UNIVERSITY OF
NEWCASTLE UPON TYNE

NONLINEAR PARTIAL LEAST SQUARES

Per Anker Hassel

A Thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

School of Chemical Engineering and Advanced Materials
University of Newcastle upon Tyne

2003

ABSTRACT

Partial Least Squares (PLS) has been shown to be a versatile regression technique with an increasing number of applications in the areas of process control, process monitoring and process analysis. This Thesis considers the area of nonlinear PLS; a nonlinear projection based regression technique. The nonlinearity is introduced as a univariate nonlinear function between projections, or to be more specific, linear combinations of the predictor and the response variables. As for the linear case, the method should handle multicollinearity, underdetermined and noisy systems. Although linear PLS is accepted as an empirical regression method, none of the published nonlinear PLS algorithms have achieved widespread acceptance. This is confirmed from a literature survey where few real applications of the methodology were found. This Thesis investigates two nonlinear PLS methodologies, in particular focusing on their limitations. Based on these studies, two nonlinear PLS algorithms are proposed.

In the first of the two existing approaches investigated, the projections are updated by applying an optimization method to reduce the error of the nonlinear inner mapping. This ensures that the error introduced by the nonlinear inner mapping is minimized. However, the procedure is limited as a consequence of problems with the nonlinear optimisation. A new algorithm, Nested PLS (NPLS), is developed to address these issues. In particular, a separate inner PLS is used to update the projections. The NPLS algorithm is shown to outperform existing algorithms for a wide range of regression problems and has the potential to become a more widely accepted nonlinear PLS algorithm than those currently reported in the literature.

In the second of the existing approaches, the projections are identified by examining each variable independently, as opposed to minimizing the error of the nonlinear inner mapping directly. Although the approach does not necessary identify the underlying functional relationship, the problems of overfitting and other problems associated with optimization are reduced. Since the underlying functional relationship may not be established accurately, the reliability of the nonlinear inner mapping will be reduced. To address this problem a new algorithm, the Reciprocal Variance PLS (RVPLS), is proposed. Compared with established methodology, RVPLS focus more on finding the underlying structure, thus reducing the difficulty of finding an appropriate inner mapping. RVPLS is shown to perform well for a number of applications, but does not have the wide-ranging performance of Nested PLS.

ACKNOWLEDGEMENTS

There are a number of people to whom I am indebted for their assistance. Without them the completion of this Thesis would not have been possible.

First of all, I would like to acknowledge my supervisors Professor Elaine B. Martin, and Professor Julian Morris, School of Chemical Engineering and Advanced Materials, who guided me through the PhD, commented and made corrections on the Thesis as well as the papers.

Furthermore, I would like to give special thanks to Dr. Kristian Helland, who convinced me to go ahead with the PhD in times when it seemed impossible, for valuable comments, and for helping me with the practicalities regarding Borealis. Furthermore, I would like to thank the people who contributed to this thesis by providing data sets, in particular Rune Mathisen (Borealis), Dr. Peter Wentzell (Dalhousie University), Dr. Marcel Blanco (Universitat Autònoma de Barcelona) and Miss Linda Stordrange (University of Bergen/Nycomed).

The author gratefully acknowledges the valuable assistance of Dr. Baibing Lee, whose collaboration both gave the direction and lifted the level of the thesis. Specially, the collaboration leading to the Nested PLS algorithm. Also thanks to Dr. Steven Lane who read through my Theses and corrected my English.

In addition, I want to acknowledge the financial support of Borealis and the financial support provided by The Norwegian Research Council (Project number 136515/221).

Finally, thanks to the CPACT secretary, Angela Bott, for helping me with all the practicalities necessary to make my life easy and a special thanks to my wife Kirsten Graf Hassel who accepted my absence from home during the time I stayed in Newcastle.

Francis Bacon (1561-1616)

If we begin with certainties, we shall end in doubts;
but if we begin with doubts, and are patient in them,
we shall end in certainties.

CONTENTS

NOTATION

Conventions

Y	Response variables or quality variables
X	Predictor variables
W (V)	Weight matrix, $\mathbf{W}=[\mathbf{w}_1\cdots\mathbf{w}_A]$, (**V** weights for inner PLS)
Q (G)	**Y** loading matrix, $\mathbf{Q}=[\mathbf{q}_1\cdots\mathbf{q}_A]$, (**G** loadings for inner PLS)
P (H)	**X** loading matrix, $\mathbf{P}=[\mathbf{p}_1\cdots\mathbf{p}_A]$, (**H** loadings for inner PLS)
T (S)	**X** score matrix, $\mathbf{T}=[\mathbf{t}_1\cdots\mathbf{t}_A]$, where $\mathbf{t}_j=\mathbf{X}_j\mathbf{w}_j$, (**S** for inner PLS)
U (R)	**Y** score matrix, $\mathbf{U}=[\mathbf{u}_1\cdots\mathbf{u}_A]$ where $\mathbf{u}_j=\mathbf{Y}_j\mathbf{q}_j$ (**R** for inner PLS)
$\hat{\mathbf{u}}$	Nonlinear function $\hat{\mathbf{u}}=f(\mathbf{t})$, where $\|\mathbf{u}-\hat{\mathbf{u}}\|$ is minimized
$\boldsymbol{\beta}$ ($\hat{\boldsymbol{\beta}}$)	Regression vector (estimate)
B ($\hat{\mathbf{B}}$)	Regression matrix (estimate)
$\boldsymbol{\Lambda}$	Matrix of eigenvalues $\boldsymbol{\Lambda}=\text{diag}(\lambda_j)$
$\boldsymbol{\Gamma}$	$\boldsymbol{\Gamma}=\mathbf{T}/\text{norm}(\mathbf{T})$ the normalized score vectors
$\boldsymbol{\Sigma}$	Matrix of singular values $\boldsymbol{\Sigma}=\text{diag}(\sqrt{\lambda_j})$
$\boldsymbol{\Omega}$	Weight matrix $\boldsymbol{\Omega}=[\boldsymbol{\varpi}_1\cdots\boldsymbol{\varpi}_A]$ in weighted least squares
$\mathbf{C}^{\mathbf{T}}$	Intermediate regression coefficient for the score matrix **T**
E	Error matrix, $\mathbf{E}=[\mathbf{e}_1\cdots\mathbf{e}_A]$
J	Jacobian, matrix of partial derivatives from a 1st order Taylor series expansion
i	Index, for the number of observation (row)
k	Index, for the number of variable (column)
j (ℓ)	Index, for the number of latent variables (ℓ applied for the inner PLS)
l	Index, for the number of iterations
A	Total number of latent variables retained in the model.
r	Rank of the matrix
μ,γ,λ	Lagrange multipliers
λ	Eigenvalue
σ	Singular value or the standard deviation
ψ	Ridge constant in Ridge Regression
ν	Scaling constant in RVPLS
s	Sign (-1 or 1).
p	Polynomial degree.

Abbreviations

AIC	Akaike's Information Criterion
ACE	Alternating Conditional Expectation
AVAS	Additivity and Variance Stabilization
BLM	Bi-Linear Modelling (PCR, PLS etc)
CART	Classification and Regression Trees
CCA	Canonical Correlation Analysis
CLS	Constrained Least Squares
CV	Cross Validation
EBPLS	Error Based Partial Least Squares
EBWU	Error Based Weight Updating
FFN	Feed Forward Network
GAM	Generalized Additive Methods
GLS	Generalized Least Squares
GN	Gauss-Newton
K	Kernel function
MARS	Multivariate Adaptive Regression Splines
MLR	Multiple Linear Regression
MSC	Multiple Signal Correction
NIR	Near Infrared
NIPALS	Nonlinear Iterative Partial Least Squares
NN	Neural Network
NP	Nonparametric
NPLS	Nested Partial Least Squares
NW	Nadaraya – Watson
OLS	Ordinary Least Squares
PCA	Principal Component Analysis
PCR	Principal Component Regression
PLS	Partial Least Squares or Projection to Latent Structures
PLSR	Partial Least Squares Regression
PPR	Projection Pursuit Regression
PRESS	Predictive Residual Error Sum of Squares
RBFN	Radial Basis Function Network
RMSE	Root Mean Square Error
RMSEC	Root Mean Square Error of Calibration
RMSECV	Root Mean Square Error of Cross Validation

RMSEP	Root Mean Square Error of Prediction
RR	Ridge Regression
RVPLS	Reciprocal Variance Partial Least Squares
SD	Steepest Descent
SDPLS	Steepest Descent Partial Least Squares
SMART	Smooth Multiple Additive Regression Technique (same as PPR)
SMLR	Stepwise Multiple Linear Regression
SIMPLS	Simple Partial Least Squares
SPLS	Spline Partial Least Squares
SVD	Singular Value Decomposition ($\mathbf{X} = \mathbf{\Gamma \Sigma P^T}$)
TLS	Total Least Squares

List of Symbols

A, … , Z	Matrices
a, … , z	Vectors
$\alpha, \beta \cdots, a, b, \cdots$	Scalars
$\hat{\alpha}_1, \hat{\boldsymbol{\beta}}, \cdots$	Estimates
$\mathbf{X}^T(\mathbf{x}')$	Matrix transpose (vector transposed used in the MATLAB language)
$\mathbf{X}^{-1}$	Matrix inverse
$\mathbf{X}^-$	Generalized matrix inverse
$\mathbf{X}^+$	Moore-Penrose matrix inverse
a_{ik}	Matrix element
$\mathbf{a}_k$	Matrix column
$\mathbf{a}_i$	Matrix row
a_i	Vector element
diag(**A**)	Diagonal elements in **A**
diag$(a, b, \cdots)$	Diagonal matrix constructed from scalars a, b, …
range(**a**, **b**, $\cdots$)	Space spanned by the vectors **a**, **b**, …
std(**a**)	Standard deviation of vector **a**
var(**a**)	Variance of vector **a**
cor(**a**, **b**)	Correlation between the vectors **a** and **b**
trace(**A**)	Trace of matrix **A** is the sum of the diagonal elements
$\|\mathbf{X}\|$	The 2-norm or Euclidean norm of **X**
$\|\mathbf{X}\|_F$	Frobenius norm $\|\mathbf{X}\|_F = \sqrt{\text{trace}(\mathbf{X}^T\mathbf{X})}$

CHAPTER 1

INTRODUCTION

1.1 Background

Although phenomenological models have significant advantages over empirical models in that they provide deeper process understanding, there is increasing demand for the empirical modelling of industrial processes in the areas of process control, process monitoring and process analysis. In pursuit of competitiveness, the need to control the process has resulted in the introduction of new instrumentation, increased use of historical data, and the need for continuous improvement of the process. Simultaneously, the products being manufactured are under constant development and new products are regularly being introduced to ensure market retention. In some areas of industry, due to the demanding and time-consuming task of constructing mechanistic models, there is a trend towards supplementing existing mechanistic models or replacing them by empirical models. In particular, for some complex processes where the development of rigorous theoretical models may not be practical, empirical data based modelling is a widely used alternative, since data based models can capture the underlying fundamental model without detailed process knowledge. However, a basic understanding of the process is still essential both to construct and validate the empirical models. Finally, the introduction of data intensive methods in the area of process analysis, e.g. on-line spectroscopy, has increased the need for multivariate statistical modelling.

Multivariate statistical regression techniques provide one family of tools for the empirical modelling of manufacturing processes. The key to the success of empirical modelling is to obtain a model that describes the underlying behaviour of the process sufficiently well to be fit for purpose. Recent developments in process instrumentation has resulted in situations where models based on least squares methodologies can lead to singular solutions and imprecise parameter estimation, due to there being more variables than observations or the presence of multicollinearity between the variables. These limitations can be overcome by applying multivariate statistical projection based techniques such as Principal Component Regression (PCR) or Partial Least Squares Regression (PLSR). These two techniques can handle both underdetermined (fewer observations than variables) data sets and collinearity amongst the variables, by capturing the underlying structure in the data in terms of a limited number of principal components or latent variables, which are linear combinations of the original variables.

The models are then constructed from the orthogonal latent variables using ordinary least squares, since both the dimensionality and multicollinearity problems have been addressed.

In PCR the latent variable model is developed based on Principal Component Analysis (PCA), such that each set of latent variables captures the maximum amount of predictor variance. In PLSR, the latent variables are calculated as a compromise between the variance explained by the predictor and the response variables. In doing so, each pair of latent variables simultaneously models the predictor and response variable space, thereby a common latent variable space is found that is less affected by noise.

The industrial necessity for empirical modelling requires new methodologies to be developed that are capable of being applied across a wide range of process plants and products. Although most industrial applications can be solved using linear regression models, the presence of nonlinearity in chemical processes has resulted in the development of nonlinear regression methods. In particular, there is a need for a universal method that can handle both the dimensionality and the multicollinearity problem and that is capable of fitting any nonlinear structure that may occur in practice. For example, there has been an increase in the use of spectroscopic measurements in industry. The resulting spectral data typically comprises a large number of variables that are collinear. Furthermore, it can exhibit nonlinear behaviour as a result of:

(i) Violations of the Beer-Lambert law.
(ii) Detector nonlinearities.
(iii) Stray light.
(iv) Nonlinearities in diffuse reflectance/transmittance.
(v) Chemically based nonlinearities.
(vi) Nonlinearities in the property/concentration relationship.

One nonlinear method that has been used with some success in industry, is the combination of PCA and Neural Networks (NN) (Kurtanjek (1995), De Vena *et al.* (1995), Turner *et al.* (1996) and Lennox *et al.* (2001)). However, if the response variance does not correspond to the major latent variables found using PCA, then the method will not be satisfactory. An alternative approach is the extension of the linear Partial Least Squares algorithm to nonlinear PLS.

The first general nonlinear PLS algorithm, reported by Wold *et al.* (1989), used Steepest Descent (SD) optimisation to update the projection parameters of the PLS model. Although a

quadratic polynomial expansion defined the nonlinear function that was introduced between each pair of latent variables, the updating procedure was independent of the choice of the nonlinear mapping. The algorithm, renamed Steepest Descent PLS (SDPLS) was investigated by Baffi *et al.* (1999a) and the steepest descent optimisation was replaced by Gauss-Newton (GN). This Error Based Weight Updating (EBWU) procedure was first reported using a quadratic polynomial expansion (Baffi *et al.*, 1999a) and later using a Radial Basis Function Network (RBFN) or a sigmoid function as the inner nonlinear mapping (Baffi *et al.*, 1999b). Once more, the method was not constrained in terms of the form of the nonlinear mapping. This algorithm is denoted Error Based PLS (EBPLS).

A limitation of the SDPLS algorithm is that it does not use any covariance information in constructing the weight updating vector. Thus, SDPLS is subjected to convergence problems associated with local minima, but is less likely to result in overfitting. For the EBPLS algorithm, that uses the covariance information, the problem is the opposite. Consequently, overfitting is likely for underdetermined, multicollinear or noisy data sets but convergence is generally achieved.

In this Thesis a new development is proposed, which combines these two methods, Nested PLS (NPLS). The NPLS algorithm resolves the same optimization problem as defined in SDPLS and EBPLS, by incorporating a separate inner PLS algorithm where the number of latent variables is determined using cross validation. The advantage of this approach is that the multicollinear problem of the EBWU procedure is removed, whilst the convergence problem of the steepest descent method is addressed. The multicollinear problems arise as a consequence of the matrix inverse being used in the EBWU procedure, whilst the convergence problem is associated with local minima. In particular, the steepest decent method is a local minimization algorithm, with no mechanism that allows it to escape the influence of a local minimum (Morris, 1993). SDPLS, EBPLS and NPLS are subject to the typical problems associated with nonlinear optimisation and the difference in performance between the procedures is associated with how the different optimisation challenges are handled, in particular the multicollinear problem, the termination criteria and the problem of local minima.

The second general nonlinear PLS algorithm was described by Wold *et al.* (1992). In this algorithm, the covariance criterion of linear Partial Least Squares was generalized to include the nonlinear case. The algorithm was termed Spline PLS (SPLS), since a spline function is used to capture the process nonlinearity. Again, the updating procedure is independent of the choice of the nonlinear mapping. This method is similar in concept to linear PLS and is not directly an

optimisation method. Based on the framework of Spline PLS, but focusing on the problem of identifying the nonlinear function between each pair of latent variables, a new criterion is proposed. It is termed the reciprocal error variance criterion and originates from the idea of the weighted average. As for the covariance criterion, it estimates each weight by independently assessing the capability of the actual predictor variable, thus eliminating the problem of dimensionality and collinearity. It gives greater weight to the more important variables. Since the Reciprocal Variance PLS (RVPLS) approach focuses on explaining the response variance only, it identifies the inner mapping better than when the covariance criterion is used, thereby reducing the error of the nonlinear mapping. The improvement in the performance of RVPLS over SPLS depends on the data set, however generally the performance increases with increased levels of multicollinearity of the predictor matrix.

Fitting a nonlinear function increases the risk of overfitting, and in particular the final model will be less parsimonious than any linear model if it involves several latent variables. Whereas the linear model of one response variable can always be represented by a single linear combination, the regression coefficient vector, this may not be the case for the nonlinear extension of PLS if the number of latent variables is larger than unity. The aim of any nonlinear PLS algorithm is therefore to minimize the number of latent variables, thus decreasing the number of nonlinear functions fitted. Consider a pair of latent variables, the latent variables in the pair being linear combinations of the predictor and response variables, respectively. The more closely the pair of latent variables capture the true underlying nonlinearity in the data, the less the estimated inner mapping will be subject to a fitting error. Since this error will be present in the residual responses, it will influence the models of subsequent pairs of latent variables.

