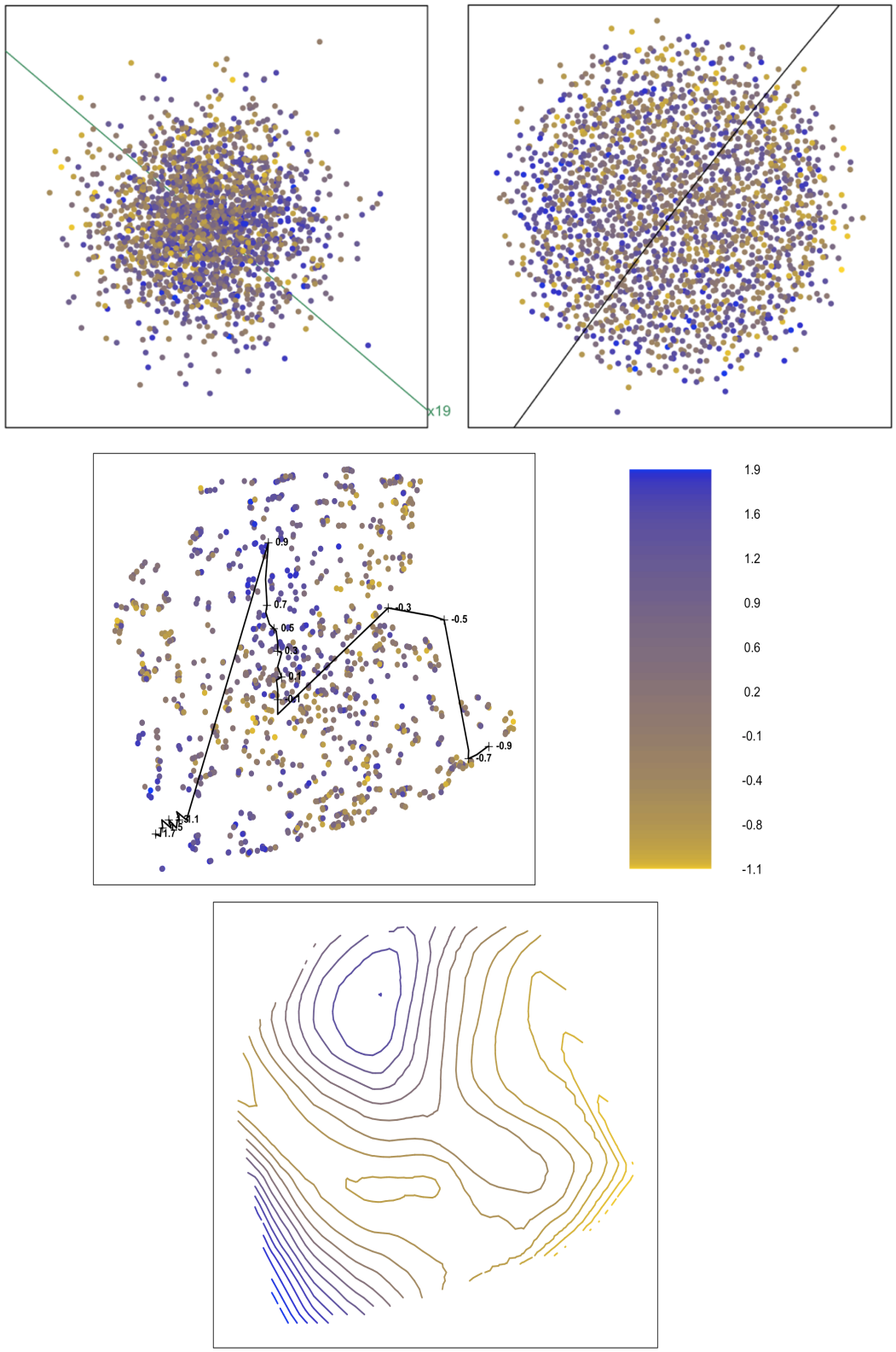

FIGURE 9.31: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_{19} and sample points coloured according to the values of X_{19}.

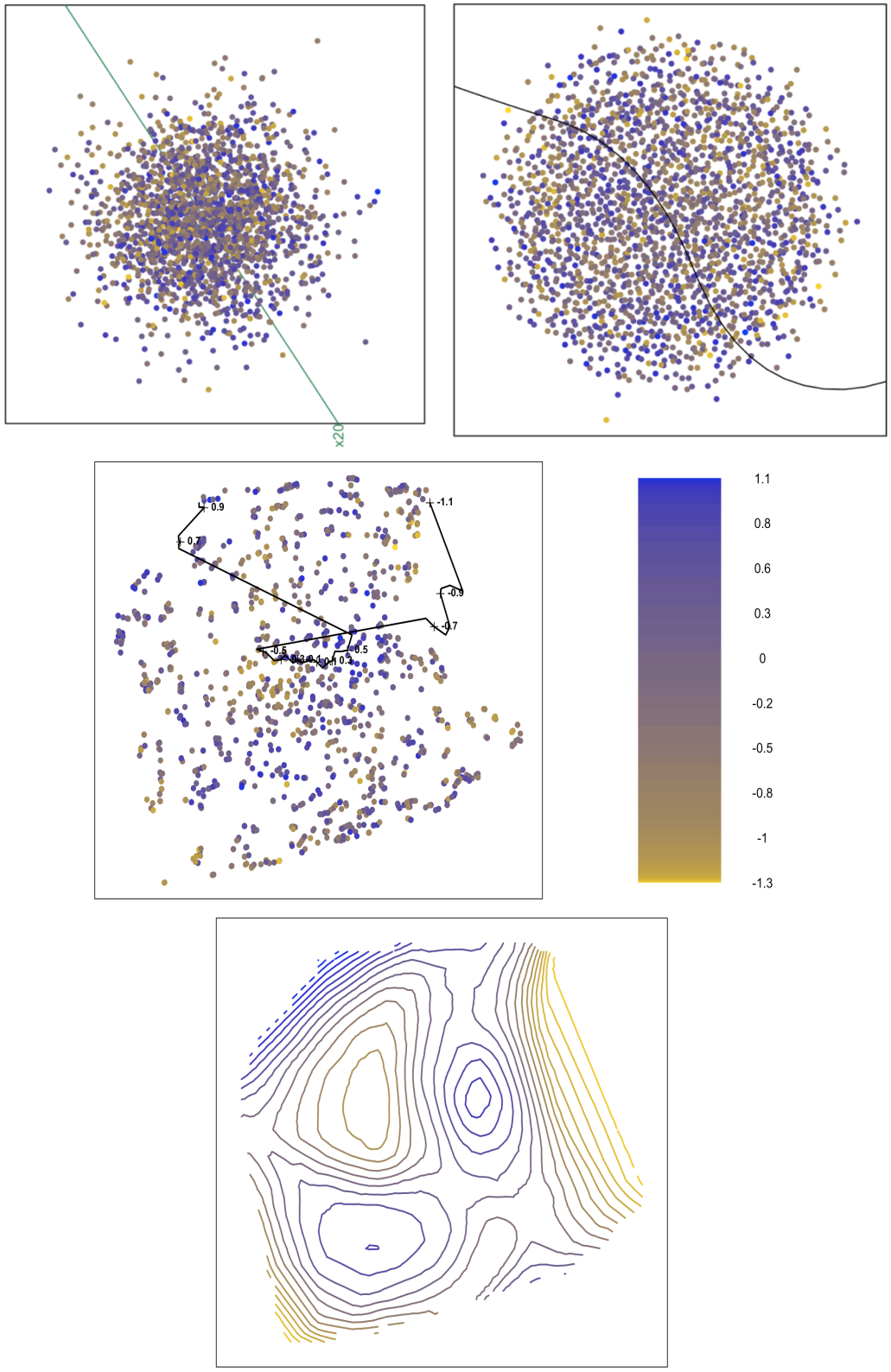

FIGURE 9.32: PCA biplot, spline-based nonlinear biplot, principal surface biplot and contour lines of the Madelon data (from left to right), with biplot axis X_{20} and sample points coloured according to the values of X_{20}.

9.6 Summary

This chapter presented contributions that the principal surface biplot can make when applied to different types of data. The principal surface biplot when utilised with sets of contours for variables, for prediction, is found to predict samples better than any other biplot especially for nonlinear data. The contributions provide interesting and valuable features of higher dimensional data. This is mainly due to the flexibility of the principal surface that follows the data, which makes it a more accurate idea. Measures such as means and medians are appropriate for describing centrality. It is not true, that centrality is the only important aspect for analysts. Visualisation methods, in contrast to summary statistics, is particularly appropriate to represent the whole data set, including the outer edges where deviations from linearity are particularly interesting.

The principal surface biplot, either used with variable axes or sets of contours give insight into the data, in neat and concise two-dimensional plots. Some of these insights, highlighted in the sections of this chapter included: nonlinear axes, class separation, better prediction of sample points including potential outliers and sample trends or behaviour through colour gradients.

Chapter 10

Conclusion

In this thesis, the focus was on the visualisation of multivariate data that exists in higher dimensions. Two main techniques in visualisation were used, namely principal surfaces and biplots. The idea was to create a new visualisation technique by combining the ideas of principal surfaces and biplots.

Principal surfaces was introduced in Chapter 2. The chapter gave a theoretical overview of principal surface for probability distributions and for data sets. The algorithm was demonstrated by constructing principal surfaces for different data sets. Principal surfaces serve the purpose of providing a nonlinear summary of higher dimensional data. Chapters 3, 4 and 5 were dedicated to understanding different types of biplots. This included PCA biplots, nonlinear biplots and spline-based nonlinear biplots, respectively. Application using different data sets to these biplot techniques was also illustrated.

The heart of the thesis lies in Chapter 6, 7 and 8 where the new method of principal surface biplots was introduced. Chapter 6 introduced the idea of using the principal surface as a nonlinear biplot plane. Samples are represented in the two-dimensional space by the λ coordinates of the principal surface. Intersection spaces are then found from the constructed normal planes and the biplot surface. Chapter 7 illustrated different designs of constructing biplot axes, and found that the design of back-projection provided the best set of biplot axes in terms of sample predictions. Chapter 8, defined a measure of prediction accuracy using a squared standardised prediction error (SSPE) to be used to compare the prediction of samples for different biplots. It then demonstrated that centering the surface proved to improve prediction of sample points in the principal surface biplot.

The overall prediction of samples in a principal surface biplot in Chapter 8 indicated that nonlinear contours can lead to biplot axes providing poorer predictions than strictly linear PCA biplots or other forms of biplots. In Chapter 9, the prediction of samples is rather found using the contour lines. This gave better and more accurate predictions. The chapter outlined four contributions the principal surface biplot can make to multivariate visualisation. The contributions listed were (but not limited to):

- Nonlinearity of axes: The principal surface biplot will reduce to a PCA biplot when data is described linearly, thus indicating that the principal surface biplot shows the extent of nonlinearity (if any) in a data set.
- Prediction of edge points: Edge points in data are better catered for in a principal surface biplot than in a PCA biplot, as the surface is nonlinear and does not remain in the same direction through all the samples.

- Separation of classes: The representation of samples in a principal surface biplot show better separation of classes, thus resulting in more concrete conclusions of trends in samples.
- Nonlinear prediction: When the representation of samples in a principal surface biplot are plotted according to a colour gradient for a variable, structures are more visible in the samples and patterns are more easily distinguishable. Thus, illustrating that the representation of samples have more meaning and create an additional way of representing a variable. The principal surface biplot has the added feature of demonstrating the extent of nonlinearity in the data, which the PCA and spline-based nonlinear biplot does not show.

The PCA biplot is simple with linear axes and provides a single representation of all variables and samples. The principal surface biplot provides a better representation in cases of nonlinearity which other biplots does not show. The trade-off is that the contour plots have to be made for each variable. With any new research, there is room for improvement and further work that can be done. The next section ends this thesis by adding a few notes on further research.

10.1 Further Research

- An alternative and better solution for a biplot axis would need to be found for variables in data that produce nonlinear and especially elliptical contour lines. For example, in the half sphere data from Chapter 2, variable x_1 will have elliptical-looking contours, which currently makes it difficult for all biplots (not only the principal surface biplot) to construct a biplot axis. The contour lines for variable x_1 can be seen in Figure 10.1.