In the optimisation approach that defines the first nonlinear PLS framework (Wold *et al.*, 1989), the error from fitting a nonlinear function is reduced by selecting this error as the objective function to be minimized. An iterative procedure is used to minimize the objective function and the aim is to obtain a pair of latent variables that describe the underlying structure with as small an error as possible, through the nonlinear function fitted between the pair of latent variables.

In the second approach that includes SPLS and RVPLS, the focus is not on minimizing the error of the inner mapping as in the optimization approach, but is based on estimating the linear combinations by assigning each variable a weight by independently calculating the capability of the actual predictor variable. For SPLS, this capability is estimated from the covariance between the response and the given predictor variable, whilst for RVPLS the capability is defined from the inverse of the variance of the residual of a model between the given predictor variable and

the response. Thus this approach does not use the covariance structure when constructing the weights and generally requires a larger number of latent variables. Consequently, the underlying structure may not be accurately identified when the covariance structure between the predictor variables is complex, resulting in greater difficulties in terms of estimating the inner mapping. Whilst SPLS gives the same significance to both the response and predictor variables when constructing each pair of latent variable, RVPLS models the response variance only. As a result the RVPLS algorithm identifies the underlying structure more accurately than the SPLS method and the nonlinear function becomes easier to identify due to a higher signal to noise ratio.

1.2 Objectives of the Thesis

The main objective of this Thesis was to address issues concerning the existing nonlinear PLS methodologies and to propose enhancements. The two central issues were the estimation of the weights in nonlinear PLS and the use of nonlinear functions to model the inner relationship of the latent variable space. The methodology for constructing the weight vectors of the latent variables will strongly influence the final model. Methods for estimating univariate nonlinear mapping are well known and will not influence the performance of the nonlinear PLS methods as much as methods for identifying an appropriate weight vector. Thus, the main focus of the Thesis is on the estimation of the weight vectors. In particular the weight updating approaches of Wold *et al.* (1989) and Baffi *et al.* (1999a,b) and the weighting methodology reported by Wold (1992) are studied, and two new algorithms are proposed.

1.3 Contributions of the Thesis

The main contributions of the Thesis are in the area of nonlinear PLS. In particular, the motivation comes from how to find the best projection of the predictor variables, as defined by the weight vector. Based on the issues identified from the existing methods, two new procedures for estimating the weight vector in nonlinear PLS are proposed. In addition, the effect of introducing a nonlinear function in the latent variable space was investigated. Different methods for selecting the nonlinear mapping of the inner space are investigated. These were both parametric methods including polynomial least squares, and nonparametric methods such as smoothing spline and kernel regression. Finally, a comparison between different nonlinear PLS approaches is presented where the issues concerning the different methodologies are discussed.

1.3.1 Chapter 2

This chapter describes the theoretical background to the Thesis, and is based on what has previously been published in the literature. One exception is the calculation of the analytical derivative for local linear kernel regression, since this is required for the optimisation updating procedure used in the first nonlinear PLS framework (SDPLS, EBPLS and NPLS). A second contribution is the comparison of different nonlinear univariate mapping functions. Although the literature survey is predominately undertaken in this chapter, subsequent chapters provide their own introductions along with appropriate references.

1.3.2 Chapter 3

Chapter 3 contains the main contributions of the Thesis, in particular the introduction of two new methods for determining the weight vector in nonlinear PLS.

The first contribution, Nested Partial Least Squares, was developed based on the work of Wold *et al.* (1989) and Baffi *et al.* (1999a,b). The Nested PLS method captures the best features of the two methods with both being special cases. It is believed that the Nested PLS algorithm represents an important contribution in the area of nonlinear PLS since it demonstrates good performance when applied to a wide range of regression problems, Chapters 4 and 5. It addresses the multicollinearity problem of the Error Based PLS algorithm of Baffi *et al.* (1999a) and reduces the problem of local minima experienced when using Steepest Descent PLS (Wold *et al.* 1989). Furthermore, the following issues are investigated; the starting vector, the termination criterion, and the effect of introducing an error as a result of the nonlinear function fitted in the inner relationship.

The second contribution, Reciprocal Variance Partial Least Squares (RVPLS) was developed from Spline PLS (SPLS) (Wold, 1992). This algorithm focuses on finding an inner mapping whereby the number of latent variables required for constructing the model is reduced, compared to SPLS. This methodology does not have the performance potential of Nested PLS as the covariance information is not used in the construction of the latent variables, but could be an appropriate choice when modelling multicollinear data sets where a simple relationship exists with the predictor matrix. The idea of using the reciprocal variance within a PLS framework, including both the linear and nonlinear case, is novel and may find specific industrial applications. Moreover, RVPLS has the potential to be used for acquiring an improved starting vector for Nested PLS.

1.3.3 Chapter 4

The contribution of Chapter 4 is a comparison between the different methods. Three data sets from different industrial applications form the basis of the study. These are two underdetermined data sets based on spectroscopy measurements and one overdetermined data set formed from process variables. The algorithms are compared in detail for these data sets. In particular, the effect of different starting and stopping criteria are investigated for the optimization based approaches. Finally, the impact of group size in cross validation when applied in the inner PLS loop of the NPLS algorithm is examined. In addition, the prediction results from seven additional data sets, included in Appendix A1, are discussed

1.3.4 Chapter 5

The contribution of Chapter 5 arises from the summary of the performance of ten independent data sets including the three data sets investigated in Chapter 4. The performance is primarily quantified in terms of as the Root Mean Squared Error of Prediction (RMSEP). The various data sets demonstrate different issues regarding the performance of the algorithms. From these data sets it is possible to draw a number of conclusions about the general performance of the methods.

1.4 Layout of the Thesis

A general introduction has been presented in Chapter 1. In Chapter 2, the scientific areas of interest are presented, and a general overview of a number of regression methods is provided. The Chapter is divided into two parts, linear modelling and nonlinear modelling.

(i) For the linear modelling section, multiple linear regression, principal component regression and partial least squares (PLS) are described

(ii) For the nonlinear modelling section, a number of the more commonly applied nonlinear methods are described. In particular, three nonlinear PLS algorithms are fully described. These are Steepest Descent PLS (Wold *et al.*, 1989), Error Based PLS (Baffi *et al.*, 1999a,b) and Spline PLS (Wold, 1992).

In addition, different univariate nonlinear regression methods are briefly discussed. Finally, the impact on the algorithm resulting from deviations between the inner mapping and the true underlying function is considered.

In Chapter 3, the two main contributions of this thesis are presented, i.e. Nested PLS and Reciprocal Variance PLS. They are compared theoretically with the algorithms on which they are formulated, and a number of important issues are discussed.

The different nonlinear PLS approaches are then compared in detail in Chapter 4, using three different data sets. The data sets comprised both underdetermined and overdetermined data sets, with varying levels of correlation, collinearity and noise. A less detailed comparison of seven supplementary data sets is included in Appendix A1. The ten data sets are mainly drawn from industrial applications, but two simulations are also included to highlight different properties of the algorithms.

In Chapter 5 the different methods are discussed and based on the discussion and the results from Chapter 4, conclusions about the performance of the different algorithms are drawn. Furthermore, suggestions for further work are made based on the ideas and work undertaken during the course of the Thesis.

CHAPTER 2

THEORETICAL BACKGROUND

2.1 Introduction

This chapter is divided into three parts. First, the fundamental basis of regression is discussed, focusing specifically on the linear regression techniques of multiple linear regression (MLR), ridge regression (RR), principal component regression (PCR) and partial least squares (PLS). The latter three methods handle the problem of highly correlated or collinear data. These methods are then extended to their nonlinear counterparts, nonlinear least squares, nonlinear PCR and nonlinear PLS. Finally, the modelling of univariate nonlinear relationships is briefly summarised.

2.2 Multivariate Multiple Linear Modelling

Multivariate multiple linear modelling is a statistical methodology for the prediction of values of one or more response variables from a number of predictor variables. It may also be used for assessing the effect of the predictor variables on the responses. In this chapter, multiple linear modelling of a single response is first discussed. This model is then generalized to handle the prediction of several dependent variables. A vast literature exists on the subject, for example Mardia *et al.* (1997) and Johnson and Wichern (1998). Attention in this chapter focuses on the underlying assumptions and their implications in the context of multicollinear problems, in addition to presenting an alternative reduced rank formulation of multivariate linear models, and its applicability in a number of situations. These scenarios include the impact of different ratios between the number of variables and observations (dimensionality), the amount of multicollinearity, the level of measurement noise, and the presence of unknown variables and noise distributions.

Initially, attention is restricted to the standard multiple regression case, where m observations are recorded for a response variable $\mathbf{y}$ and n predictor variables, $\mathbf{X}$. The aim is to model the dependence of $\mathbf{y}$ on $\mathbf{X}=[\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$. The objective for developing a model is varied. Reasons include the need to learn more about the process that determines $\mathbf{y}$, to assess the relative contribution of each of the predictors in explaining $\mathbf{y}$, or to infer future values of $\mathbf{y}$ using a new, previously unseen set of predictor variables.

2.2.1 Least Squares Regression Estimation

In this section, the modelling of the relationship between a single response variable **y** and a single set of *n* predictor variables $\mathbf{X}=[\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$ is considered. Assuming that a linear relationship exists between the response variable **y** (*m* x *1*) and the predictor variables **X** (*m* x *n*), interest is in deriving a vector $\boldsymbol{\beta}$ (*n* x *1*) through least squares estimation:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{e}, \tag{2.1}$$

where **e** is a (*m* x *1*) vector of deviations (errors) of the observations from the model, measured in the direction of the **y**-axis. The objective of least squares regression is to determine $\boldsymbol{\beta}$ such that some norm of the residual vector, **e**, is minimized. The use of the Euclidean norm to calculate $\boldsymbol{\beta}$ is called ordinary least squares (OLS). Alternatively, the use of $\boldsymbol{\beta}$, obtained through the minimisation of the *distances* of the observed values of **y** from $\mathbf{X}\boldsymbol{\beta}$, is termed Total Least Squares (TLS) (Van Huffel and Vandervalle, 1991).

Consider the situation where the data matrix, **X**, is *nonsingular*, i.e. of full rank:

(i) If $m = n$, then a general unique solution exists where $\boldsymbol{\beta} = \mathbf{X}^{-1}\mathbf{y}$.

(ii) If $m < n$, i.e. there are fewer equations than unknowns, the problem is *underdetermined* and hence more than one solution exists that satisfies $\mathbf{y} = \mathbf{X}\boldsymbol{\beta}$.

(iii) If $m > n$, i.e. there are more equations than unknowns, the problem is *overdetermined* and generally no exact solution to $\mathbf{y} = \mathbf{X}\boldsymbol{\beta}$ exists.

When using noise contaminated data to fit a linear model it is assumed in least squares regression that the system is overdetermined. Consequently, OLS minimizes the Euclidean norm of each regression model for an overdetermined system. The estimator becomes the regression vector $\hat{\boldsymbol{\beta}}$ that minimizes the inner product:

$$(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}})^{\mathrm{T}}(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}) = \sum_{i=1}^{m}(y_i - \mathbf{x}_i^{\mathrm{T}}\hat{\boldsymbol{\beta}})^2 \tag{2.2}$$

By differentiating with respect to $\boldsymbol{\beta}$, the minimum of the square of this norm occurs at values of $\hat{\boldsymbol{\beta}}$ that satisfy the square system (the normal equations):

$$\mathbf{X}^{\mathrm{T}}\mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{X}^{\mathrm{T}}\mathbf{y} \tag{2.3}$$

whose solution is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y} \tag{2.4}$$

From Equation (2.2) that is visualized in Figure 2.1, the response **y** is projected onto range(**X**), the space defined by all possible linear combinations of the **X** variables. The (n x n) system of equations, known as the normal equations, is *nonsingular* if and only if **X** has full rank. Consequently the solution, $\hat{\boldsymbol{\beta}}$, is unique if and only if **X** has full rank, i.e. the variables in **X** are linearly independent. From the normal equation it is observed that the residual vector $\mathbf{e} = \mathbf{y} - \hat{\mathbf{y}}$ is orthogonal to each column in **X**, i.e. $\mathbf{X}^{\mathrm{T}}(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}) = 0$, thus **e** lies in the space orthogonal to the range of **X**, denoted by $range(\mathbf{X})^{\perp}$.

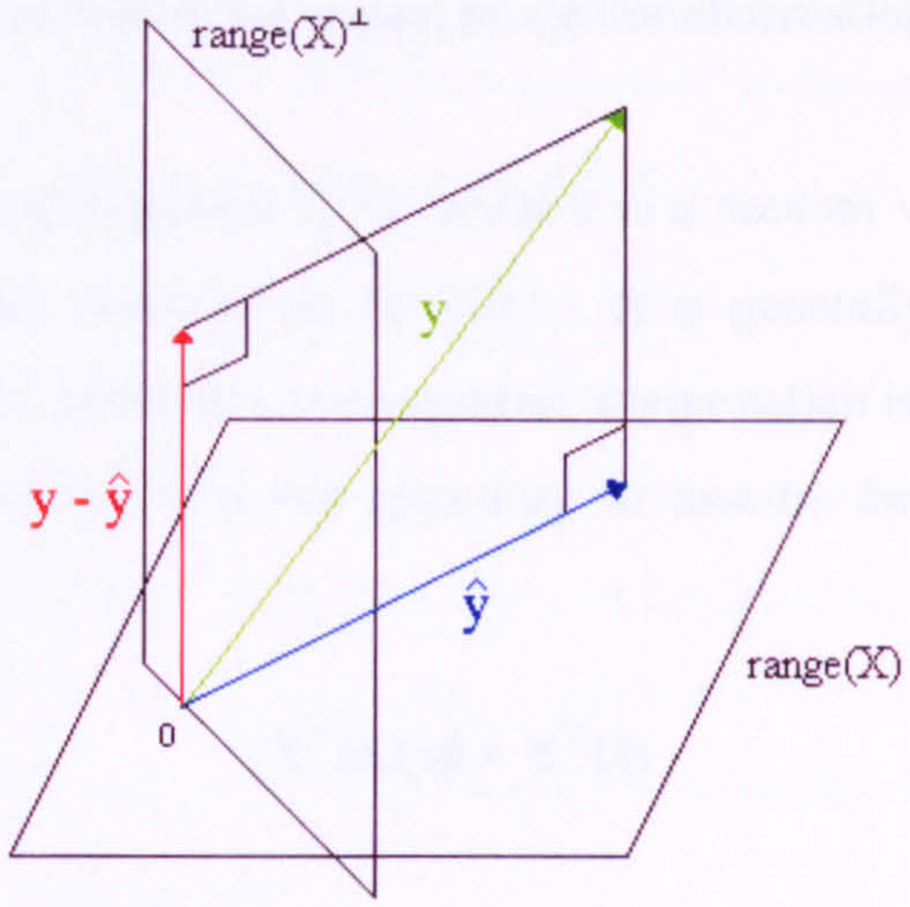

Figure 2.1. Formulation of the least squares problem in terms of orthogonal projections.

Furthermore, if **X** is not of full rank, that is, if **X** has rank $r < n$, the least squares solution is not unique. Bearing in mind that OLS assumes that the rank $r < m$, where m is the number of observations, a general inverse may be found using the Moore-Penrose inverse. This is normally achieved through Gram-Schmidt orthogonalization or singular value decomposition (SVD), and results in **X** being reduced to a matrix of full rank. The resulting full rank matrix is an orthogonal basis for **X**, i.e. it is a matrix of latent variables that explain the variance in **X**.

It is interesting to note that all linear latent variable orthogonalization methods discussed subsequently results in **X** being reduced to a matrix of full rank, but use different criteria. The fact that a matrix is not of full rank confirm that correlation exists between the **X** variables and thus the data set is said to be *multicollinear*. A low rank is often an indication of high level of collinearity (($\|\mathbf{X}\| / \|\mathbf{X}\|_F$)$^2 > 0.5$), but high level of collinearity can exist in a full rank matrix due to noise in the **X** matrix.

2.2.1.1 Weighted Least Squares

In deriving the model $\mathbf{y} \approx \mathbf{X}\boldsymbol{\beta}$ it may be desirable to weight the observations, i.e. instead of minimizing Equation (2.2), minimize:

$$\sum_{i=1}^{m} \omega_i (y_i - \mathbf{x}_i^T \boldsymbol{\beta})^2, \tag{2.5}$$

where ω_i represents a nonnegative weight that is applied to the i^{th} observation. The objective of the weighting function is to control the impact of a given observation on the overall fit.