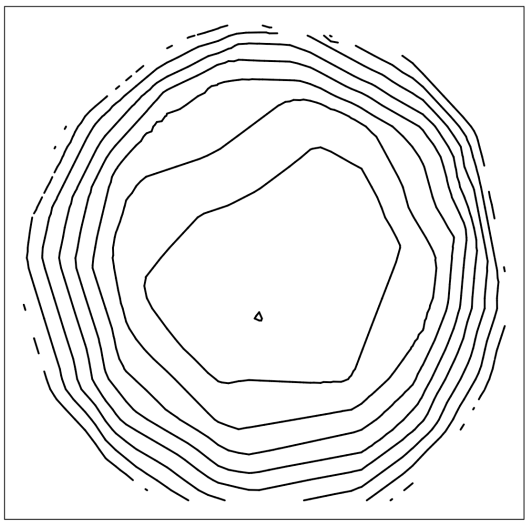

FIGURE 10.1: Contour lines for variable x_1 in the half sphere data in Chapter 2.

- To find a solution to create smoother biplot axes. An attempt to obtain this was done through a parametric solution of the principal surface, namely

generative topographical mapping (GTM) (Bishop, Svensén, and Williams, 1998) and probabilistic principal surfaces (PPS) (Chang and Ghosh, 1999). However, it was found that even though these methods produced smoother contour lines, it did not lead to smoother axes nor better prediction. The non-smooth axes in all attempts, comes from the poor join of marker points on the contour lines that does move along the principal surface. For example, from the blanket surface data; variable Y has a poor join of points. This can be seen in Figure 10.2. The sample points are plotted in shades of colour from green to red to indicate the quality of predictability. The set of red points on the left side of the principal surface possibly predicts poorly because of the join between the two marker points that lie far apart on the surface.

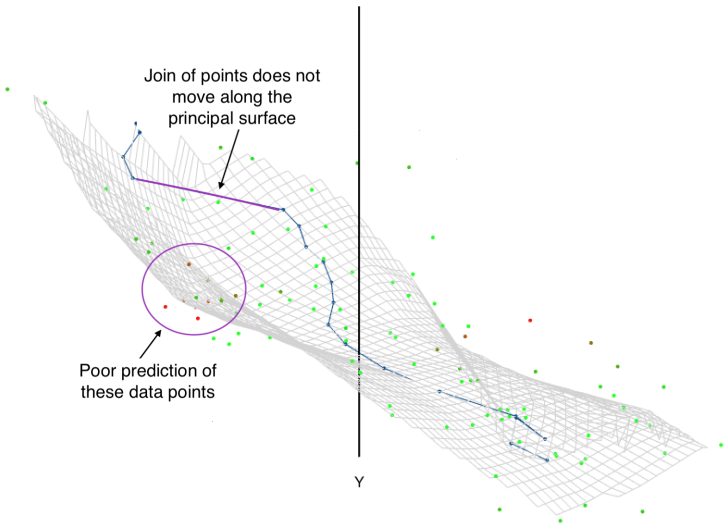

FIGURE 10.2: Principal surface with original Y axis shown in black, and biplot axis on surface shown in blue. Sample points shown in colour gradients from green to red, where green points indicate a better prediction than the red points.

- An alternative solution to smoother biplot axes is either to create more marker points, which essentially creates more sets of contour lines (this becomes computationally expensive) or to interpolate new marker points between the marker points on the biplot axis. However, this does not guarantee a smooth axis. The added computation time of creating more marker points is also not worth the effort, since the biplot axes does not represent the contours adequately in the presence of nonlinearity in any case.
- The back-projection method finds a point on a contour line that is closest to the respective marker point μ in the p-dimensional space. It would be of interest to see how far the other points on the contour line is to μ. By some measure of closeness, a band can be created around the chosen marker point. This is illustrated in Figure 10.3 by the orange points and lines. Ideally, one would like to see that the orange points are close to the respective chosen blue points. If the orange points are further from the blue points, then a better marker point on the contour line can be chosen. This could ultimately change the orientation of the axis in the hope to improve prediction.

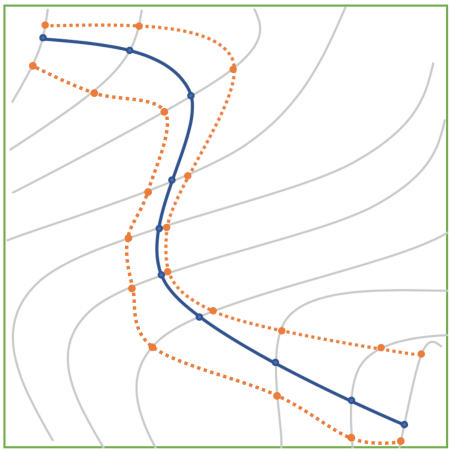

FIGURE 10.3: Contour lines for a certain variable shown by grey lines, biplot axis through back-projection shown in blue and band of closeness shown in orange.

- Cluster variables with similar shaped contours to find a simpler representation than having p individual plots, but rather only k plots where k is the number of clusters.
- Expand the principal surface methodology to a supervised classification setting, such as Canonical Variance Analysis (CVA).

10.2 Concluding Remarks

From this thesis, the significant finding is the new concept of a principal surface biplot. The principal surface biplot together with predicting samples like any other biplot in the literature, is shown to provide interesting features to the visualisation of higher dimensional data. The principal surface biplot focuses on nonlinearity and can follow the data, where the PCA biplot and even nonlinear biplots are only linear approximations. The principal surface biplot has the advantage nonlinearity, but that complicates construction of axes. It is not universally the best visualisation option, but as a non-exhaustive set of examples in Chapter 9 show, it assists an analyst with better insight into the data.

In the end, data visualisation is a means to communicate statistical results. Thus, interpretability of the visualisation is of importance. For users, the principal surface biplot may be of interest as a source of research and useful examples, or as a means of presenting certain types of data in informative and give background to make valid interpretations. It is assumed in the interpretation of these biplots, that reasonable understanding in constructing these biplots is taken, to communicate the visualisations successfully.

Appendix A

R Code

A.1 Principal surface function

```
# Principal Surface
principal.surface <- function(X, max.iter = 10, alpha = 0.6,N=50)
{
  # Purpose of function: Construct a principal surface to a p-dimensional data
      set X.

  # Arguments:
  # X: data of size n x p
  # max.iter: maximum number of iterations in the principal surface algorithm if
      the algorithm does not converge.
  # alpha: span argument in loess() function
  # N: creates an N^2 x p interpolated f_grid surface

  # Output:
  # fj.mat: final principal surface fitted - f(lambda) n x p matrix
  # lambda.j: representation of samples in two dimensions
  # fgrid: list of p, each of size N^2.
  # fgrid2 and fgrid3: interpolated principal surface of size N^2 x p
  # N: value of N selected in arguments

  require(rgl)
  require(Matrix)
  require(akima)
  n <- nrow(X)
  p <- ncol(X)

  # Initial Step
  X0 <- scale(X, scale = F)
  SVD <- svd(X0)
  # Set f(0) = A * lambda
  fj.mat <- X0 %*% SVD$v[, 1:2] %*% t(SVD$v[, 1:2]) + matrix(1,
                                                             ncol = 1, nrow = n)
                                                                 %*% apply(X, 2,
                                                                 mean)

  SVD2 <- svd(fj.mat)
  lambda.j <- fj.mat %*% SVD2$v[, 1:2]

  sumD <- sum(diag((fj.mat - X) %*% t(fj.mat - X)))
  eps <- 0.001
  count <- 0
  finish <- F

  while (!finish && count < max.iter) {
    # Expectation Step
    # Set f(k)(.) = E[...]
    for (j in 1:p) fj.mat[, j] <- fitted(loess(X[, j] ~ lambda.j[,
                                                                 1] + lambda.j[,
                                                                     2], span =
                                                                     alpha))

    PP <- PP2 <- matrix(0, nrow = n, ncol = p)
    distances = as.matrix(dist(fj.mat))
    max_dist = max(distances)
    distances[distances == 0] <- max_dist
    for (i in 1:n) {
```

```
      fA <- c() ; fB <- c() ; c.vec <- c() ; d.vec <- c()
      for (k in 1:n) {
        close_flam = sort(distances[k, ])
        flamA = as.numeric(names(close_flam)[1])
        flamB = as.numeric(names(close_flam)[2])
        b1 <- fj.mat[flamA, ] - fj.mat[k, ]
        b2 <- fj.mat[flamB, ] - fj.mat[k, ]
        a <- X[i, ] - fj.mat[k, ]
        B = cbind(b1, b2)
        c1 = as.numeric((t(a) %*% b1 %*% (t(b2) %*% b2) -
                           t(a) %*% b2 %*% (t(b1) %*% b2))/det(t(B) %*%
                                                                 B))
        d1 = as.numeric((t(a) %*% b2 %*% (t(b1) %*% b1) -
                           t(a) %*% b1 %*% (t(b2) %*% b1))/det(t(B) %*%
                                                                 B))
        if (is.na(c1))
          c1 = 0
        if (c1 < 0)
          c1 = 0
        if (c1 > 1)
          c1 = 1
        if (is.na(d1))
          d1 = 0
        if (d1 < 0)
          d1 = 0
        if (d1 > 1)
          d1 = 1
        P <- c1 * b1 + d1 * b2
        PP[k, ] <- P + fj.mat[k, ]
        fA[k] = flamA ; fB[k] <- flamB ; c.vec[k] = c1 ; d.vec[k] = d1
      }
      dik <- as.matrix(dist(rbind(X[i, ], PP)))[1, -1]
      small.d <- which.min(dik)
      PP2[i, ] <- PP[small.d, ]
    }
    fj.mat <- PP2

    # Projection step
    # Choose lambda(k) = lambda_f(k)
    SVD2 <- svd(fj.mat)
    lambda.j <- fj.mat %*% SVD2$v[,1:2]

    # Calculate distances from surface to data
    # Evaluate D2(k)
    sumD.new <- sum(diag((fj.mat - X) %*% t(fj.mat - X)))
    eps1 = abs(sumD.new - sumD)/sumD
    if (eps1 < eps)
      finish <- T
    sumD <- sumD.new
    count <- count + 1
    print(c(count, eps1, sumD))
  }

  f.grid <- list(len = p)
  nn = N
  for (j in 1:p) {
    out <- interp(x = lambda.j[, 1], y = lambda.j[, 2], z = fj.mat[,j],
                  xo = (seq(min(lambda.j[, 1]), max(lambda.j[,1]), length = nn)),
                  yo = (seq(min(lambda.j[, 2]), max(lambda.j[,2]), length = nn)),
                  linear = T, extrap = F,duplicate = "median")$z
    out[out > max(fj.mat[, j])] <- max(fj.mat[, j])
    out[out < min(fj.mat[, j])] <- min(fj.mat[, j])
    f.grid[[j]] <- out
  }


  f.grid2 = matrix(NA, nrow = nn^2, ncol = p)
  f.grid3 = matrix(NA, nrow = nn^2, ncol = p)
  index = 0
  for (i1 in 1:nn) for (i2 in 1:nn) {
    index = index + 1
    for (j in 1:p) {
```

```
      f.grid2[index, j] = f.grid[[j]][i1, i2]
      f.grid3[index, j] = f.grid[[j]][i2, i1]
    }
  }

  D.mat <- matrix(NA, nrow = (nn - 1)^2, ncol = nn)
  I1 = c()
  I2 = c()
  block = c()
  for (i1 in 2:nn) {
    for (i2 in 2:nn) {
      l11 <- i1 - 1
      l12 <- i1
      l21 <- i2 - 1
      l22 <- i2
      b1 <- b2 <- zero.point <- rep(NA, p)
      A <- X
      for (j in 1:p) {
        b1[j] <- f.grid[[j]][l12, l21] - f.grid[[j]][l11,
                                                     l21]
        b2[j] <- f.grid[[j]][l11, l22] - f.grid[[j]][l11,
                                                     l21]
        zero.point[j] <- f.grid[[j]][l11, l21]
        A[, j] <- X[, j] - f.grid[[j]][l11, l21]
      }
      mat <- rbind(0, b1, b1 + b2, b2)
      mat <- scale(mat, center = -zero.point, scale = F)
    }
  }
  lambda.grid <- f.grid2 %*% SVD2$v[,1:2]

  SVD2 = SVD2

  points3d(X,col="red")
  lines3d(f.grid2,col="lightgrey")
  lines3d(f.grid3,col="lightgrey")

  plot(lambda.j, col="seagreen",xlab = expression(lambda[1]),ylab=expression(
      lambda[2]),pch=16)
  return(list(X = X, fj.mat = fj.mat, f.grid2 = f.grid2, f.grid3=f.grid3,lambda.
      grid = lambda.grid,
              lambda.j = lambda.j, f.grid = f.grid,N=N))
}
```