Given a model of the form of Equation (2.1), where **e** is a random variable, such that e_i has variance σ_i^2, an appropriate value of ω_i is $1/\sigma_i^2$. It is generally assumed that e_i is an *independent* random variable, however a more general interpretation is given that enables wider interpretation of the model, i.e. it is not necessary to assume *independence*. The normal equations can be written as:

$$(\mathbf{X}^T \boldsymbol{\Omega} \mathbf{X})\hat{\boldsymbol{\beta}} = \mathbf{X}^T \boldsymbol{\Omega} \mathbf{y} \tag{2.6}$$

where $\boldsymbol{\Omega} = \text{diag}(\omega_1 \cdots \omega_m)$. The weight matrix, $\boldsymbol{\Omega}$, can be generalized to the case where the weight matrix is not diagonal. Use of a nondiagonal $\boldsymbol{\Omega}$ is sometimes referred to as generalized least squares (GLS), with the weight matrix being symmetric and positive definite, i.e. it does not assume independence. The weighted least squares estimator is thus given by:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T \boldsymbol{\Omega} \mathbf{X})^{-1} \mathbf{X}^T \boldsymbol{\Omega} \mathbf{y} \tag{2.7}$$

In a model $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{E}$ where **E** is a dependent random variable with variance-covariance matrix **S**, the choice of $\boldsymbol{\Omega}$ as $\mathbf{S}^{-1}$ yield estimators with certain desirable statistical properties, i.e. the observations are given weights according to the covariance structure, thereby taking into account the dependency in the data.

Selection of the weight matrix should not be undertaken without understanding the assumptions for the choice. For example, for GLS it is assumed that the errors are normally distributed.

2.2.1.2 Ridge Regression

Ridge Regression (Hoerl, 1962) is a modelling technique for dealing with ill-conditioned data, i.e. small changes in the input data result in significant changes in the parameters estimated (Noble, Chapter 8, 1969). Ill-conditioned data is often a consequence of high level of collinearity. Other sources of ill-conditioning may be more subtle, e.g. large variations in the leverage of observations. This may materialise either naturally due to the nature of the data or by outliers caused by faults when sampling the data. Another issue is, if an important variable is not measured the estimators will be biased. To address ill-conditioning and possible bias, Ridge Regression (RR) was introduced. The ridge parameter is defined in terms of a diagonal constant matrix, $\psi \mathbf{I}$, which is added to the covariance matrix:

$$\hat{\boldsymbol{\beta}}_r = (\mathbf{X}^T\mathbf{X} + \psi \mathbf{I})^{-1}\mathbf{X}^T\mathbf{y} \tag{2.8}$$

Adding the constant diagonal matrix, $\psi \mathbf{I}$, has the effect of stabilizing and shrinking the coefficient $\hat{\boldsymbol{\beta}}_r$ (Hoerl and Kennard, 1970). Hoerl and Kennard (1970) showed that inclusion of the ridge parameter leads to a reduction in the prediction error at the cost of biasing the coefficients. The relationship between the ridge regression estimate $\hat{\boldsymbol{\beta}}_r$ and the OLS estimate, $\hat{\boldsymbol{\beta}}$, is:

$$\hat{\boldsymbol{\beta}}_r = ((\mathbf{X}^T\mathbf{X})^{-1}\psi + \mathbf{I})^{-1}\hat{\boldsymbol{\beta}} \tag{2.9}$$

Draper and Smith (1998) showed that the regression vector calculated from ridge regression is the least squares solution subject to the constraint that the vector is confined to a sphere centred around the origin. Thus, one way of looking at ridge regression is that it assumes that the regression coefficients are more likely to be small, i.e. close to zero. The key issue in ridge regression is to determine the optimum value of ψ. One approach was described by Hoerl *et al.* (1975). They argued that $\psi^* = os^2/(\hat{\boldsymbol{\beta}}^T\hat{\boldsymbol{\beta}})$ is a reasonable choice, where o is the number of parameters in the model (not including β_0), s^2, is the residual mean square of the analysis of variance from OLS, and $\hat{\boldsymbol{\beta}}$ is the OLS estimate. Other possibilities are to plot the regression coefficient against ψ and select the value of ψ to be where the parameters have stabilized (Hoerl and Kennard 1970), or to use cross-validation (Section 3.2.5.1).

2.2.2 Multivariate Multiple Least Squares

The problem of modelling the relationship between p responses $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_p]$ and a single set of n predictor variables $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$ is now considered. Again, a linear relationship between the response variables $\mathbf{Y}$ (m x p) and the predictor variables $\mathbf{X}$ (m x n) is

assumed. In least squares, each response is assumed to have its own regression model. Thus, interest is in a matrix $\mathbf{Y} \approx \mathbf{XB}$, where $\mathbf{B} = [\beta_1, \beta_2, \cdots, \beta_p]$. The regression model may be written as:

$$\mathbf{Y} = \mathbf{XB} + \mathbf{E}, \tag{2.10}$$

where $\mathbf{E}$ is a matrix of deviations (errors) of the observations from the functional model. Analogous to Equation (2.4) the solution becomes:

$$\hat{\mathbf{B}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \tag{2.11}$$

It should be noted that each response variable $\mathbf{y}_j$, where $j = 1, 2, \cdots, p$ is defined as for multiple linear regression, Equation (2.1), thus the regression vectors in the regression matrix, $\hat{\mathbf{B}}$, are mutually independent.

2.2.3 Principal Component Regression (PCR)

In this section, the concept of Principal Component Regression (PCR) is described. PCR is widely used and is described in textbooks such as Mardia *et al.* (1979), Bishop (1995), and Næs, *et al.* (2002). Examples of the use of PCR in industry can be found in Mejdel and Skogstad (1991) and Wise *et al.* (1995).

PCR is the application of ordinary least squares regression of $\mathbf{Y}$ on selected orthogonal latent variables of $\mathbf{X}$. In PCR the latent variables are called principal components and they describe the underlying variance of $\mathbf{X}$. If all principal components are selected (full rank), the Moore-Penrose inverse is employed, and the least squares solution is obtained. The idea of PCR is to use reduced rank regression, i.e. select the first A principal components that contain the most information relating to $\mathbf{Y}$, excluding the remaining principal components that are primarily associated with noise. The principal components to be included in the regression model can be selected based on variable selection (Martens and Næs, 1989). Frequently those components with the largest variance best describe the response variables, if not the method may produce inadequate models due to a low signal to noise ratio associated with lower order principal components.

The orthogonal principal components or score vectors $\mathbf{T} = \{\mathbf{t}_1, \cdots, \mathbf{t}_A\}$ are found by applying Principal Component Analysis (PCA) to the original data matrix $\mathbf{X}$. Hotelling (1933) developed this technique, after the original derivation was proposed by Karl Pearson (1901). PCA is a

method for reducing the dimensionality of a multivariate system. The aim is to summarise the information in terms of a few latent variables, which are standardized linear combinations of the original variables. The objective is to find the latent variables of **X**, which are defined in decreasing order of variance explained. Thus the method has been described as a "parsimonious summarisation" (Mardia *et al.*, 1979). PCR has a number of similarities to ridge regression, but while PCR normally deletes the influence of the smaller eigenvectors, ridge regression only decreases the influence of them. In PCA, the j^{th} orthonormal loading vectors $\mathbf{p}_j$ are selected such that the inner products:

$$\mathbf{t}_j^T \mathbf{t}_j = (\mathbf{X}\mathbf{p}_j)^T (\mathbf{X}\mathbf{p}_j) \tag{2.12}$$

are maximised. This is an eigenvalue problem. Rearranging Equation (2.12) gives:

$$\mathbf{p}_j^T \mathbf{X}^T \mathbf{X} \mathbf{p}_j = \mathbf{t}_j^T \mathbf{t}_j \text{ or } \mathbf{X}^T \mathbf{X} \mathbf{p}_j = \lambda_j \mathbf{p}_j \text{ , where } \lambda_j = \mathbf{t}_j^T \mathbf{t}_j \tag{2.13}$$

Likewise, the score vectors $\mathbf{t}_j$ represent the j^{th} eigenvector of $\mathbf{X}\mathbf{X}^T$, scaled to length $\sqrt{\lambda_j}$. If all r eigenvectors are extracted, i.e. the rank of **X** is r, **T** becomes an orthogonal basis for **X**, and **X** can be written as $\mathbf{X} = \mathbf{T}\mathbf{P}^T$:

$$\mathbf{X} = \mathbf{T}\mathbf{P}^T = \mathbf{t}_1 \mathbf{p}_1^T + \cdots + \mathbf{t}_r \mathbf{p}_r^T \tag{2.14}$$

Equation (2.14), represents the decomposition of **X** into rank-one matrices such that each outer product ($\mathbf{t}_j \mathbf{p}_j^T$) captures as much of the underlying variance of **X** as possible. This statement holds both where the variance is defined either as the Euclidean or the Frobenius matrix norm. This is in fact the singular value decomposition of **X**:

$$\mathbf{X} = \mathbf{\Gamma}\mathbf{\Sigma}\mathbf{P}^T, \text{ where } \mathbf{\Sigma} = diag(\sqrt{\lambda_j}) \text{ and } j \in \{1,2,\cdots,r\} \tag{2.15}$$

where $\mathbf{\Gamma} = \mathbf{T}/\text{norm}(\mathbf{T})$ is the left singular matrix of **X** or the eigenvectors of $\mathbf{X}\mathbf{X}^T$. If only the first few principal components are retained, i.e. $A < r$, the data matrix **X** is approximated by $\mathbf{T}\mathbf{P}^T$:

$$\mathbf{X} = \mathbf{T}\mathbf{P}^T + \mathbf{E} = \mathbf{\Gamma}\mathbf{\Sigma}\mathbf{P}^T + \mathbf{E} \tag{2.16}$$

For any A where $0 \le A \le r$, the matrix, **E**, of Equation (2.16) also satisfies:

$$\| \mathbf{X} - \mathbf{T}\mathbf{P}^T \|_2 = \| \mathbf{E} \|_2 = \sqrt{\lambda_{A+1}} \tag{2.17}$$

for the Euclidean norm, or for the Frobenius norm:

$$\| \mathbf{X} - \mathbf{T}\mathbf{P}^T \|_F = \| \mathbf{E} \|_F = \sqrt{\lambda_{A+1} + \cdots + \lambda_r} \tag{2.18}$$

2.2.3.1 Geometric Interpretation of PCA

Consider the data points defined by a hyperellipsoid, an ellipsoid in multidimensional space. A hyperellipsoid can be approximated in terms of a line segment by the longest axis, Figure 2.2. A two-dimensional ellipsoid can be approximated in terms of the longest and the second-longest axes. Continuing in this fashion, at each step the approximation is improved by adding to the approximation, the largest axis of the hyperellipsoid not yet included. After r steps, all the variance of $\mathbf{X}$ is captured. The axes of the hyperellipsoid represent the principal components. The first principal component, $\mathbf{t}_1$, is obtained from the projection of the data onto the first (and largest) principal axis. The second principal component, $\mathbf{t}_2$, is obtained from the projection of the data onto the second largest principal axis, which is orthogonal to the first. Continuing in this manner, the j^{th} principal component, $\mathbf{t}_j$, is obtained from the projection of the data onto the j^{th} largest principal axis, which is orthogonal to the first $j-1$ axes.

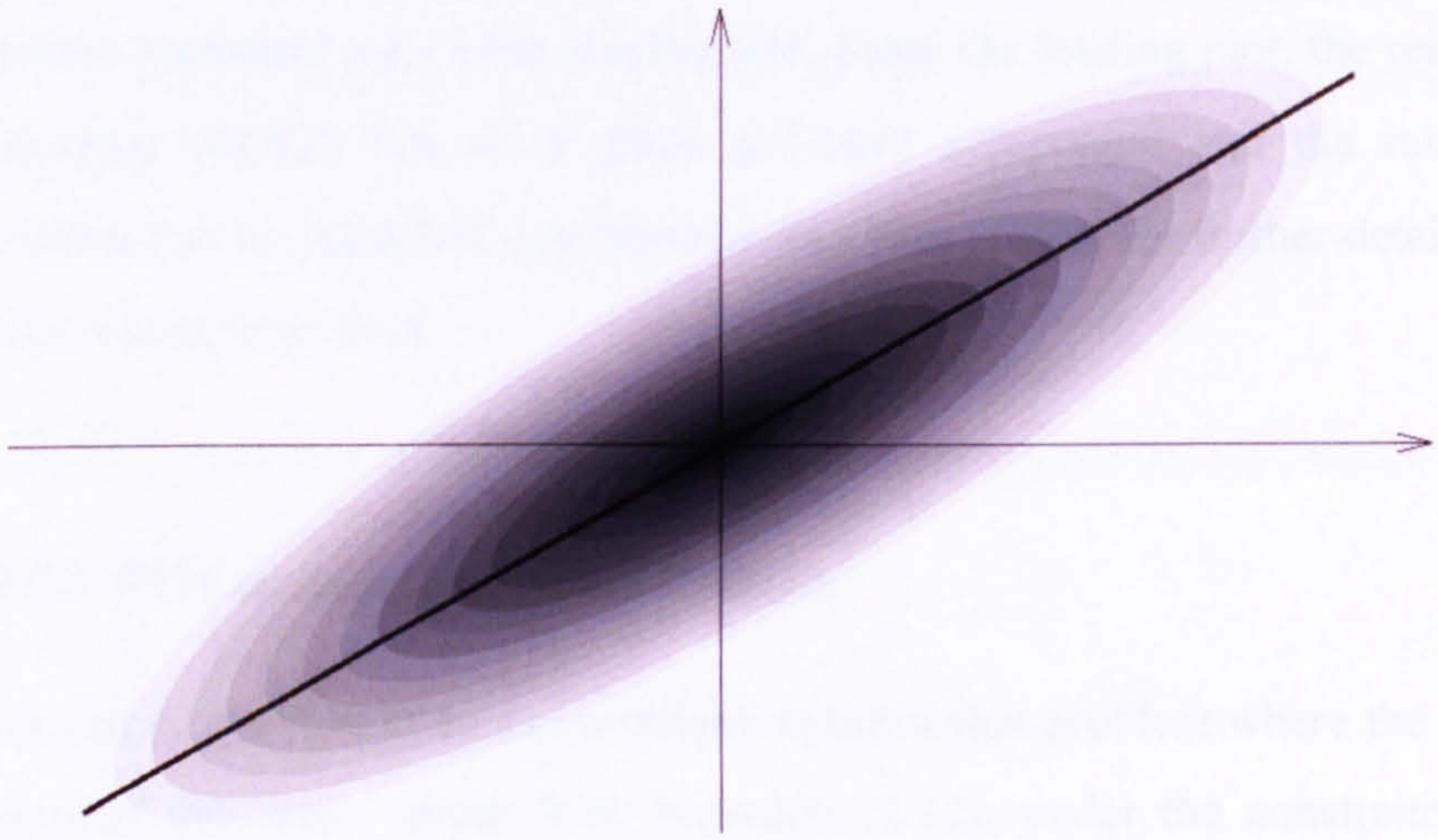

Figure 2.2. The first principal component of an elliptical distribution in the plane.

An example of the geometric interpretation of PCA for a multicollinear (*100* x *3*) data matrix, $\mathbf{X}$, is shown in Figure 2.3.

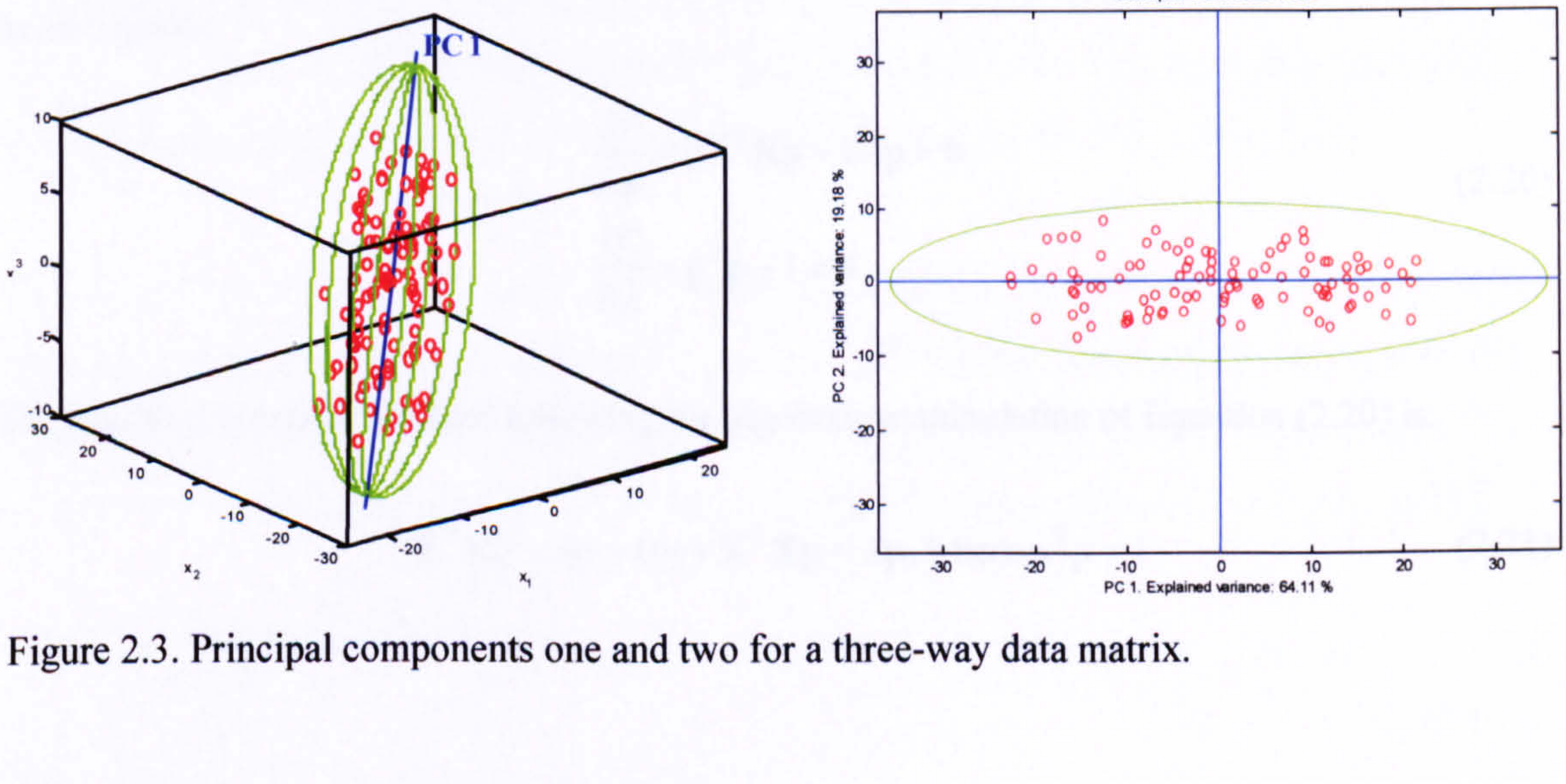

Figure 2.3. Principal components one and two for a three-way data matrix.

The three-dimensional data is shown on the left in Figure 2.3, with the first principal axis (PC1) sketched. On the right, the score plot for the first and second principal component is given. This score plot reduces the dimensionality from three to two, thereby simplifying the interpretation of the underlying structure in the data, using the maximum variance of the original data. From the score plot it is possible to detect outliers, nonlinearities, clusters, and it has been used for example in process monitoring by including limits (Wise *et al.* 1990, MacGregor *et al.* 1991).

Other plots associated with PCA include the Scree plot (Cattell, 1966) where the relative eigenvalue is plotted versus the corresponding principal component number. The Scree plot displays the importance of each principal component, based on the relative variance explained. For this example (Figure 2.3) the three individual principal components capture 64.1%, 19.2%, and 16.7% of the variance of the **X** matrix.

Another important plot is the loading plot. From the loading plot, the relative influence of each individual variable has on a given principal component and the relationship between the variables can be identified, see Martens and Næs (1989) for further details on the interpretation of the results from PCA.

2.2.3.2 PCA as an Optimization Problem

PCA can be thought of as a constrained optimisation problem where the aim is to maximise the length of the score vector from Equation (2.12), under the constraint $\mathbf{p}^T\mathbf{p}=1$. Using the Lagrange multiplier, λ, the optimisation equation can be written as:

$$f(\mathbf{p},\lambda)=(\mathbf{t}^T\mathbf{t})-\lambda(\mathbf{p}^T\mathbf{p}-1)=\mathbf{p}^T\mathbf{X}^T\mathbf{X}\mathbf{p}-\lambda(\mathbf{p}^T\mathbf{p}-1) \tag{2.19}$$

Taking the partial derivative of Equation (2.19) with respect to $\mathbf{p}$ and λ, and setting them equal to zero gives:

$$\begin{aligned}\frac{\partial f}{\partial \mathbf{p}}&=2\mathbf{X}^T\mathbf{X}\mathbf{p}-2\lambda\mathbf{p}=\mathbf{0}\\ \frac{\partial f}{\partial \lambda}&=\mathbf{p}^T\mathbf{p}-1=\mathbf{0}\end{aligned} \tag{2.20}$$

The resulting function obtained following the algebraic manipulation of Equation (2.20) is:

$$\mathbf{X}^T\mathbf{X}\mathbf{p}-\lambda\mathbf{p}=0\leftrightarrow\mathbf{X}^T\mathbf{X}\mathbf{p}=\lambda\mathbf{p},\text{ where }\mathbf{p}^T\mathbf{p}=1 \tag{2.21}$$

This is no different to the eigenvalue problem defined in Equation (2.13), thus PCA can be defined as an optimisation problem.

In PCA, the latent variables are found based on a linear criterion. However it is possible to extend PCA to include nonlinearity, i.e. nonlinear PCA, and solve it as a constrained optimisation problem using various objective functions (Oja, 1989). Dong and McAvoy (1996), Jia *et al.*, (1998) and Shao *et al.* (1999), give recent examples of nonlinear PCA using neural networks.

2.2.3.3 The Principal Component Regression Equations

Principal component regression is obtained by regressing the response matrix, **Y** on the score matrix **T** obtained from the application of PCA to the data matrix **X**, where $\mathbf{TP}^{\mathrm{T}}$ is typically a low rank approximation ($A < r$) of **X** as in Equation (2.16). The regression coefficients **B** are defined as:

$$\mathbf{B} = \mathbf{P}\boldsymbol{\Lambda}^{-1}\mathbf{P}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{Y} \tag{2.22}$$

where $\boldsymbol{\Lambda} = \mathrm{diag}([\lambda_1, \cdots, \lambda_A])$. The regression coefficients, **B**, are obtained by first regressing **Y** on the scores **T**:

$$\mathbf{C}^{\mathrm{T}} = \left(\mathbf{T}^{\mathrm{T}}\mathbf{T}\right)^{-1}\mathbf{T}^{\mathrm{T}}\mathbf{Y} = \boldsymbol{\Lambda}^{-1}\mathbf{P}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{Y} \tag{2.23}$$

where **T** comprises A columns given by $\mathbf{T} = \mathbf{XP}$. The matrix $\mathbf{C}^{\mathrm{T}}$ is then transformed to a matrix of regression coefficients as follows:

$$\mathbf{B} = \mathbf{PC}^{\mathrm{T}} = \mathbf{P}\,\mathrm{diag}(1/[\lambda_1, \cdots, \lambda_{\mathrm{A}}])\mathbf{P}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{Y} \tag{2.24}$$

where the eigenvalues $\boldsymbol{\Lambda} = \mathrm{diag}([\lambda_1, \cdots, \lambda_{\mathrm{A}}]) = \mathbf{T}^{\mathrm{T}}\mathbf{T} = \boldsymbol{\Sigma}^2$ are the singular values squared and **P** is the loading matrix. Since the columns of **T** are orthogonal, the least squares estimators, $\mathbf{c}_j$, are unaltered if some of the columns ($\neq j$) of **T** are deleted from the regression analysis.

Furthermore, their distribution is also unaltered. Altering the model by selecting those components that are significant, e.g. Hill *et al.* (1977), is straightforward. This can be achieved by giving the other regression coefficients a value of zero. In practise any variable selection method may be used to select the most appropriate modelling space (Martens and Næs, 1989).

As for the **X** matrix, in Equation (2.14), **Y** can be written as a decomposition of rank-one matrices:

$$\begin{aligned}\mathbf{X} &= \mathbf{T}\mathbf{P}^{\mathrm{T}} + \mathbf{E} = \mathbf{t}_1\mathbf{p}_1^{\mathrm{T}} + \cdots + \mathbf{t}_A\mathbf{p}_A^{\mathrm{T}} + \mathbf{E} \\ \mathbf{Y} &= \mathbf{T}\mathbf{C}^{\mathrm{T}} + \mathbf{F} = \mathbf{t}_1\mathbf{c}_1^{\mathrm{T}} + \cdots + \mathbf{t}_A\mathbf{c}_A^{\mathrm{T}} + \mathbf{F}\end{aligned} \tag{2.25}$$

such that the model is built from the reduced low-rank approximations of the **X** matrix that capture the information that explain the majority of the underlying variability and excludes those that capture the noise.

2.2.3.4 The Nonlinear Iterative Partial Least Squares Algorithm

In this Section the Nonlinear Iterative Partial Least Squares (NIPALS) algorithm is discussed as an approach to solving the eigenvalue problem in PCA. It was originally proposed by Wold (1966). The approach belongs to the class of *power iteration* methodologies, and takes advantage of the fact that the sequence

$$\frac{\mathbf{p}}{\|\mathbf{p}\|}, \frac{(\mathbf{X}^{\mathrm{T}}\mathbf{X})\mathbf{p}}{\|\mathbf{p}\|}, \frac{(\mathbf{X}^{\mathrm{T}}\mathbf{X})^2\mathbf{p}}{\|\mathbf{p}\|}, \frac{(\mathbf{X}^{\mathrm{T}}\mathbf{X})^3\mathbf{p}}{\|\mathbf{p}\|}, \cdots \tag{2.26}$$

converges, under certain assumptions, to an eigenvector corresponding to the largest eigenvalue of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$. Although the power iteration method is well known, it can be slow to converge except where there are large differences between the largest and the second largest eigenvalues. If only the first few eigenvectors are to be retained, the algorithm is often fast, since the power iteration extracts the largest factor and since the differences between the first few eigenvalues tend in practise to be large ($\lambda_1 >> \lambda_2 >> \lambda_3 >> \cdots \geq \lambda_{p-2} \geq \lambda_{p-1} \geq \lambda_p$).

One interpretation of this observation is that the first few latent variables extracted explain a large amount of the variation of the **X** matrix, whilst the lower order latent variables contain mostly noise. Thus it is more likely that the first eigenvalues differ more in value than the latter eigenvalues. However, one should be aware of the possibility of two subsequent eigenvalues having similar values since it may prevent convergence.

Convergence problems can easily be confirmed by applying an enhanced method for solving the eigenvalue problem. Furthermore, if convergence is not achieved within a limited number of iterations, e.g. 100 iterations, it is an indication that two subsequent eigenvalues lies the same region.

Instead of calculating the covariance matrix and using the power method as above in Equation (2.26) $\mathbf{p}_{l+1} = (\mathbf{X}^T\mathbf{X})\mathbf{p}_l$, the NIPALS algorithm can be applied. A similar approach to the power iteration is followed, but in two steps:

(i) Calculate successive estimates of the score vector: $\mathbf{t}_l = \mathbf{X}\mathbf{p}_l$

(ii) Calculate an improved estimate of the loading vector: $\mathbf{p}_{l+1} = \mathbf{X}^T\mathbf{t}_l$

Consequently, the NIPALS algorithm avoids the construction of the covariance matrix, but every iteration costs more in terms of computer time. Each factor is obtained through the repeated regression of **X** on the **p** vector to obtain an enhanced score vector **t**, and then the regression of **X** on the score vector **t** to obtain an enhanced loading vector **p**. Enhanced is defined as a vector that is closer to the first principal component (eigenvector) given by Equation (2.26).

Explicitly, **p** converges to the largest eigenvector of $\mathbf{X}^T\mathbf{X}$. At the same time, **t** converges to the largest eigenvector of $\mathbf{X}\mathbf{X}^T$. Following convergence, the variance of the first eigenvector is subtracted as a rank one updating of the **X** matrix, $\mathbf{X}_{j+1} = \mathbf{X}_j - \mathbf{t}_j\mathbf{p}_j$, and the next principal component (j+1) is obtained in the same manner using the residual matrix $\mathbf{X}_{j+1}$. The basic algorithm is described in Algorithm 2.1.

```
function [T, P] = NIPALS(X, lv, limit)      % NIPALS algorithm
for j = 1 : lv,                             % Repeat until lv latent variables are calculated
   t = X(:, 1);                             % The first variable (or the one with highest variance)
   conv = limit + 1;                        % Initializing
   while conv > limit,                      % Repeat until convergence
     told = t;                              % Retain the old score vector to regulate the convergence
     p = X'*t;                              % Covariance between t and the X variables
     p = p/norm(p);                         % Normalize to unit length
     t = X*p;                               % Update t using the linear combination p
     conv = norm(t-told)/norm(t);           % Convergence if conv < limit (and no. of iterations > max iter.)
   end                                      % Inner loop of iterations
   X = X – t*p';                            % Rank one reduction of the X matrix
   T(:, j) = t;                             % Retain the scores
   P(:, j) = p;                             % Retain  the loadings
end                                         % Outer loop of iterations
```

Algorithm 2.1. The NIPALS algorithm for PCA (MATLAB code).

The NIPALS algorithm is an iterative algorithm that requires a termination criterion to determine when a satisfactory solution (eigenvector) has been obtained. It is normal to use the relative difference between the current and the previous estimated eigenvector (or score vector) per iteration as a termination criterion. If the relative change is lower than a selected value (limit typically 10^{-8}) the inner 'while' loop in Algorithm 2.1 terminates and the estimated eigenvector (**t** of $\mathbf{XX}^T$ or **p** of $\mathbf{X}^T\mathbf{X}$) is used for the rank one reduction of the **X** matrix. The next pair of eigenvectors are then calculated using the rank one reduced **X** matrix. Furthermore, it is usual to include a maximum number of iterations as a second termination criterion (typically 100) and include a warning if this number is reached.

2.2.4 Partial Least Squares

2.2.4.1 The History of Linear Partial Least Squares

Partial Least Squares or Projection to Latent Structures (PLS) originated from the work of Wold (1966a,b), and was further developed by Wold *et al.* (1983) in the early eighties. Since then, many developments and applications have been proposed, mainly in the fields of chemistry, biology, medicine and the food and process industries. Two comprehensive reviews of the development of PLS are given by Geladi (1988) and Geladi and Esbensen (1990).

During the late eighties, a number of papers were published concerning the theoretical basis of PLS. For example Manne (1987) showed that there was no need to reduce **X** (or **Y**) to undertake the PLS computations. This was an important contribution to the development of the PLS algorithm and relates PLS to the numerical eigenvalue decomposition algorithms of Lanczos (1952), Hestnes (1952) and others. The following year, Höskuldsson (1988) published a paper on the properties of the PLS algorithm, in particular showing that PLS could be defined as a Singular Value Decomposition (SVD) problem or as an eigenvalue problem. The work of Manne (1987) and Höskuldsson (1988) formed the basis of the Simple Partial Least Squares (SIMPLS) algorithm of De Jong (1993a) and De Jong (1993b) where it was shown that one response variable is explained better by PLS than PCR for the same number of components.

In 1987, Lorber *et al.* showed how the solution of the regression stage in PLS can be formulated as a pseudo inverse of the covariance matrix. Another important article was that of Helland (1988), where he related PLS regression to theoretical statistical terms and also identified directions for further research and modifications to the PLS algorithm. Moreover, Helland (1988) proposed an alternative basis for the PLS regression space (one y-variable), i.e. the space

spanned by the loading weights. Later, a comparison of PLS with the statistical techniques of Multiple Linear Regression (MLR) and Ridge Regression (RR) was undertaken by Frank and Friedman (1993). From the perspective of model building the interpretation of PLS was investigated by Kvalheim and Karstang (1989), and later the geometry of PLS was illustrated by Phatak *et al.* (1992) and developed further by Phatack and De Jong (1997).

From the work of Frank (1987), it follows that each PLS factor is a compromise between the maximum correlation explained in terms of **y** and the maximum explained variance of **X**, and that an infinite number of alternative compromises exist. The concept of considering PLS regression as a member of a wider class of methods was also described in Höskuldsson (1988) and Lorber *et al.* (1987). This resulted in the development of Continuum Regression (Stone and Brooks, 1990). In Continuum Regression, MLR, PLSR and PCR are considered as a single modelling class, where a constant α is used as a measure to define a model relationship. For MLR, PLS and PCR , α is 0, ½, 1. Continuum Regression was demonstrated to perform well for some examples (Stone and Brooks, 1990 and De Jong *et al.*, 2001), but a limitation was that cross validation was required to determine both α and the number of latent variables. Since cross validation is an issue in iterative procedures such as PLS regression, the problem is perceived as a limitation in continuum regression, i.e. the result is dependent on which method of cross validation is applied (Section 3.2.5.1).

2.2.4.2 The Theory of Partial Least Squares.

Partial Least Squares (PLS) uses the same idea as PCR, i.e. **Y** is regressed on a reduced orthogonal representation of the original variables, **T**. The difference is that the scores are calculated using a different criterion to that of PCR. In PLS, **X** and **Y** are represented by the linear combinations $\mathbf{t} = \mathbf{Xw}$ and $\mathbf{u} = \mathbf{Yq}$, using the weight vector **w** and the loading vector **q**, respectively. The objective is not to maximize $\mathbf{t}^T\mathbf{t}$, i.e. not to find the dominant direction (the largest eigenvector) in the **X** matrix as in PCA, but to maximise $\mathbf{u}^T\mathbf{t}$, the covariance between each pair of score vectors:

$$\mathbf{u}^T\mathbf{t} = (\mathbf{Yq})^T(\mathbf{Xw}), \text{ for } \|\mathbf{w}\| = \|\mathbf{q}\| = 1 \tag{2.27}$$

that is to find the dominant direction common to both the **X** and **Y** matrices. Thus, PLS can be thought of as a constrained optimisation problem where the aim is to maximise the expression in Equation (2.27), under the constraint $\mathbf{w}^T\mathbf{w} = \mathbf{q}^T\mathbf{q} = 1$.