A.2 Intersection spaces

```
intersection_v3 <- function(k=1,pos=1,nom=10,prettyn = 10, legend=T,Col="black",
    frid_C)
{
  # Purpose of function: Create intersection spaces

  # Arguments:
  # k: numerical value, indicating which variable to create intersection spaces
      for
  # pos: position of label for each contour line
  # nom: number of markers to create
  # prettyn: n argument in pretty() function
  # legend: logical - whether to add a legend or not
  # Col: colour of contour lines
  # frid_C: logical - whether the surface is centered or not.

  # Output:
  # all_contour_points: all points on the contour lines for variable k
  # lable_punte: which marker points are pretty. 1: non-pretty 2: pretty
  # punte: all marker points


  ax <- f.grid[,k]
  stP = round(min(na.omit(ax)),2)
```

```
enP = round(max(na.omit(ax)),2)
punte_seq = round(seq(stP,enP,length.out=nom),2)

extra.bit <- diff(range(na.omit(ax)))/10
largemu <- seq(from=min(na.omit(ax)), to=max(na.omit(ax)), length=200)
pretty_punte <- pretty(largemu,n=prettyn)

lable_punte <- c(rep(1,nom),rep(2,length(pretty_punte)))
order_punte <- order(c(punte_seq,pretty_punte))

punte <- c(punte_seq,pretty_punte)[order_punte]
#punte <- c(pretty_punte)
lable_punte <- lable_punte[order_punte]

mean_x = round(mean(na.omit(lambda.grid[,1])),2)
mean_y = round(mean(na.omit(lambda.grid[,2])),2)
#text(mean_x,mean_y,labels="O")

colours <- c("brown","deepskyblue3","coral","darkorchid3","darkgoldenrod2",
             "forestgreen","olivedrab3","maroon3","palevioletred1","royalblue",
             "chocolate","cyan4","blueviolet","olivedrab3")

total_nom <- length(punte)
all_contour_points <- matrix(0,ncol=3,nrow=1)
marker <- matrix(NA,ncol=2,nrow=total_nom)
count = 0
for(m in 1:total_nom)
{
  if(lable_punte[m] == 2) count = count + 1
  points_int <- matrix(0,ncol=2,nrow=1)
  for(i in 1:(N^2-N-1))
  {
    take_out <- (1:N)*N
    # each block
    if(all(i != take_out))
    {
      A = f.grid[i,]
      B = f.grid[N+i,]
      C = f.grid[1+i,]


      lambdaA = lambda.grid[i,]
      lambdaB = lambda.grid[N+i,]
      lambdaC = lambda.grid[1+i,]

      # intersection
      mu = punte[m]

      if((is.na(A[1]) | is.na(B[1]) | is.na(C[1])) == FALSE)
      {
        # check if plane intersects tile
        if((A[k] < mu & B[k] < mu & C[k] < mu) | (A[k] > mu & B[k] > mu & C[k]
            > mu) == FALSE)
        {

          # equation 1
          lambda11 = lambdaA[1] + ((mu-A[k])/(C[k]-A[k]))*(lambdaC[1]-lambdaA
              [1])
          lambda12 = lambdaA[2] + ((mu-A[k])/(C[k]-A[k]))*(lambdaC[2]-lambdaA
              [2])

          # equation 2
          lambda21 = lambdaA[1] + ((mu-A[k])/(B[k]-A[k]))*(lambdaB[1]-lambdaA
              [1])
          lambda22 = lambdaA[2] + ((mu-A[k])/(B[k]-A[k]))*(lambdaB[2]-lambdaA
              [2])

          # equation 3
          lambda31 = lambdaC[1] + ((mu-C[k])/(B[k]-C[k]))*(lambdaB[1]-lambdaC
              [1])
          lambda32 = lambdaB[2] + ((mu-B[k])/(C[k]-B[k]))*(lambdaC[2]-lambdaB
              [2])
```

```
          # solution
          lambda_sol_1 <- lambda_sol_2  <- c()
          if((A[k] > mu & C[k] > mu & B[k] < mu) | (A[k] < mu & C[k] < mu & B[k
              ] > mu))
          {
            # 2 + 3
            lambda_sol_1 <- c(lambda21,lambda22)
            lambda_sol_2 <- c(lambda31,lambda32)
          }

          if((B[k] > mu & C[k] > mu & A[k] < mu)|(B[k] < mu & C[k] < mu & A[k]
              > mu))
          {
            # 1 + 2
            lambda_sol_1 <- c(lambda11,lambda12)
            lambda_sol_2 <- c(lambda21,lambda22)
          }

          if((A[k] > mu & B[k] > mu & C[k] < mu)|A[k] < mu & B[k] < mu & C[k] >
               mu)
          {
            # 1 + 3
            lambda_sol_1 <- c(lambda11,lambda12)
            lambda_sol_2 <- c(lambda31,lambda32)
          }

          lambda_sol_1 <- c(lambda_sol_1[[1]],lambda_sol_1[[2]])
          lambda_sol_2 <- c(lambda_sol_2[[1]],lambda_sol_2[[2]])
          points_int <- rbind(points_int,
                              c(lambda_sol_1[1],lambda_sol_1[2]),
                              c(lambda_sol_2[1],lambda_sol_2[2]))
          if(lable_punte[m] == 2) {
            #points(c(lambda_sol_1[1],lambda_sol_2[1]),c(lambda_sol_1[2],lambda
                _sol_2[2]),pch=".",col=Col)
            #lines(c(lambda_sol_1[1],lambda_sol_2[1]),c(lambda_sol_1[2],lambda_
                sol_2[2]),lwd=2,col=Col)
          }
        }
      }
    }
  }
}
points_int1 <- cbind(points_int,rep(m,nrow(points_int)))

points_int <- matrix(0,ncol=2,nrow=1)
for(i in 2:(N^2-N))
{
  take_out <- (1:N)*N + 1
  # each block
  if(all(i != take_out))
  {
    A = f.grid[N+i-1,]
    B = f.grid[N+i,]
    C = f.grid[i,]

    lambdaA = lambda.grid[N+i-1,]
    lambdaB = lambda.grid[N+i,]
    lambdaC = lambda.grid[i,]


    # intersection
    mu = punte[m]

    if((is.na(A[1]) | is.na(B[1]) | is.na(C[1])) == FALSE)
    {
      # check if plane intersects tile
      if((A[k] < mu & B[k] < mu & C[k] < mu) | (A[k] > mu & B[k] > mu & C[k]
          > mu) == FALSE)
      {

        # equation 1
```

```
            lambda11 = lambdaA[1] + ((mu-A[k])/(C[k]-A[k]))*(lambdaC[1]-lambdaA
                [1])
            lambda12 = lambdaA[2] + ((mu-A[k])/(C[k]-A[k]))*(lambdaC[2]-lambdaA
                [2])

            # equation 2
            lambda21 = lambdaA[1] + ((mu-A[k])/(B[k]-A[k]))*(lambdaB[1]-lambdaA
                [1])
            lambda22 = lambdaA[2] + ((mu-A[k])/(B[k]-A[k]))*(lambdaB[2]-lambdaA
                [2])

            # equation 3
            lambda31 = lambdaC[1] + ((mu-C[k])/(B[k]-C[k]))*(lambdaB[1]-lambdaC
                [1])
            lambda32 = lambdaB[2] + ((mu-B[k])/(C[k]-B[k]))*(lambdaC[2]-lambdaB
                [2])

            # solution
            lambda_sol_1 <- lambda_sol_2  <- c()
            if((A[k] > mu & C[k] > mu & B[k] < mu) | (A[k] < mu & C[k] < mu & B[k
                ] > mu))
            {
              # 2 + 3
              lambda_sol_1 <- c(lambda21,lambda22)
              lambda_sol_2 <- c(lambda31,lambda32)
            }

            if((B[k] > mu & C[k] > mu & A[k] < mu)|(B[k] < mu & C[k] < mu & A[k]
                > mu))
            {
              # 1 + 2
              lambda_sol_1 <- c(lambda11,lambda12)
              lambda_sol_2 <- c(lambda21,lambda22)
            }

            if((A[k] > mu & B[k] > mu & C[k] < mu)|A[k] < mu & B[k] < mu & C[k] >
                 mu)
            {
              # 1 + 3
              lambda_sol_1 <- c(lambda11,lambda12)
              lambda_sol_2 <- c(lambda31,lambda32)
            }

            lambda_sol_1 <- c(lambda_sol_1[[1]],lambda_sol_1[[2]])
            lambda_sol_2 <- c(lambda_sol_2[[1]],lambda_sol_2[[2]])
            points_int <- rbind(points_int,
                                c(lambda_sol_1[1],lambda_sol_1[2]),
                                c(lambda_sol_2[1],lambda_sol_2[2]))
            if(lable_punte[m] == 2) {
              #points(c(lambda_sol_1[1],lambda_sol_2[1]),c(lambda_sol_1[2],lambda
                  _sol_2[2]),pch=".",col=Col)
              #lines(c(lambda_sol_1[1],lambda_sol_2[1]),c(lambda_sol_1[2],lambda_
                  sol_2[2]),lwd=2,col=Col)
            }
          }
        }
      }

    }
    points_int2 <- cbind(points_int,rep(m,nrow(points_int)))

    points_intAA <- rbind(points_int1,points_int2)
    points_intAA <- points_intAA[-which(points_intAA[,1] == 0
                                        & points_intAA[,2]==0),]

    # # circular projection (biplot trajectory)
    dist_ = c()
    if(length(points_intAA)==3) marker[m,] <- c(points_intAA[1:2])
    else if(nrow(points_intAA)>1) {
      for(j in 1:nrow(points_intAA))
      {
        dist_[j] = sqrt(sum((points_intAA[j,1:2] - c(mean_x,mean_y)) ^ 2) )
```

```
      }
      pm = which.min(dist_)
      if(lable_punte[m] == 2) {
        #text(points_intAA[pm,1],points_intAA[pm,2],labels=round(pretty_punte[
            count],2),
        #     cex= 0.7,pos=pos,col="black",font=2)
        #points(points_intAA[pm,1],points_intAA[pm,2],pch=3,col=Col)
        #segments(mean_x,mean_y,points_intAA[pm,1],points_intAA[pm,2],lty=3)
      }
      marker[m,] <- points_intAA[pm,1:2]
    }


    all_contour_points <- rbind(all_contour_points,points_int1,points_int2)
  }

  if(legend==T) legend("bottomright",legend=c(round(pretty_punte,2)),pch=16,col=
      colours,cex=0.7)

  #zeros = apply(marker, 1, function(row) all(row !=0 ))
  #marker = marker[zeros,]

  all_contour_points <- all_contour_points[-which(all_contour_points[,1] == 0
                                                  & all_contour_points[,2]==0),]

  if(frid_C==T) punte = punte + round(scaled_f.grid[k],1)

  #points(x=marker[,1],y=marker[,2],col=Col,pch=3)
  #text(x=marker[,1],y=marker[,2],col="black",labels=round(punte,2),font=2,cex
      =0.8,pos=pos)
  #for(i in 1:nrow(marker)) lines(x=c(mean_x,marker[i,1]),y=c(mean_y,marker[i,2])
      ,lty=2)
  #lines(x=marker[,1],y=marker[,2],col=Col,lwd=1)


  return(list(all_contour_points=all_contour_points,lable_punte=lable_punte,punte
      =punte))
}
```