As a result, PLS is a compromise between the approximation of the prediction variables and the prediction of the response variables. Using the Lagrange multipliers, μ and ν, the optimisation equation can be written as (Höskuldsson, 1996):

$$f(\mathbf{w},\mathbf{q},\mu,\nu)=(\mathbf{u}^{\mathrm{T}}\mathbf{t})+\mu(1-\mathbf{w}^{\mathrm{T}}\mathbf{w})+\nu(1-\mathbf{q}^{\mathrm{T}}\mathbf{q}) \tag{2.28}$$

Taking the partial derivatives of Equation (2.28) with respect to $\mathbf{w}$, $\mathbf{q}$, μ and ν, and setting the partial derivatives equal to zero and observing that $(\mathbf{u}^{\mathrm{T}}\mathbf{t})=(\mathbf{t}^{\mathrm{T}}\mathbf{u})$ gives:

$$\frac{\partial f}{\partial \mathbf{w}}=2\mathbf{X}^{\mathrm{T}}\mathbf{Y}\mathbf{q}-2\mu\mathbf{w}=\mathbf{0} \tag{2.29a}$$

$$\frac{\partial f}{\partial \mathbf{q}}=2\mathbf{Y}^{\mathrm{T}}\mathbf{X}\mathbf{w}-2\nu\mathbf{q}=\mathbf{0} \tag{2.29b}$$

$$\frac{\partial f}{\partial \mu}=1-\mathbf{w}^{\mathrm{T}}\mathbf{w}=\mathbf{0} \tag{2.29c}$$

$$\frac{\partial f}{\partial \nu}=1-\mathbf{q}^{\mathrm{T}}\mathbf{q}=\mathbf{0} \tag{2.29d}$$

From Equations (2.29a) and (2.29b):

$$\mathbf{q}^{\mathrm{T}}\mathbf{Y}^{\mathrm{T}}\mathbf{X}\mathbf{w}=\nu \quad \text{or} \quad \mathbf{w}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{Y}\mathbf{q}=\mu \tag{2.30}$$

By transposing the first expression and comparing with the second, it can be shown that the Langrange multipliers are equal, i.e. $\nu=\mu$. Equation (2.30) is a SVD problem, i.e. $\mathbf{w}$ and $\mathbf{q}$ can be found through the application of SVD to either of the two expressions in Equation (2.30). Thus, the weight matrix $\mathbf{W}=[\mathbf{w}_1,\cdots,\mathbf{w}_A]$ and the loading matrix $\mathbf{Q}=[\mathbf{q}_1,\cdots,\mathbf{q}_A]$ can be found using a reduced SVD of $\mathbf{X}^{\mathrm{T}}\mathbf{Y}$:

$$\mathbf{X}^{\mathrm{T}}\mathbf{Y}=\mathbf{W}\boldsymbol{\Sigma}\mathbf{Q}^{\mathrm{T}} \tag{2.31}$$

where $\boldsymbol{\Sigma}=diag(\sigma_1,\cdots,\sigma_A)$, i.e. the singular values. Combining the two expressions in Equation (2.30) and factoring out $\mathbf{q}$, the following eigenvalue problem is obtained:

$$\mathbf{X}^{\mathrm{T}}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{X}\mathbf{w}=\sigma^2\mathbf{w}=\lambda\mathbf{w} \tag{2.32}$$

The eigenvectors $\mathbf{W}=[\mathbf{w}_1,\cdots,\mathbf{w}_A]$ and their corresponding eigenvalues $\Lambda=\mathrm{diag}([\lambda_1\cdots\lambda_r])$, are ordered such that $\lambda_1\geq\lambda_2\geq\cdots\geq\lambda_r$, and they can be extracted from Equation (2.32) by any algorithm that solves the symmetric eigenvalue problem. As customary for Principal Component Regression (PCR) only the first A eigenvectors are retained ($A<r$). Solving Equation (2.29a) with respect to $\mathbf{w}$ and inserting it into Equation (2.29b), the resulting eigenvalue problem is obtained:

$$\mathbf{Y}^T\mathbf{X}\mathbf{X}^T\mathbf{Y}\mathbf{q} = \lambda\mathbf{q} \tag{2.33}$$

Furthermore, multiplying by **X** and **Y** in Equations (2.32) and (2.33) respectively, the following alternative expressions for the score vectors can be found (Höskuldsson, 1996):

$$\mathbf{X}\mathbf{X}^T\mathbf{Y}\mathbf{Y}^T\mathbf{t} = \lambda\mathbf{t} \tag{2.34}$$

$$\mathbf{Y}\mathbf{Y}^T\mathbf{X}\mathbf{X}^T\mathbf{u} = \lambda\mathbf{u} \tag{2.35}$$

Thus the PLS model can be obtained by solving one of the four eigenvalue problems given in Equations (2.32) – (2.35). The regression coefficient can be found by first regressing **Y** on the selected score matrix **T** as in PCR, but since the loading matrix **P** is not the eigenvectors of **X**, i.e. $\mathbf{T} \neq \mathbf{XP}$, the matrix **T** is calculated as $\mathbf{T} = \mathbf{XW}\left(\mathbf{P}^T\mathbf{W}\right)^{-1}$, see Helland (1988) or Ter Braak and De Jong (1998):

$$\begin{aligned}\mathbf{C}^T &= \left(\mathbf{T}^T\mathbf{T}\right)^{-1}\mathbf{T}^T\mathbf{Y}\\ \hat{\mathbf{Y}} &= \mathbf{T}\mathbf{C}^T = \mathbf{T}\mathbf{P}^T\mathbf{W}\left(\mathbf{P}^T\mathbf{W}\right)^{-1}\mathbf{C}^T\\ \hat{\mathbf{Y}} &= \hat{\mathbf{X}}\hat{\mathbf{B}} = \hat{\mathbf{X}}\mathbf{W}\left(\mathbf{P}^T\mathbf{W}\right)^{-1}\mathbf{C}^T\end{aligned} \tag{2.36}$$

Note that the prediction of **Y** usually is computed as $\hat{\mathbf{Y}} = \mathbf{X}\hat{\mathbf{B}} = (\hat{\mathbf{X}} + \mathbf{E})\hat{\mathbf{B}}$ where $\mathbf{E} = \mathbf{X} - \mathbf{T}\mathbf{P}^T$. However, $\mathbf{E}\hat{\mathbf{B}}$ will be insignificant since **W** is orthogonal basis for $\hat{\mathbf{B}}$. Again, the same expression as for PCR, Equation (2.25), is obtained:

$$\begin{aligned}\mathbf{X} &= \mathbf{T}\mathbf{P}^T + \mathbf{E} = \mathbf{t}_1\mathbf{p}_1^T + \cdots + \mathbf{t}_k\mathbf{p}_k^T + \mathbf{E}\\ \mathbf{Y} &= \mathbf{T}\mathbf{C}^T + \mathbf{F} = \mathbf{t}_1\mathbf{c}_1^T + \cdots + \mathbf{t}_k\mathbf{c}_k^T + \mathbf{F}\end{aligned} \tag{2.37}$$

where PLS simultaneously reduces the dimensionality of the **X** and **Y** matrices by seeking those latent variables that maximise Equation (2.27).

2.2.4.3 The PLS Algorithm, Derived from the NIPALS PCA Algorithm.

PLS can be formulated as an eigenvalue problem, Equations (2.32) to (2.35), consequently all eigenvalue decomposition algorithms can be applied. In this section, the NIPALS version of the algorithm is discussed. The NIPALS based PLS algorithm extracts one factor at a time, with each factor being obtained through the repeated regression of **X** on **u** to obtain an updated **w** vector, and then **X** is regressed on the weight vector **w** to obtain an updated score vector **t**. The algorithm continues with the regression of **Y** on **t** to obtain an updated **q** vector and finally the regression of **Y** on **q** provides an enhanced vector **u**, where the improvement is measured in terms of the covariance between the scores ($\mathbf{u}^T\mathbf{t}$). This sequence is repeated in an iterative manner until convergence is reached.

These four steps can be combined as follows (Höskuldsson, 1988):

$$\begin{aligned} &\mathbf{w} = \mathbf{X}^T\mathbf{u} \\ &\mathbf{w} = \mathbf{w}/norm(\mathbf{w}) \\ &\mathbf{t} = \mathbf{X}\mathbf{w} \\ &\mathbf{q} = \mathbf{Y}^T\mathbf{t} \\ &\mathbf{q} = \mathbf{q}/norm(\mathbf{q}) \\ &\mathbf{u} = \mathbf{Y}\mathbf{q} \end{aligned} \quad \rightarrow \quad \begin{aligned} &\mathbf{w} = \mathbf{X}^T\mathbf{u} \\ &c = norm(\mathbf{w}) \\ &\mathbf{t} = \mathbf{X}(\mathbf{X}^T\mathbf{u}/c) \\ &\mathbf{q} = \mathbf{Y}^T(\mathbf{X}\mathbf{X}^T\mathbf{u}/c) \\ &d = norm(\mathbf{q}) \\ &\mathbf{u} = \mathbf{Y}(\mathbf{Y}^T\mathbf{X}\mathbf{X}^T\mathbf{u}/cd) \end{aligned} \tag{2.38}$$

After convergence, Equation (2.35) results with $\lambda = cd$. As the NIPALS PCA algorithm is the same as solving the eigenvalue problem $\mathbf{X}\mathbf{X}^T$ and using the fact that the sequence in Equation (2.26) converges, the NIPALS based PLS algorithm can be considered to be equivalent to the power method applied to the eigenvalue problem of $\mathbf{Y}\mathbf{Y}^T\mathbf{X}\mathbf{X}^T$ in Equation (2.35). Again, the method belongs to the class of *power iteration* and takes advantage of the fact that the sequence:

$$\frac{\mathbf{u}}{\|\mathbf{u}\|}, \frac{(\mathbf{Y}\mathbf{Y}^T\mathbf{X}\mathbf{X}^T)\mathbf{u}}{\|\mathbf{u}\|}, \frac{(\mathbf{Y}\mathbf{Y}^T\mathbf{X}\mathbf{X}^T)^2\mathbf{u}}{\|\mathbf{u}\|}, \frac{(\mathbf{Y}\mathbf{Y}^T\mathbf{X}\mathbf{X}^T)^3\mathbf{u}}{\|\mathbf{u}\|}, \cdots \tag{2.39}$$

converges, under certain assumptions, to an eigenvector corresponding to the largest eigenvalue of $\mathbf{Y}\mathbf{Y}^T\mathbf{X}\mathbf{X}^T$. The assumptions are that the eigenvalues are significantly different, and that the starting vector is not identical to an eigenvector that corresponds to one of the smaller eigenvalues. Using Equations (2.32) - (2.34) as different starting points, the vectors **w**, **q**, and **t**, can be calculated using any algorithm that solves the symmetric eigenvalue problem.

Note that if **Y** consists of only one response variable the algorithm becomes non-iterative, since the **w** vector is found directly from $\mathbf{w} = \mathbf{X}^T\mathbf{y} / \| \mathbf{X}^T\mathbf{y} \|$ where $\mathbf{q} = 1$. In this case the PLS algorithm becomes very fast and does not have the convergence problems of the power method that is non-iterative. If **Y** consists of more than one response variable, the algorithm is associated with all the normal issues related to the power method. After convergence, the first latent variable is subtracted as a rank one reduction of the **X** and **Y** matrices, $\mathbf{X}_{j+1} = \mathbf{X}_j - \mathbf{t}_j\mathbf{p}_j^T$ and $\mathbf{Y}_{j+1} = \mathbf{Y}_j - b_j\mathbf{t}_j\mathbf{q}_j^T$, respectively. The algorithm is given in Algorithm 2.2, and describes the standard PLS algorithm based on the NIPALS or the power method (Wold *et al.*, 1983).

Another difference between PLS and PCR, is that in PLS, **Y** (m x p) is modelled separately for each pair of latent variables extracted, whilst in PCR all the latent variables are extracted before regressing **Y** on the latent variable matrix **T**. However, there is no difference between linearly regressing **Y** on the individual latent pairs or on the final **T** matrix, since the latent variables in **T** are mutually orthogonal, as discussed in Section 2.2.3.2. Thus, both approaches could be applied for PCR and PLS as they collapse to the same solution for the linear case.

```
function [Yp,T,P,W,U,Q]= PLS(X,Y,lv,lim)   % Standard PLS algorithm using the power method
Yp = zeros(size(Y));                       % Initializing
  for j = 1 : lv,                          % Repeat until lv latent variables are calculated
    u = Y(:, 1);                           % The first variable (or the one with highest variance)
    w = X'*u;                              % Estimate the weight vector
    w = w/norm(w);                         % Normalize to unit length
    conv = limit + 1;                      % Initializing
    while conv > limit,                    % Repeat power iterations until convergence
      uold = u;                            % Retain the old Y score vector to regulate the convergence
      t = X*w;                             % Calculate the X score vector
      q = Y'*t;                            % Estimate the loading vector for Y
      q = q/norm(q);                       % Normalize to unit length
      u = Y*q;                             % Update u using the linear combination q
      w = X'*u;                            % Estimate the weight vector
      w = w/norm(w);                       % Normalize to unit length
      conv = norm(u-uold)/norm(u);         % Convergence if conv < limit (and no. of iter. > max iter.)
  end                                      % Inner loop of iterations (while loop)
  p = X'*t; /t'*t;                         % Calculate the loading vector for X
  b = u'*t; /t'*t;                         % Calculate the regression constant for u on t
  X = X - t*p';                            % Rank one reduction of the X matrix
  Y = Y – b.*t*q';                         % Rank one reduction of the Y matrix
  Yp = Yp + b.*t*q';                       % Calculate the Y predicted
  T(:, j) = t;                             % Store the X scores
  P(:, j) = p;                             % Store the X loadings
  W(:, j) = w;                             % Store the X weights
  U(:, j) = u;                             % Store the Y scores
  Q(:, j) = q;                             % Store the Y loadings
end                                        % Outer loop of iterations (for loop)
```

Algorithm 2.2. The PLS algorithm (MATLAB code).

2.2.4.4 The Weight Vector in PLS

Consider the case of one response variable **y**, where the **X** matrix has been mean centred. As mentioned in the previous section, with only one response variable the PLS algorithm is non-iterative, and the scores and loadings can be calculated directly. The most important step is the calculation of the weight vector, $\mathbf{w} = \mathbf{X}^T\mathbf{u}$, in Algorithm 2.2. With only one response variable,

$\mathbf{q} = 1$, and $\mathbf{u} = \mathbf{y}$. From $\mathbf{w} = \mathbf{X}^{\mathrm{T}}\mathbf{y}$, it can be seen that the k^{th} weight is estimated *independently* as the inner product or covariance between the response variable, $\mathbf{y}$, and its corresponding predictor variable, $\mathbf{x}_k$, thus $w_k = \mathbf{x}_k^{\mathrm{T}}\mathbf{y}$, where $k \in \{1,\cdots,n\}$, and n is the number of $\mathbf{X}$ variables. This can be expanded further (since $\mathbf{X}$ being mean centred):

$$w_k = \mathbf{x}_k^{\mathrm{T}}\mathbf{y} \propto cor(\mathbf{x}_k, \mathbf{y})\, std(\mathbf{x}_k) \tag{2.40}$$

$\mathbf{w}$ is then normalized to unit length. Thus, in principle $\mathbf{y}$ is not required to be mean centred for the calculation of the weights, but it is recommended because of the regression of $\mathbf{t}$ on $\mathbf{y}$. Generally, this is due to the regression of $\mathbf{u}$ on $\mathbf{t}$ in PLS is performed without applying any constant term ($\hat{\mathbf{u}} = b\mathbf{t}$). However, if $\mathbf{X}$ is centred (i.e. $\mathbf{t}$ is centred) and $\mathbf{Y}$ is not centred (i.e. $\mathbf{u}$ is not centred), a constant term is needed to model $\mathbf{u}$ efficiently ($\hat{\mathbf{u}} = a + b\mathbf{t}$). From Equation (2.40), the construction of the weight vector can be understood. The score vector $\mathbf{t}$ is a linear combination (or a weighted sum) of the $\mathbf{X}$ variables:

(i) where the weight of the k^{th} variable, w_k, has the same sign as the correlation between $\mathbf{x}_k$ and $\mathbf{y}$.

and

(ii) the magnitude of the k^{th} weight, w_k, is a product of the correlation coefficient and the standard deviation of the given variable, $\mathbf{x}_k$.

Hence, the magnitude of the weights comprise two parts, the correlation and the variance. If the variance of the variables is not related to the importance of the variables, as through Beers law, the $\mathbf{X}$ variables are often scaled to a standard deviation of unity. Equation (2.40) then reduces to $\mathbf{w}_k = cor(\mathbf{x}_k, \mathbf{y})$, with the weights calculated from the correlation between $\mathbf{x}_k$ and $\mathbf{y}$. It should be noted that, if $\mathbf{X}$ is orthonormal, $\mathbf{w}$ will be identical to the regression coefficients from least squares.

2.2.5 Comparison

There are three main benefits of using PCR and PLS regression compared to OLS.

(i) The regression variables, $\mathbf{T}$, are linearly independent (orthogonal) so that the problem of multicollinearity is addressed.

(ii) The reduction in the number of parameters through regression on $\mathbf{T}$ instead of $\mathbf{X}$ can often make an underdetermined system overdetermined.

(iii) Generally, only the most important latent variables, $\mathbf{T}$, are included thus that the risk of modelling noise in the data is reduced.

Alternatively, one can use ridge regression (Frank and Friedman, 1993). The advantage of PCR and PLS over ridge regression (Section 2.2.1.2), is that the final model is more easily interpreted, since the weights are found independently of each other (Sections 2.2.4.4). Consequently, the regression coefficient from ridge regression is often lacking the structure found in the regression coefficient from PCR and PLS, making it more difficult to relate the importance of each variable with the underlying physical process. This is often verified by looking at the difference in the structure of the regression coefficients when applied to typical spectral data. However, the prediction ability of the three methods is often found to be similar for large data sets (Frank and Friedman, 1993).

The difference between PCR and PLS is the way in which the score matrix **T** is estimated:

- PCR maximizes the variability in the **X** matrix only, by maximizing the length of each score vector **t**, subject to the constraint $\|\mathbf{p}\| = 1$.
- PLS maximizes simultaneously the variability in both the **X** and **Y** matrices, by maximizing the covariance between **t** and **u**, subject to the constraint $\|\mathbf{w}\| = \|\mathbf{q}\| = 1$.
- As mentioned earlier, there is often a tendency for those components with the largest variances to best explain the response variables. If this is not the case, PLS is more appropriate, Almøy (1996). However, if the major sources of variance in **X** are unrelated to **Y**, both techniques will fail to produce satisfactory models. Variable selection methods should then be applied.

2.3 Multivariate Multiple Nonlinear Modelling

As long as the model depends linearly on the adjustable parameters, the solution will lead to a set of linear equations in the parameters, which can be cast in the form of matrix equations. Thus any linear regression procedure can be used to obtain a solution. However, if the underlying model is nonlinear in the parameters, alternative approaches are required. The progression from linear to nonlinear modelling is challenging, with many of the linear algorithms not being easily extended to the nonlinear case. This is the case in PLS. The first approach adopted is to linearize the problem, for example by applying a suitable transformation or expanding the **X** matrix with functions of **X**, as in polynomial least squares fitting. Another method is to expand the **T** matrix with functions of **T**, to reduce the size of the resultant matrix, Martens and Næs (1989); Blanco *et al.* (2000). However, there exist situations where finding a suitable transformation is not possible and hence more sophisticated methods are required. This section describes the theory behind such methods.

Consider a data set consisting of m observations on two sets of variables, **X** and **Y**, where the response variables **Y** (m x p) are to be predicted by a set of predictor variables **X** (m x n), and the underlying relationship is assumed to be nonlinear. Many different approaches have been proposed in the fields of chemometrics, statistics and neural networks. By restricting the neural network to a two-layer network with a linear output, the general regression model takes the form:

$$\mathbf{Y} = \sum_{j=1}^{A} f_j(\mathbf{X}_j \mathbf{w}_j)\mathbf{q}_j^{\mathrm{T}} + \mathbf{E} \qquad (2.41)$$

where $\mathbf{w}_j$ and $\mathbf{q}_j$ are the **X** and **Y** weighting vectors, f_j is a nonlinear function, and **E** is the error matrix. A is the number of latent variables, ridge functions, or nodes, depending on the methodology. The methods encapsulated by Equation (2.41) are termed projection-based regression methods and are the focus of the nonlinear regression methods discussed in the Thesis.

An alternative projection based interpretation of Equation (2.41) is now given. The **X** variables are projected onto a one dimensional score vector $\mathbf{t}_j$ by the linear combination, $\mathbf{t}_j = \mathbf{X}_j \mathbf{w}_j$, whilst the **Y** variables are projected onto the **Y** score vector $\mathbf{u}_j$ by the linear combination, $\mathbf{u}_j = \mathbf{Y}_j \mathbf{q}_j$. The inner relationship between the pair of latent variables $\mathbf{u}_j$ and $\mathbf{t}_j$ is then fitted through the nonlinear regression function, f_j, using a univariate regression method, $\mathbf{u}_j = f_j(\mathbf{t}_j) + \mathbf{e}_j$.

The regression model in Equation (2.41) is general and includes nonlinear PLS, Smooth Multiple Additive Regression Technique (SMART), and the Radial Basis Function Network (RBFN). Parametric models such as quadratic partial least squares (QPLS), multiple linear regression (MLR) and feed forward network (FFN), are special forms of Equation (2.41), where the function, f_j, can be quadratic, linear or logistic (Sekulic *et al.* 1993). Additive models, such as general additive models (Hastie and Tibshirani, 1990) or Multivariate Adaptive Regression Splines (MARS) by Friedman (1991), do not follow the projection based approach, Equation (2.41), and are therefore not discussed further.

The main difference between the projection based modelling procedures are that for neural networks the nodes are optimised simultaneously, whilst for nonlinear PLS and SMART, each latent variable (or ridge function) is optimised separately. In particular for nonlinear PLS, the matrices **X** and **Y** are decomposed for each latent variable using a rank one reduction: $\mathbf{X}_{j-1} = \mathbf{X}_j - \mathbf{t}_j \mathbf{p}_j^T$ and $\mathbf{Y}_{j-1} = \mathbf{Y}_j - \hat{\mathbf{u}}_j \mathbf{q}_j^T$; where $\mathbf{p}_j = \mathbf{X}_j{}^T \mathbf{t}_j / (\mathbf{t}_j \mathbf{t}_j)$ and $\hat{\mathbf{u}}_j = f_j(\mathbf{t}_j)$. Thus, the only difference from the linear case is the construction of $\hat{\mathbf{u}}_j$.

2.3.1 Nonlinear Least Squares

To optimize the fit using least squares when the model is nonlinear, an objective function is defined, and an optimization method is used to find the parameters that minimize the objective or error function. Although a wide spectrum of methods exists for optimisation, the methods can be broadly categorized in terms of whether the derivative information is used. The objective function may include constraints, for example each parameter should lie within a given range. Search methods that use function evaluations include the simplex search method, Nelder and Mead (1965). This approach is most suitable for problems that are highly noisy or have discontinuities. Gradient methods are generally more appropriate when the function to be minimized is continuous in its first derivative. Higher order gradient methods such as Newton's method, are only suitable when the second order information can be calculated. Here the simpler Gauss-Newton method is explained, as it is the optimization methodology used in the Thesis. This method belongs to the family of gradient methods and only the first derivative information is required.

Let $\mathbf{X}$ (m x n) and $\mathbf{Y}$ (m x 1) be matrices comprising n predictors and one response variable, respectively. The aim is to minimise the error of the objective function using least squares. Equation (2.41) can be simplified when there is only one response variable:

$$\mathbf{y} = f(\mathbf{X}\boldsymbol{\beta}) + \mathbf{e} \tag{2.42}$$

where $\boldsymbol{\beta}$ represents the weight vector or the regression coefficients. Thus, the objective function $F_{(\hat{\boldsymbol{\beta}})}$ is given by:

$$F_{(\hat{\boldsymbol{\beta}})} = \left\| \mathbf{y} - f(\mathbf{X}\hat{\boldsymbol{\beta}}) \right\|^2 = \left\| \mathbf{e} \right\|^2 = \mathbf{e}^{\mathrm{T}}\mathbf{e} \tag{2.43}$$

The optimisation method may be Gauss-Newton or Levenberg-Marquardt. In this approach, the search direction $\partial\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{\beta}}_{l+1} - \hat{\boldsymbol{\beta}}_{l}$ is found from a Taylor series expansion of $F_{(\hat{\boldsymbol{\beta}})}$ about the current values of $\hat{\boldsymbol{\beta}}_{l}$, retaining only the first-order terms. Consider Newton's method where $\nabla F_{(\hat{\boldsymbol{\beta}})}$ and $\nabla^2 F_{(\hat{\boldsymbol{\beta}})}$ are the first and second derivatives of $F_{(\hat{\boldsymbol{\beta}})}$ with respect to $\hat{\boldsymbol{\beta}}$, respectively. The search direction $\partial\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{\beta}}_{l+1} - \hat{\boldsymbol{\beta}}_{l}$ (n x 1) can be written as:

$$\partial\hat{\boldsymbol{\beta}} = -[\nabla^2 F_{(\hat{\boldsymbol{\beta}})}]^{-1} \nabla F_{(\hat{\boldsymbol{\beta}})} \tag{2.44}$$

The gradient can be defined as $\nabla F_{(\hat{\boldsymbol{\beta}})} = 2\mathbf{J}^{\mathrm{T}}_{(\hat{\boldsymbol{\beta}})} \mathbf{e}_{(\hat{\boldsymbol{\beta}})}$ where $\mathbf{J}_{(\hat{\boldsymbol{\beta}})} = \nabla \mathbf{e}_{(\hat{\boldsymbol{\beta}})}$. ($m$ x n) is known as the Jacobian matrix, estimated from $\mathbf{J}_{(\mathrm{i,k})} = [\partial \mathrm{e}_{\mathrm{i}} / \partial \hat{\beta}_{\mathrm{k}}]$. The Hessian matrix, $\mathbf{H} = \nabla^2 F_{(\hat{\boldsymbol{\beta}})}$ (n x n) is then given by $\mathbf{H} = 2\mathbf{J}^{\mathrm{T}}_{(\hat{\boldsymbol{\beta}})}\mathbf{J}_{(\hat{\boldsymbol{\beta}})} + 2\mathbf{S}_{(\hat{\boldsymbol{\beta}})}$, where $\mathbf{S}_{(\hat{\boldsymbol{\beta}})} = \mathbf{e}_{(\hat{\boldsymbol{\beta}})} \nabla^2 \mathbf{e}_{(\hat{\boldsymbol{\beta}})}$. For Gauss-Newton, $\mathbf{H}$ is

approximated linearly by setting the second order term $\mathbf{S}_{(\hat{\beta})} = 0$, i.e. $\mathbf{H}_{(\hat{\beta})} \approx 2\mathbf{J}^{\mathrm{T}}_{(\hat{\beta})}\mathbf{J}_{(\hat{\beta})}$. This gives the Gauss-Newton search direction as:

$$\partial\hat{\boldsymbol{\beta}} = -[2\mathbf{J}^{\mathrm{T}}\mathbf{J}]^{-1}2\mathbf{J}^{\mathrm{T}}\mathbf{e} = -[\mathbf{J}^{\mathrm{T}}\mathbf{J}]^{-1}\mathbf{J}^{\mathrm{T}}\mathbf{e} \tag{2.45}$$

which is the least squares solution of $\mathbf{e} = -\mathbf{J}\,\partial\hat{\boldsymbol{\beta}}$. If the covariance matrix $[\mathbf{J}^{\mathrm{T}}\mathbf{J}]$ is singular, it can be made invertible by letting $\dot{\mathbf{H}} = [\mathbf{J}^{\mathrm{T}}\mathbf{J}] + \psi\mathbf{I}$ where ψ is a constant, thus $\partial\hat{\boldsymbol{\beta}}$ becomes

$$\partial\hat{\boldsymbol{\beta}} = -[\dot{\mathbf{H}}]^{-1}\mathbf{J}^{\mathrm{T}}\mathbf{e} \tag{2.46}$$

This is the Levenberg-Marquardt approach, and corresponds to the ridge regression expression given in Equation (2.8). As $\psi \to 0$, the Levenberg-Marquardt approach tends to Gauss-Newton and as $\psi \to \infty$, the Levenberg-Marquardt method tends to the Steepest Descent (SD) method, $\partial\hat{\boldsymbol{\beta}} = -\mathbf{J}^{\mathrm{T}}\mathbf{e}$. The similarity between the steepest descent expression and the estimation of the weight vector in PLS ($\mathbf{w} = \mathbf{X}^{\mathrm{T}}\mathbf{y}$), Equation (2.40) should be observed with the two approaches being compared in detail in Chapter 3. The constant ψ, from Equation (2.8), can either be selected such that $\dot{\mathbf{H}}$ becomes non-singular, or through a convergence scheme (Baylis and Pradhan, 1984). That is, as ψ increases, $F_{(\hat{\beta})}$ will eventually decrease, since a small step in the steepest descent direction is taken.

From the Taylor series expansion used to establish Newton's method, it is assumed that the function $f(\mathbf{X}\hat{\boldsymbol{\beta}})$ and its first derivative are continuous. Newton's method is a fast optimisation method when the function is continuous, but it has a number of limitations:

(i) The method does not necessarily find the global solution if multiple local solutions exist, but this is a characteristic of all methods that use gradient information to define the search direction.

(ii) The method involves matrix inversion.

(iii) The method requires the analytical first partial derivatives, which may not be practical to obtain. However, it is possible to use the numerical first partial derivative (Quasi Gauss-Newton).

A common feature of all estimation procedures is that they require the user to specify start values, initial step sizes, and a criterion or limit for convergence. All methods begin with a particular set of initial estimates (*starting values*) that will be updated at each iteration; the *step direction* determines the direction, the *step size* determines how much the parameters will be updated, and the *convergence criterion* determines when the iteration process will stop.

The greatest problem in unconstrained function minimisation is *local minima*. Local minima can be thought of as local "valleys" in the objective function. For example, a particular objective function may increase, regardless of the step size applied. However, if the parameters are changed randomly, the objective function may decrease further. The problem can be overcome by selecting a different parameter space, i.e. by trying a number of different starting values.

2.3.2 Neural Networks

Only a brief review of neural networks is given, as a vast literature exists on the subject, (Bishop (1995), Lippmann (1987)). The review focuses on the basic two-layer network. The area of training or learning algorithm is not considered but the methods are similar to these described in the nonlinear least squares section, Section 2.3.1. Two-layer networks were widely studied in the 1970's, and the history of such networks from this period have been reviewed by Widrow and Lehr (1990). Considering only one response variable, the objective function F_{NN} is given by:

$$F_{NN} = \left\| \mathbf{y} - \sum_{j=1}^{A} f_j(\mathbf{X}\hat{\boldsymbol{\beta}}_j) \right\|^2 \tag{2.47}$$

The key aspect of a neural network is that the hidden functions are themselves adapted to the data as part of the training process, consequently the number of functions increases as the complexity of the problem increases.

Furthermore, the training, i.e. the optimization of Equation (2.47), and the number of nodes in the network, A, are often controlled by using a second data set, a test set. Validation should be performed on a separate third data set, the validation set, which has not been used in any aspect of the training of the network.

Alternatively the optimal number of nodes, A, can be found by the backward pruning method of Friedman (1985), where nodes of least importance are eliminated from an over-fitted model using cross validation. The problem of selecting the appropriate number of nodes is addressed by Murata *et al.* (1994), who utilize Akaike's Information Criterion (AIC).

2.3.3 Nonlinear PCR

The first step in nonlinear principal component regression is the same as for the linear case, i.e. a principal component analysis is performed on the **X** matrix. A nonlinear regression method is then applied to the retained principal components. As discussed in Section 2.2.4.5, the latent variables are orthogonal and the major latent variables will contain less noise than the individual variables. Thus, the issue of multicollinearity is removed and the possibility of fitting noise is normally reduced. Furthermore, the reduction in the number of variables modelled often ensures that the system is overdetermined.

Restricting the nonlinear model to the projection based family of methods, nonlinear PCR for one response variable is defined as:

$$\mathbf{y} = f(\mathbf{T}\boldsymbol{\beta}) + \mathbf{e} \tag{2.48}$$

where f is a nonlinear function, $\mathbf{T}$ is the matrix of latent variables and $\boldsymbol{\beta}$ is a vector of coefficients.

Nonlinear PCR approaches range from the Optimal Minimal Neural Interpretation of Spectra (OMNIS) (Borgaard and Thodberg, 1992), where the emphasis is on finding the most parsimonious model possible consistent with the data, to the use of polynomial PCR as described by Vogt (1989). A recent discussion involving nonlinear PCR is given by Gemperline (1992). Finally, in the work of Næs and Isaksson (1995), PCR was adjusted for the presence of nonlinearities, and Næs *et al.* (1993) discussed the relationship between neural networks and principal component regression.

2.3.4 Nonlinear PLS

A number of nonlinear extensions to PLS have also been developed. Most follow the expression given in Equation (2.41):

$$\mathbf{Y} = \sum_{j=1}^{A} f_j(\mathbf{X}_j\mathbf{w}_j)\mathbf{q}_j^T + \mathbf{E} \tag{2.41}$$

Given a linear combination of the response variables, $\mathbf{u} = \mathbf{Yq}$, the modelling task in PLS can be defined as finding the best projection $\mathbf{w}$, for each sequential projection. A nonlinear function, f, is then fitted between the pair of scores $\mathbf{t}$ and $\mathbf{u}$, such that $\mathbf{u} = f(\mathbf{t}) + \mathbf{e} = \hat{\mathbf{u}} + \mathbf{e}$.

For the linear case, the weight $\mathbf{w}_L$ is defined as the largest normalised eigenvector of the product $\mathbf{X}^T\mathbf{Y}\mathbf{Y}^T\mathbf{X}$, i.e. $\mathbf{X}^T\mathbf{Y}\mathbf{Y}^T\mathbf{X}\mathbf{w}_L = \lambda\mathbf{w}_L$. For nonlinear PLS, using the weight matrix $\mathbf{w}_L$ directly is reasonable if the nonlinearity is weak. However, if the nonlinearity is strong, an iterative updating procedure is required. In this case $\mathbf{w}_L$ can be chosen as a starting vector for an updating procedure. Both these approaches have been utilized in nonlinear PLS. A selection of the main published nonlinear PLS algorithms are listed in Table 1.