A.3 Principal surface biplot axes

```
PS_biplot_axes <- function(data,k=1,nom=10,prettyn = 10,color="red",ms = 0.4,posi
    =4,frid_C)
{

  # Purpose of function: Construct a principal surface biplot axes through back-
      projection

  # Arguments:
  # data: data of size n x p
  # k: numerical value, indicating which variable to create biplot axis for
  # nom: number of markers to create
  # prettyn: n argument in pretty() function
  # color: colour of biplot axis
  # ms: size of marker label on axis
  # posi: position of marker label on axis
  # frid_C: logical - whether the surface is centered or not.


  # Output:
  # biplot_axis: Biplot axis
  # marker_points: Marker points on biplot axis
  # contours_Vp: Contours in p dimensions
  # contours_V: Contours in 2 dimensions

  # Back projection
  n <- nrow(data)
  p <- ncol(data)
```

```
Intersection <- intersection_v3(k=k,nom=nom,prettyn=prettyn, legend=F,Col="grey
    ",frid_C=frid_C)
contours_V <- Intersection$all_contour_points # intersection lines in 2D

# Find contour points in 2d closest to lambda grid
dist_m <- c()
for(i in 1:nrow(contours_V))
{
  dist <- c()
  for(j in 1:nrow(lambda.grid2))
  {
    dist[j] <- sqrt(sum((contours_V[i,1:2]-lambda.grid2[j,])^2))
  }
  dist_m[i] <- which.min(dist)
}


# Create planes at different markers
punte <- Intersection$punte

markers_V <- matrix(0,ncol=p,nrow=length(punte))
markers_V[,k] <- punte   # points on axes

contours_Vp <- na.omit(f.grid)[dist_m,] # intersection points in p dimensions
contours_Vp <- cbind(contours_Vp,contours_V[,3])

# list of intersection points for different markers
list_contoursP <- list()
for(i in 1:length(punte))
{
  pp <- matrix(0,ncol=p,nrow=0)
  for(j in 1:nrow(contours_Vp))
    if(contours_Vp[j,(p+1)] == i) pp <- rbind(pp,contours_Vp[j,1:p])
    list_contoursP[[paste("p",i,sep="")]] <- pp
}

list_contours <- list()
for(i in 1:length(punte))
{
  pp <- matrix(0,ncol=2,nrow=0)
  for(j in 1:nrow(contours_V))
    if(contours_V[j,3] == i) pp <- rbind(pp,contours_V[j,1:2])
    list_contours[[paste("p",i,sep="")]] <- pp
}


# Project axes onto intersection line
project_d <- list()
for(i in 1:length(punte))
{
  dist <- c()
  if(nrow(list_contoursP[[i]]) != 0)
  {
    for(j in 1:nrow(list_contoursP[[i]]))
    {
      dist[j] <- sqrt(sum((markers_V[i,] - list_contoursP[[i]][j,])^2))
    }
    project_d[[paste("r",i,sep="")]] <- which.min(dist)
  }
  else
    project_d[[paste("r",i,sep="")]] <- NA

}

# projected points in p dimensions
project_pointsP <- matrix(NA,nrow=length(punte),ncol=p)
for(i in 1:length(punte))
{
  if(is.na(project_d[[i]]) == FALSE)
    project_pointsP[i,] <- list_contoursP[[i]][project_d[[i]],1:p]
}
```

```
  points3d(project_pointsP,col="dodgerblue4")
  lines3d(project_pointsP,col="dodgerblue4")

  # projected points in 2 dimensions
  project_points <- matrix(NA,nrow=length(punte),ncol=2)
  for(i in 1:length(punte))
  {
    if(is.na(project_d[[i]]) == FALSE)
      project_points[i,] <- list_contours[[i]][project_d[[i]],1:2]
  }

  #plot(lambda.grid,col="white",xlab = expression(lambda[1]),ylab=expression(
      lambda[2]))
  points(project_points[Intersection$lable_punte==2,],col=color,pch=3)
  #points(lambda.grid2[dist_m,],col="lightgrey",pch=16)
  text(project_points[Intersection$lable_punte==2,1],project_points[Intersection$
      lable_punte==2,2],round(punte[Intersection$lable_punte==2],2),cex=ms,pos=
      posi,col=color,font=2)
  lines(project_points,col=color,lwd=2)

  biplot_axis <- project_points
  marker_points <- round(punte,2)

  return(list(biplot_axis=biplot_axis,marker_points=marker_points,
              contours_Vp=contours_Vp,contours_V=contours_V))
}
```