Method	Author	Comments
Quadratic-PLS	Wold *et al.*, (1989)	Minimizing of $\{\mathbf{u}_i - f_i(\mathbf{t}_i)\}$ using steepest descent method.
NL-PLS	Frank, (1990)	Using linear $\mathbf{w}_L$ and nonparametric inner fit of $\{\mathbf{u}_i - f_i(\mathbf{t}_i)\}$.
NN-PLS	Qin & McAvoy, (1992)	Using $\mathbf{w}_L$ and 2-layer feed forward NN for f: $\{\mathbf{u}_i - f_i(\mathbf{t}_i)\}$.
Spline-PLS	Wold *et al.*, (1992)	Updating $w_i = cor[\mathbf{u}, f\{\mathbf{X}_i(\sigma_t / \sigma_{\mathbf{X}i})\}]\sigma_{\mathbf{X}i}$, (§ 2.3.4.3).
NN-PLS	Holcomb & Morari, (1992)	PLS algorithm integrated in a neural network.
RBF-PLS	Walcak & Massart, (1996)	Radial basis function as transformation prior to linear PLS.
NN-PLS	Malthouse *et al.*, (1997)	NN deflating $\mathbf{X}$ and $\mathbf{Y}$, maximizing $\|\mathbf{X} - \mathbf{t}\mathbf{p}^T\| + \|\mathbf{Y} - f(\mathbf{t})\mathbf{q}^T\|$.
RBF-PLS	Wilson *et al.*, (1997)	Using linear $\mathbf{w}_L$ and RBF network on inner relationship
NN-PLS	Durand & Sabatier, (1997)	Spline transformation before applying linear PLS
NT-PLS	Höskuldsson , (1998)	Polynomial PLS, includes previous modelled score terms.
Quadratic-PLS	Baffi *et al.*, (1999a)	Minimizing $\{\mathbf{u}_i - f_i(\mathbf{t}_i)\}$ using Gauss-Newton method.
RBF-PLS	Baffi *et al.*, (1999b)	Gauss-Newton updating of weights as above, RBF inner fit.
BTPLS	Li *et al.*, (2000)	Gauss-Newton updating of weights, Box-Tidwell inner fit.

Table 1. Brief summary of a selection of nonlinear PLS algorithms.

The various nonlinear PLS can be divided into two groups:

(I) Weight vector, $\mathbf{w}_L$, from linear PLS is used.

Frank (1990) proposed a nonlinear PLS algorithm (NL-PLS) where a local linear smoothing procedure was used to fit the relationship between each pair of scores with the bandwidth being estimated using cross validation, Frank (1995). Qin and McAvoy (1992) used a feed forward neural network to fit the inner relationship whilst Wilson *et al.* (1997) used a radial basis functions network (RBFN). An alternative RBFN approach was proposed by Walczak and Massart (1996), where radial basis functions were used as a nonlinear transformation to linearize the **X** variables prior to applying PLS. Spline transformations have also been used to linearize the predictor variables before the application of PLS (Durand and Sabatier, 1997).

(II) The weight vector, **w**, is updated through an updating procedure.

In 1989, Wold proposed polynomial (quadratic) PLS. Here the weight vector **w** was updated using a form of steepest descent optimisation. Later, Wold (1992) developed a Spline PLS algorithm where spline functions (quadratic or cubic) were used as the smoothing functions. The updating procedure differed to that of polynomial PLS in that the weight vector **w** is derived from the covariance criterion used in linear PLS and is modified to include the nonlinear mapping in the calculation of the covariance function.

Based on the quadratic PLS paper of Wold *et al.* (1989), an error based updating procedure was developed by Baffi *et al.* (1999a,b). It differs from that of Wold *et al.* (1989) since it directly optimises the error function using Gauss-Newton (GN) optimisation. Another nonlinear PLS method is the quadratic PLS method of Höskuldsson (1992, 1998). Here, both polynomial and cross terms of the current and the previous modelled latent variables are considered. Since the numbers of terms increases considerably for only a few latent variables, only a subset of terms were included. These were selected using statistical significance testing.

Neural network based PLS algorithms have been published using different objective functions. Holcomb and Morari (1992), included a linear PLS model into the first layer of a neural network, with the second layer being used to model the underlying nonlinearity. The objective of the neural network PLS approach of Malthouse *et al.* (1997) was to minimise the sum of the variance of **X** and **Y** sequentially, i.e. for each latent variable. This is one interpretation of the PLS criterion (Höskuldsson, 1996). Later, Bakshi and Utojo (1998) developed a nonlinear continuous regression algorithm based on the continuum regression ideas of Stone and Brooks (1990). The result, Nonlinear Continuum Regression (NLCR), was claimed to embrace the methods of OLS, PCR, PLS, Projection Pursuit Regression (PPR) and nonlinear PLS.

The next three sections describe in detail the nonlinear PLS algorithms proposed by Wold (1989,1992) and Baffi *et al.* (1999a,b). These approaches form the basis of the subsequent work.

2.3.4.1 The Steepest Descent PLS Algorithm

In 1989, Wold *et al.* proposed a polynomial (quadratic) PLS algorithm where the inner mapping was modelled by a quadratic function. However, the authors stated that any nonlinear function that is continuous and differentiable could be used. The algorithm retains the form of the linear PLS algorithm, including the orthogonality of the latent variables, $[\mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_A]$ with two modifications to the NIPALS based PLS algorithm being proposed:

(i) For each pair of latent variables, a nonlinear (quadratic) function was used in the inner relationship.

(ii) A weight updating scheme was introduced whereby the weight vector **w** was updated using an optimisation routine.

For each pair of latent variables (**t**, **u**), the objective function from Equation (2.43) becomes:

$$F_{(\mathbf{w})} = \| \mathbf{u} - f(\mathbf{Xw}) \|^2 = \| \mathbf{u} - \hat{\mathbf{u}} \|^2 \tag{2.49}$$

The objective of the optimization procedure defined in Equation (2.49) is to determine a weight vector **w** that identifies a score vector such that it minimises the error function, thereby identifying the underlying nonlinear function.

Wold *et al.* (1989) proposed updating the weight vector **w** by means of a first order Taylor series expansion of $f(\mathbf{Xw})$ about the current weight vector $\mathbf{w}_l$, and solving it with respect to the weight increments $\partial\mathbf{w}$, i.e. $\mathbf{w}_{l+1} = \mathbf{w}_l + \partial\mathbf{w}$. From the Taylor series expansion, the mismatch, **e**, between $\mathbf{u} = \mathbf{Yq}$ and $\hat{\mathbf{u}} = f(\mathbf{Xw})$, can be approximated by:

$$\mathbf{e} = \mathbf{u} - \hat{\mathbf{u}} \approx -\mathbf{J}\partial\mathbf{w} \tag{2.50}$$

where $\hat{\mathbf{u}}$ is a continuous function, differentiable with respect to **w**, and **J** is defined as $\mathbf{J}_{(i,k)} = [\partial e_i / \partial w_k]$. Furthermore, $[\partial e_i / \partial w_k] = -[\partial \hat{u}_i / \partial w_k]$ since $\mathbf{e} = \mathbf{u} - \hat{\mathbf{u}}$ and **u** is assumed constant with respect to **w**. Using the partial derivative form $\partial\hat{\mathbf{u}}/\partial\mathbf{w} = (\partial\hat{\mathbf{u}}/\partial\mathbf{t})(\partial\mathbf{t}/\partial\mathbf{w})$, the Jacobian becomes:

$$\mathbf{J} = -\mathbf{X}\,diag(\partial\hat{\mathbf{u}}/\partial\mathbf{t}) \tag{2.51}$$

since $\partial\mathbf{t}/\partial\mathbf{w}$ is given by $\partial(\mathbf{Xw})/\partial\mathbf{w} = \mathbf{X}$. The term $\partial\hat{\mathbf{u}}/\partial\mathbf{t}$ is the derivative of the nonlinear inner function $\hat{\mathbf{u}} = f(\mathbf{t})$ fitted between **u** and **t**. Wold *et al.* (1989) applied a steepest descent method to solve Equation (2.50), thus the approach is termed Steepest Descent PLS (SDPLS). Equation (2.50) can be solved as follows $\partial\mathbf{w} = -\mathbf{J}^T\mathbf{e}$, i.e. $\partial\mathbf{w} = (\mathbf{X}\partial\hat{\mathbf{u}}/\partial\mathbf{t})^T(\mathbf{u} - \hat{\mathbf{u}})$.

Closer examination reveals that the solution to solve Equation (2.50) is similar to that of PLS with one latent variable, since $\partial\mathbf{w} = -\mathbf{J}^T\mathbf{e}$ is the same as the covariance criterion used in linear PLS ($\mathbf{w} = \mathbf{X}^T\mathbf{u}$), where $-\mathbf{J}$ represents the **X** matrix and **e** represents the residual of the latent response, **u**. The difference is that in PLS, the weight vector is normalised, and the regression vector provides the solution to the problem. The regression vector, $\boldsymbol{\beta}$, is proportional to the weight vector when only one latent variable is included, ($\boldsymbol{\beta} = b\,\mathbf{w}$, where $b = \mathbf{t}'\mathbf{u}/(\mathbf{t}'\mathbf{t})$ from

Algorithm 2.2). The relationship between the steepest descent and PLS will be further discussed in Chapter 3.

```
...                                                    ...
    while conv > limit,                                % Repeat until convergence
        uold = u;                                      % Retain the old Y score vector to regulate the convergence
        t = X*w;                                       % Calculate the X score vector
(i)     [û, dûdt] = f(t, u);                           % Calculate the nonlinear fit, and the 1st derivative
(ii)    q = Y'*û;                                      % Regress Y on û  (not on t as in linear PLS)
        q = q/norm(q);                                 % Normalize to unit length
        u = Y*q;                                       % Update u using the linear combination q
(iii)   e = u – û;                                     % Calculate the error vector e
(iv)    J = X.*[dûdt(:, ones(1, xcol))];               % Calculate the positive Jacobian (Kronecker product)
(v)     dw = J'*e;                                     % Calculate the weight update vector using least squares
(vi)    w = w + dw;                                    % Improved weight vector
        w = w/norm(w);                                 % Normalize to unit length
        conv = norm(u-uold)/norm(u);                   % Convergence if s < limit (and no. of iter. > max iter.)
    end                                                % Inner loop of iterations (while loop)
```

Algorithm 2.3. The Steepest Descent PLS (SDPLS) algorithm (MATLAB code).

The difference between linear PLS and the SDPLS algorithm is highlighted in Algorithm 2.3, by the numbered points. The difference between SDPLS and linear PLS can be summarized as:

(i) A nonlinear function $\hat{\mathbf{u}}$ is fitted between the pair of scores, **t** and **u,** and the first derivative of $\hat{\mathbf{u}}$, $(\partial\hat{\mathbf{u}}/\partial\mathbf{t})$, is estimated.

(ii) Instead of regressing **Y** on **t** as in the linear case, **Y** is regressed on $\hat{\mathbf{u}}$.

(iii) The error $\mathbf{u}-\hat{\mathbf{u}}$, is calculated.

(iv) The Kronecker product $\dot{\mathbf{J}} = (\partial\hat{\mathbf{u}}/\partial\mathbf{t})(\partial\mathbf{t}/\partial\mathbf{w})$ is calculated, where $\dot{\mathbf{J}} = -\mathbf{J}$.

(v) The search direction $\partial\mathbf{w}$ is identified using the method of steepest descent.

(vi) An updated weight is calculated, and the procedure is repeated until convergence.

In the original SDPLS algorithm of Wold *et al*. (1989), the weight vector is also optimized with respect to the parameters of the nonlinear (quadratic) function. Furthermore, calculation of the increment of the weight vector, $\partial\mathbf{w} = \mathbf{w}_{l+1} - \mathbf{w}_l$, is based on **u** as opposed to $\mathbf{e} = \mathbf{u} - \hat{\mathbf{u}}$. Wold *et al*. (1989) stated that the approach was rather complicated and that it may converge slowly when the data lacks structure. The rank updating procedure is given by:

$$\mathbf{X} = \mathbf{t}_1\mathbf{p}_1^T + \cdots + \mathbf{t}_k\mathbf{p}_k^T + \mathbf{E}$$
$$\mathbf{Y} = \hat{\mathbf{u}}_1\mathbf{q}_1^T + \cdots + \hat{\mathbf{u}}_k\mathbf{q}_k^T + \mathbf{F} \qquad (2.52)$$

The method of steepest descent is the simplest of the gradient methods discussed in Section 2.3.1. The choice of direction is that where $\hat{\mathbf{u}}$ decreases most quickly with respect to $\mathbf{w}$, that is the direction of the gradient $\partial\mathbf{w} = -\mathbf{J}^T\mathbf{e}$.

The method is not guaranteed to attain the global minimum after an infinite number of iterations due to the issue of local minima, since the direction of the gradient is only controlled by local gradient information. Furthermore, the method generally has slow convergence, especially when approaching the minimum. The method hardly ever converges for badly scaled systems, i.e. if the eigenvalues of the Hessian matrix, $\mathbf{H}$, differ by several orders of magnitude. In this case it is highly dependent on a good choice of starting point as it only uses the first derivative information.

However, when the initial values are far from the global minimum, the first derivative generally describes the direction towards the global minimum. Thus, the steepest descent method is sometimes used for the first few iterations prior to reverting to Gauss-Newton (Hoyle, 1995). Finally, since there is no inversion of a covariance matrix involved, the method does not suffer any singularity or multicollinearity problems. Baffi *et al.*, (1999a) provide a comprehensive discussion of the methodology.

2.3.4.2 The Error Based Weight Updating Procedure

Due to the limitations of Steepest Descent PLS, the error based weight updating procedure was introduced (Baffi *et al.*, 1999a,b). It applies the unconstrained multivariable optimization method of Gauss-Newton, described in Section 2.3.1, to solve the same objective function as defined in Equation (2.45):

$$\partial\mathbf{w} = -[\mathbf{J}^T\mathbf{J}]^{-1}\mathbf{J}^T\mathbf{e} \qquad (2.53)$$

Thus, Equation (2.50) is solved by least squares. The difference between linear PLS and nonlinear PLS using the error based updating procedure is summarized in Algorithm 2.4. The primary difference with Algorithm 2.3 is the method by which $\partial\mathbf{w}$, in Algorithm 2.4, line (v), is calculated.

The method of Gauss-Newton involves the matrix inversion of the covariance matrix of the Jacobian, i.e. the covariance information is used in the construction of $\partial \mathbf{w}$. Thus convergence will generally be improved over the steepest descent approach, but the procedure may suffer from both singularity and multicollinearity.

```
...                                                    ...
   while conv > limit,                                 % Repeat until convergence
      uold = u;                                        % Retain the old Y score vector to regulate the convergence
      t = X*w;                                         % Calculate the X score vector
(i)   [û, dûdt] = f(t, u);                             % Calculate the nonlinear fit, and the 1st derivative
(ii)  q = Y'*û;                                        % Regress Y on û (not on u as in linear PLS)
      q = b/norm(q);                                   % Normalize to unit length
      u = Y*q;                                         % Update u using the linear combination q
(iii) e = u – û;                                       % Calculate the error vector e
(iv)  J = X.*[dûdt(:, ones(1, xcol))];                 % Calculate the Jacobian (sign cancels in (v))
(v)   dw = pinv(J'*J) * J'*e;                          % Calculate the weight update vector using least squares
(vi)  w = w + dw;                                      % Improved weight vector
      w = w/norm(w);                                   % Normalize to unit length
      conv = norm(u-uold)/norm(u);                     % Convergence if s < limit (and no. of iterations > max iter.)
   end                                                 % Inner loop of iterations (while loop)
```

Algorithm 2.4. The Error Based PLS (EBPLS) algorithm (MATLAB code).

2.3.4.3 The Spline PLS Updating Procedure

Wold (1992), proposed a methodology for the calculation of the weight vector for the nonlinear case, based on the covariance criterion defined in Equation (2.40). An initial estimate of **t**, **u** and the nonlinear function, f, between **t** and **u** is first obtained. Then each variable $\mathbf{x}_k$ is scaled to have the same variance as **t**, i.e. by using the scaling constant $v_k = std(\mathbf{t})/std(\mathbf{x}_k)$, where $std(\mathbf{t})$ and $std(\mathbf{x}_k)$ are the standard deviations of **t** and the k^{th} variable of **X**, respectively. Then each variable is regressed on **u**, using the nonlinear function, f, already identified between **u** and **t** using nonlinear least squares, as $f(v_k \mathbf{x}_k)$. Each weight w_k is then calculated as:

$$w_k = s_k \, COR(f(s_k \, v_k \, \mathbf{x}_k), \mathbf{u}) std(\mathbf{x}_k) \tag{2.54}$$

where the sign, s_k, can be found from the sign of $(\mathbf{t}^T \mathbf{x}_k)$, or as Wold (1992) showed, fitting both $f(\mathbf{x}_k v_k)$ and $f(-\mathbf{x}_k v_k)$, and then selecting the one with the highest absolute correlation

with **u**. After all weights are calculated, the obtained weight vector is normalized, i.e. $\mathbf{w} = \mathbf{w} / \| \mathbf{w} \|$. The algorithm is given in Algorithm 2.5.

```
...                                                    ...
    while conv > limit,                                % Repeat until convergence
        uold = u;                                      % Retain the Y score vector to regulate the convergence
        t = X*w;                                       % Calculate the X score vector
(i)     û = f(t, u);                                   % Calculate the nonlinear fit
(ii)    q = Y'*û;                                      % Regress Y on û (not on u as in linear PLS)
        q = q/norm(q);                                 % Normalize to unit length
        u = Y*q;                                       % Update u using the linear combination q
        for k = 1 : m,                                 % For each of the k weights/variables
(iii)       x = X(:,k);                                % Select the i^th variable
(iv)        s = sign(t'*x);                            % Identify if x is negative or positive correlated to t
(v)         sx = std(x);                               % Calculate the standard deviation of the i^th variable
(vi)        v = std(t)/sx;                             % Calculate the scaling constant
(vii)       w(k) = s*cor(u, f(s*v*x))*sx;              % Calculate each weight according to criterion
        end                                            % Weight loop
        w = w/norm(w);                                 % Normalize to unit length
        conv = norm(u-uold)/norm(u);                   % Convergence if s < limit (and no. of iterations > max iter.)
    end                                                % Inner loop of iterations (while loop)
```

Algorithm 2.5. The Spline PLS (SPLS) algorithm (MATLAB code).