A.4 Prediction of samples in principal surface biplot

```
PS_prediction <- function(data,nom=20,prettyn=10,k=1,predict,frid_C)
{
  # Purpose of function: Predict samples in the data through the constructed
      biplot axis.

  # Arguments:
  # data: data of size n x p
  # nom: number of markers to create
  # prettyn: n argument in pretty() function
  # k: numerical value, indicating which variable to predict samples on.
  # predict: list of which samples in the to predict
  # frid_C: logical - whether the surface is centered or not.

  # Output:
  # predicted_point: list of predicted sample points for variable k.

  distance_to_axis <- function(p1,p2,point)
  {
    a = point - p1
    b = p2 - p1
    ff <- sum(a * b)/sum(b * b)
    if(is.na(ff)) ff <- 0
    if(ff < 0) ff <- 0
    if(ff > 1) ff <- 1
    P <- ff * b + p1
    distance <- sqrt(sum((P - point)^2))
    #points(P[1],P[2])
    A = P - p1
    prop = sum(A*A)/sum(b*b)
    new_m <- marker_points[i] + prop*(marker_points[i+1]-marker_points[i])

    return(list(P=P,distance = distance,new_m=new_m))
  }

  predict_points <- lambda.j[predict,]

    # Back-projection
  create_axes <- PS_biplot_axes(data,k=k,nom=nom,frid_C=frid_C,prettyn=prettyn)
  biplot_axis <- create_axes$biplot_axis
  marker_points <- create_axes$marker_points
```

```
  predicted_point <- c()
  for(j in 1:length(predict))
  {
    dist <- c() ; P <- matrix(0,nrow=(nom-1),ncol=2) ; new_m = c()
    for(i in 1:(nom-1))
    {
      projections <- distance_to_axis(biplot_axis[i,],biplot_axis[i+1,],lambda.j[
          j,])
      dist[i] = projections$distance
      P[i,] = projections$P
      new_m[i] = projections$new_m
    }
    small.d <- which.min(dist)
    PP = P[small.d,]
    predicted_point[j] <- new_m[small.d]
  }
  return(list(predicted_point=predicted_point))
}
```

A.5 Prediction of samples in principal surface biplot using contour lines

```
PS_prediction_contours <- function(k=1,nom=10,prettyn=10,frid_C)
{
  # Purpose of function: Predict samples in the data through the contour lines.

  # Arguments:
  # k: numerical value, indicating which variable to predict samples on.
  # nom: number of markers to create
  # prettyn: n argument in pretty() function
  # frid_C: logical - whether the surface is centered or not.

  # Output:
  # predicted_point: list of predicted sample points for variable k.

  contours <- intersection_v3(k=k,nom=nom,prettyn=prettyn,legend=F,Col="grey",
      frid_C=frid_C)

  punte <- contours$punte
  allcontours <- contours$all_contour_points

  predict_points <- lambda.j[predict,]

  start <- allcontours[1,3]
  finish <- allcontours[nrow(allcontours),3]

  distance_to_axis <- function(p1,p2,point)
  {
    a = point - p1
    b = p2 - p1
    ff <- sum(a * b)/sum(b * b)
    if(is.na(ff)) ff <- 0
    if(ff < 0) ff <- 0
    if(ff > 1) ff <- 1
    P <- ff * b + p1
    distance <- sqrt(sum((P - point)^2))
    #points(P[1],P[2])
    A = P - p1
    prop = sum(A*A)/sum(b*b)
    new_m <- punte[j] + prop*(punte[j+1]-punte[j])

    return(list(P=P,distance = distance,new_m=new_m))
  }

  predicted_point <- c()
  for(i in 1:length(predict))
  {
```

```
    dist <- c() ; P <- matrix(NA,nrow=nom,ncol=2) ; new_m = c()
    for(j in start:(finish-1))
    {
      contour_a <- allcontours[allcontours[,3]==j,]
      contour_b <- allcontours[allcontours[,3]==(j+1),]

      dist_a <- c() ; dist_b <- c()

      for(r in 1:nrow(contour_a)) dist_a[r] <- sqrt(sum((contour_a[r,1:2]-lambda.
          j[i,])^2))
      for(r in 1:nrow(contour_b)) dist_b[r] <- sqrt(sum((contour_b[r,1:2]-lambda.
          j[i,])^2))

      point_a <- contour_a[which.min(dist_a),1:2]
      point_b <- contour_b[which.min(dist_b),1:2]


      projections <- distance_to_axis(point_a,point_b,lambda.j[i,])
      dist[j] = projections$distance
      P[j,] = projections$P
      new_m[j] = projections$new_m
    }
    small.d <- which.min(dist)
    PP = P[small.d,]
    predicted_point[i] <- new_m[small.d]
  }

  return(list(predicted_point=predicted_point))
}
```

References

Akima, H. (1978). "A method of bivariate interpolation and smooth surface fitting for irregularly distributed data points". In: *ACM Transactions on Mathematical Software (TOMS)* 4.2, pp. 148–159.

Bishop, C.M, M. Svensén, and C.K.I. Williams (1998). "GTM: The generative topographic mapping". In: *Neural Computation* 10.1, pp. 215–234.

Borg, I. and P.J.F. Groenen (2005). "Springer series in statistics". In: *Modern multidimensional scaling: Theory and applications (2nd ed.). New York, NY, US: Springer Science+ Business Media*.

Brand, H. (2013). *PCA and CVA biplots: A study of their underlying theory and quality measures*. Tech. rep. Stellenbosch University.

Chang, K. and J. Ghosh (1999). "Probabilistic principal surfaces". In: *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339)*. Vol. 2. IEEE, pp. 1107–1112.

Clark, P.J. (1952). "An extension of the coefficient of divergence for use with multiple characters". In: *Copeia* 1952.2, pp. 61–64.

Cox, T.F. and M.A. Cox (2001). *Multidimensional scaling*. Chapman and hall/CRC.

De'ath, G. (1999). "Principal curves: a new technique for indirect and direct gradient analysis". In: *Ecology* 80.7, pp. 2237–2253.

Dheeru, D. and E.K. Taniskidou (2017). "UCI machine learning repository". In: *University of California, Irvine, School of Information and Computer Sciences*.

Fisher, R.A. (1936). "The use of multiple measurements in taxonomic problems". In: *Annals of Eugenics* 7.2, pp. 179–188.

Friendly, M. (2008). "A brief history of data visualization". In: *Handbook of Data Visualization*. Springer, pp. 15–56.

Gabriel, K.R. (1971). "The biplot graphic display of matrices with application to principal component analysis". In: *Biometrika* 58.3, pp. 453–467.

Ganey, R. (2014). *Principal points and surfaces*. Tech. rep. University of Cape Town.

Gower, J.C. (1968). "Adding a point to vector diagrams in multivariate analysis". In: *Biometrika* 55.3, pp. 582–585.

— (1982). "Euclidean distance geometry". In: *Mathematical Scientist* 7.1, pp. 1–14.

Gower, J.C. and D.J. Hand (1996). *Biplots*. Vol. 54. CRC Press.

Gower, J.C. and S.A. Harding (1988). "Nonlinear biplots". In: *Biometrika* 75.3, pp. 445–455.

Gower, J.C. and P. Legendre (1986). "Metric and Euclidean properties of dissimilarity coefficients". In: *Journal of Classification* 3.1, pp. 5–48.

Gower, J.C., S. Lubbe, and N.J. Le Roux (2011). *Understanding biplots*. John Wiley & Sons.

Gower, J.C. and R.F. Ngouenet (2005). "Nonlinearity effects in multidimensional scaling". In: *Journal of Multivariate Analysis* 94.2, pp. 344–365.

Gower, John C (1992). "Generalized biplots". In: *Biometrika* 79.3, pp. 475–493.

Greenacre, M.J. (2010). *Biplots in Practice*. Fundacion BBVA.

Groenen, P.J.F., N.J. Le Roux, and S. Gardner-Lubbe (2015). "Spline-based nonlinear biplots". In: *Advances in Data Analysis and Classification* 9.2, pp. 219–238.

Hastie, T. (1984). *Principal points, principal curves and principal surfaces*. Tech. rep. Stanford University.

Hastie, T. and W. Stuetzle (1989). "Principal curves". In: *Journal of the American Statistical Association* 84.406, pp. 502–516.

Hastie, T., R. Tibshirani, and J. Friedman (2009). "Unsupervised learning". In: *The Elements of Statistical Learning*. Springer, pp. 485–585.

Hotelling, H. (1933). "Analysis of a complex of statistical variables into principal components." In: *Journal of Educational Psychology* 24.6, p. 417.

Julian, I.A. (2008). *Modern multivariate statistical techniques: regression, classification, and manifold learning*. Springer, New York.

Pearson, K. (1901). "Principal components analysis". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 6.2, p. 559.

R Core Team (2018). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. URL: https://www.R-project.org/.

Tukey, J.W. (1962). "The future of data analysis". In: *The Annals of Mathematical Statistics* 33.1, pp. 1–67.

Vines, S.K. (2015). "Predictive nonlinear biplots: maps and trajectories". In: *Journal of Multivariate Analysis* 140, pp. 47–59.