The method is consistent with the principles of PLS, i.e. it explains both the variance in **X** and **Y**. Moreover, it does not have any problems when applied to underdetermined systems or multicollinearity. However, the focus on simultaneously explaining the variance in the **X** and **Y** matrices may result in an increase in the error when constructing the inner nonlinear mapping, as the "true" underlying structure may not be identified. The error introduced by the nonlinear function between **t** and **u** will be retained in the model.

2.3.5 Alternative Methods

Since the 1980's a number of nonlinear regression methods have been developed including Projection Pursuit Regression (PPR), (Friedman and Stuetzle, 1981 and Friedman, 1985), Classification and Regression Trees (CART), (Breiman *et al,*. 1984), Alternating Conditional Expectation (ACE), (Breiman and Friedman, 1985), Additivity and Variance Stabilization

(AVAS), (Tibshirani, 1987) and Generalized Additive Methods (GAM), (Hastie and Tibshirani, 1990). A tutorial on some of the methods was written by Frank (1995). Of these methods, only projection pursuit regression (PPR) can handle a multivariate response using the inter-relationship between the response variables. As a consequence of its close relationship to the methods of SDPLS, EBPLS and SPLS, it is discussed in more detail.

2.3.5.1 Projection Pursuit Regression.

Projection pursuit regression (PPR), also called SMART (Smooth Multiple Additive Regression Technique), was developed by Friedman and Stuetzle (1981) based on work of Friedman and Tukey (1974). The projection pursuit regression mapping can be written as:

$$\mathbf{Y} = \sum_{j=1}^{A} f_j(\mathbf{X}_j \mathbf{w}_j)\mathbf{q}_j^T + \mathbf{E} \tag{2.41}$$

Determination of the model parameters is carried out by sequentially minimizing a sum-of-squares error function, as in the error based weight updating procedure of Baffi *et al.* (1999a). The main difference being that in PPR, rank-one updating of the matrix **X** is not performed between each optimisation sequence. In addition, the weight updating scheme was based on the direct search method of Rosenbrock (1960), modified to search on a unit sphere and using restarting, i.e. applying a new starting vector to reduce the problem of local minima. Thus the method has good convergence properties and can handle local minima, but should only be applied on overdetermined systems. Furthermore, it may suffer problems when modelling multicollinear systems. This is due to the increased possibility of modelling noise when having collinear variables in the predictor matrix, e.g. when two highly positively correlated variables is allowed be given weights of different sign, the noise will be augmented.

The nonlinear function used is a *smoother*, i.e. a local model is fitted to the data, either using a window or a kernel. In particular, Friedman and Stuetzle (1981) proposed using a local linear kernel regression with a variable bandwidth determined using cross validation, after applying a running median of three as a first pass in the smoother to protect against isolated outliers. Although applying a running median of three makes the inner mapping more robust against outliers, some structural information about underlying function may be lost.

2.3.6 Comparison of the Nonlinear Projection Methods

The weight updating schemes of Wold *et al.* (1999) and Baffi *et al.* (1999a,b) use the methods of Gauss-Newton and steepest descent, respectively, for the sequential minimisation of the sum-of-squares error function, Equation (2.49). An alternative approach is the non-gradient method of Rosenbrock (1960) used in projection pursuit regression. These are typical nonlinear least squares methods, and have a number of issues associated with their application:

(i) If the system is underdetermined, an infinite number of solutions will exist and severe overfitting may occur.

(ii) The lower the signal to noise ratio, the greater the risk of fitting the noise.

(iii) The effect of correlation between the errors (collinearity) is similar to a reduction in the number of observations, since the covariance matrix of the errors will have a higher *condition number*. This will be similar to an error matrix with a smaller number of observations. A well-conditioned problem is when small perturbations of the regression vector lead to only small changes in $f(\mathbf{X}\hat{\boldsymbol{\beta}})$, i.e. a small condition number for inversion.

(iv) When the degree of nonlinearity is low, linear methods can outperform due to their simplicity.

Spline PLS sequentially fits a nonlinear function between a "sub-optimal" pair of latent variables, in the sense that the criterion used does not focus on minimizing the error of the nonlinear mapping, but focuses on explaining the variance in both **X** an **Y**. Consequently, the error introduced by sequentially fitting nonlinear functions to these inner mappings can be large. Furthermore, since **X** and **Y** are rank-one updated using the score vector **t** and the nonlinear fit $\hat{\mathbf{u}}$, respectively (Equation, 2.50), the fitting error propagates to subsequent pairs of latent variables. Consequently, the rank one reduction of **X** and **Y** is not orthogonal ($\mathbf{t}$ is not orthogonal to $\hat{\mathbf{u}}$), thus the error introduced by the nonlinear mapping cannot be modelled by the residual variance in **X**.

Nonlinear PCR addresses the above issues by eliminating the correlation between the errors whilst at the same time increasing the ratio between the number of observations and variables. Issues with nonlinear PCR can occur when the response variables are not explained by the major direction in the predictor matrix, **X**. The models can therefore be less parsimonious than PLS since the information in the response variables is not used in the extraction of the latent variables.

The nonlinear projection methods allow flexibility in the modelling of the relationship between the response and predictor variables, but compared to linear methods there is a greater risk of overfitting. Consequently, in situations where there are only a few observations and/or noisy data, linear methods may outperform nonlinear techniques, even when there is a nonlinear relationship in the data, particularly if the nonlinearities are small, moderate or when an adequate transformation exists. That is, if the nonlinear relationship is known to be exponential, a logarithmic transformation of the response variable results in a linear model to be fitted. However, a logarithmic transformation will also result in a greater emphasis on the lower values that might or might not be of interest, since the variance of the higher values will be decreased relatively more compared to the variance of lower values.

2.4 Nonlinear Mapping of the Inner Relationship

Several measures have been proposed to quantify the level of nonlinearity in nonlinear problems. For a discussion of the various approaches, see Bates and Watts (1980). There are many ways to model a nonlinear relationship. The focus in this Thesis is on the projection based modelling approach defined in Equation (2.41), where the nonlinear mapping is restricted to univariate problems. Nonlinear functions can be divided into two categories: parametric and nonparametric.

2.4.1 Parametric Nonlinear Mapping

The simplest class of parametric functions, and those most frequently used are polynomial functions:

$$\mathbf{u} = \alpha_1 + \alpha_2 \mathbf{t} + \alpha_3 \mathbf{t}^2 + \cdots + \alpha_{r+1} \mathbf{t}^r = \sum_{r=0}^{p} \alpha_{r+1} \mathbf{t}^r \tag{2.55}$$

The key issue is to select the degree of the polynomial, p. Selecting too high an order can result in the data being overfitted, whilst too low a degree will result in the data being underfitted. An alternative nonlinear curve fitting method is the power function:

$$f(\mathbf{t}) = \sum_{i=1}^{g} \beta_i \mathbf{t}^{\alpha_i} \tag{2.56}$$

The power function is reasonable for some problems, but is not the general choice since the coefficient estimates will be correlated and may therefore have a high condition number. A

nonlinear PLS approach that used the error based weight updating procedure and the Box-Tidwell transformation equation, $f(\mathbf{t}) = \beta_0 + \beta_1 \mathbf{t}^{\alpha_1}$, was developed by Li *et al.* (2000).

A third possibility is to fit a curve using a linear combination of known functions. Here any known function, including polynomial and power functions may be used:

$$f(\mathbf{t}) = \sum_{i=1}^{g} \alpha_i f_i(\mathbf{t}), \quad \text{e.g.} \quad f(\mathbf{t}) = \alpha_1 + \alpha_2 \mathbf{t} + \alpha_3 \sin(\mathbf{t}) + \alpha_4 \exp(\mathbf{t}) \tag{2.57}$$

If the underlying structure of the function is known (or is symmetric), the parametric approach is the normal choice, since it generally will give the lowest model error. The nonparametric mapping, does not depend on these conditions and thus is more universal.

2.4.2 Nonparametric Nonlinear Mapping

In parametric regression, it is assumed that the form of the regression function is known (f_j, Equation, 2.41). A more general regression model is obtained by assuming the regression function is unknown. Estimating the nonparametric function may be viewed as applying a "smoother" to the data. Two of the more common forms of nonparametric regression are discussed, kernel regression and smoothing splines.

2.4.2.1 Kernel Regression

A kernel smoother, Bowman and Azzalini (1997) or Simonoff (1996), uses a set of local weights that are defined by a kernel, to generate an estimate of the target value. Typically a kernel smoother uses weights that decrease in a smooth fashion as one moves away from the target point and is typically assumed to be Gaussian. The method calculates the target value using a locally weighted polynomial least squares that is based on the kernel weights. The bandwidth, h, is the smoothing parameter, and it controls the width of the kernel function. The selection of global or local bandwidths is crucial for kernel regression. Too small a bandwidth will lead to overfitting, whilst too large a bandwidth will result in over-smoothing.

Consider a nonparametric regression model between the **u** and **t** scores for the κ^{th} observation:

$$u_\kappa = f_\kappa(t_\kappa) + e_\kappa, \; 1 \le \kappa \le m \tag{2.58}$$

The regression function $f(\mathbf{t}) = \mathrm{E}(\mathbf{u} \mid \mathbf{t})$ can be estimated by a linear form:

$$f_\kappa(t_\kappa) = K(t_k, \mathbf{t}) = \sum_{i=1}^{m} \varpi_{\kappa i} u_i \tag{2.59}$$

where $\varpi_{\kappa i}$ is the weight for the i^{th} observation, used to estimate the function value for the κ^{th} observation as a weighted average of the **u** scores. For a given general kernel function K, and the relationship for the scaling constants: $s^{(r)}$, $0 \le r \le p$ (from the minimization):

$$s_\kappa^{(r)} = \sum_{i=1}^{m} (t_\kappa - t_i)^r K((t_\kappa - t_i)/h) \tag{2.60}$$

the simple Nadaraya-Watson (NW) kernel regression estimator can be defined as:

$$f_{NW}(t_\kappa) = \frac{1}{s_\kappa^{(0)}} \sum_{i=1}^{m} K((t_\kappa - t_i)/h) u_i \equiv \sum_{i=1}^{m} \varpi_{\kappa i} u_i. \tag{2.61}$$

The NW estimator is the solution of a weighted least squares problem:

$$\min_{\beta_0} \sum_{\kappa=1}^{m} \sum_{i=1}^{m} (u_i - \beta_0)^2 K((t_\kappa - t_i)/h) \tag{2.62}$$

and corresponds to a local mean fit. It is possible to extend this approach to a general polynomial regression estimator:

$$\min_{\beta} \sum_{\kappa=1}^{m} \sum_{i=1}^{m} [u_i - \beta_0 - \beta_1(t_\kappa - t_i) - \cdots - \beta_p (t_\kappa - t_i)^p]^2 K((t_\kappa - t_i)/h) \tag{2.63}$$

where p is the degree of the polynomial and $\beta = [\beta_o, \beta_1, \cdots, \beta_p]$ is a vector of coefficients. The solution is found by least squares. The local linear estimator ($p = 1$) can be written as:

$$f(t_\kappa) = \frac{1}{s_\kappa^{(2)} s_\kappa^{(0)} - s_\kappa^{(1)} s_\kappa^{(1)}} \sum_{i=1}^{m} \left[s_\kappa^{(2)} - s_\kappa^{(1)} (t_\kappa - t_i) \right] K[(t_\kappa - t_i)/h] u_i \tag{2.64}$$

For the general solution, let $\mathbf{X}_t$ be the design matrix:

$$\mathbf{X}_t = \begin{bmatrix} 1 & t_\kappa - t_1 & \cdots & (t_\kappa - t_1)^p \\ 1 & t_\kappa - t_2 & \cdots & (t_\kappa - t_2)^p \\ \vdots & \vdots & & \vdots \\ 1 & t_\kappa - t_m & \cdots & (t_\kappa - t_m)^p \end{bmatrix} \tag{2.65}$$

and let,

$$\mathbf{\Omega}_t = diag\left[K(t_\kappa - t_1), \cdots, K(t_\kappa - t_m) \right], \tag{2.66}$$

be the weight matrix, then, if $(\mathbf{X}_t^T\boldsymbol{\Omega}_t\mathbf{X}_t)$ is invertible, the weighted least squares can be calculated (see also Section 2.2.1.2):

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}_t^T\boldsymbol{\Omega}_t\mathbf{X}_t)^{-1}\mathbf{X}_t^T\boldsymbol{\Omega}_t\,\mathbf{u} \tag{2.67}$$

The estimator $f(t_k)$ is then the first element in the $\hat{\boldsymbol{\beta}}$ vector, i.e. $\hat{\boldsymbol{\beta}}(1)$. More generally, $\hat{\boldsymbol{\beta}}(l+1)$ is an estimate of the l^{th} derivative of $f(t_\kappa)$.

Although the general polynomial kernel estimator could be used, the local linear estimator is the most commonly applied, Wand and Jones (1995). The local linear kernel estimator (Equation (2.63), $p = 1$) exhibits better properties at the edges of the fitted function compared with the NW kernel (Equation (2.63), $p = 0$), since the NW kernel does not provide boundary bias correction and tends to "flatten out" at the edges (Simonoff, 1996). The local linear kernel estimator is still sufficiently simple to derive the analytical derivative, Section 2.4.2.1.2. The asymptotic property, i.e. as h becomes large, the fitted curve approaches the linear least squares solution (Bowman and Azzalini, 1997) adds to its applicability.

Total least squares (TLS) can be applied since the method assumes that both the latent variables (**t** and **u**) contain noise. In particular, this assumption can improve the solution for the edges or the endpoints, since these are typically the most problematic areas for kernel regression, but are often more difficult to fit due to ill-conditioning of the matrix inversion involved (Section 2.3.4).

The kernel function used is the Gaussian kernel $K((t_\kappa - t_i)/h) = \exp(-((t_\kappa - t_i)/h)^2/2)$. Other kernel functions are possible but the differences in prediction error in the final models are small, Wand and Jones (1995). It should be noted that nonparametric regression, as for other methods, is sensitive to outliers, especially at the boundaries, thus any outliers should be removed prior to the analysis. However, nonparametric regression is not as sensitive to outliers as other methods such as parametric methods or splines where the model parameters are equally weighted (Simonoff, 1996). A MATLAB function for local linear kernel regression is given in Appendix A2.1.

2.4.2.1.1 The Bandwidth Selector

A number of kernel regression bandwidth approaches have been proposed. The bandwidth approach of Bowmann and Azzalini (1997), termed h_{opt}, minimises the integrated square error

theoretically through a compromise between the bias and the variance of the data, Appendix A2.2. This method results in general lower prediction error for the bandwidth, h_{opt}, than the cross validation method, h_{cv}, (Section 3.2.5.1, leave-one-out), see Cao *et al.* (1994). In the cross validation method, an optimization algorithm is applied that minimizes the cross validation error with respect of the bandwidth.

The same smoothing parameter, h_{opt}, is used for all observations. A locally varying bandwidth, h_i can alternatively be used. It varies directly with the local variance and inversely with the local curvature and local density. Each observation is given a different weight depending on the density of the observations, through a nearest neighbour approach (Simonoff, 1996). In this case, the bandwidth is defined by $h_i = h_{opt}\,(d_k(i)/\text{mean}(d_k))$, where $d_k(i)$ denotes the distance to the k^{th} nearest neighbour of observation i. Local linear and quadratic estimators with bandwidths based on nearest neighbours are often called *loess* (or older term *lowess*) estimators. A MATLAB function is given in Appendix A2.2.

2.4.2.1.2 The Analytical Derivative

As mentioned in Section 2.3.1, the Gauss-Newton method requires the analytical derivative. The Jacobian can be written:

$$\dot{\mathbf{J}} = diag(\partial f(\mathbf{t})/\partial \mathbf{t})\,\mathbf{X} \tag{2.68}$$

Consequently, it is necessary to find the derivative of $\partial f(\mathbf{t})/\partial \mathbf{t}$ where $f(\mathbf{t})$ is given by Equation (2.64) and by the Gaussian kernel $K((t_\kappa - t_i)/h) = \exp(-((t_\kappa - t_i)/h)^2/2)$:

$$f(t_\kappa) = \frac{\sum_{i=1}^{m} w_{\kappa i} u_i}{\sum_{i=1}^{m} w_{\kappa i}} \tag{2.69}$$

where t_κ is the κ^{th} element of $\mathbf{t}$ and given the intermediate calculations

$$w_{\kappa i} = \frac{(s_\kappa^{(2)} - s_\kappa^{(1)} x_{ki})\exp(-\tfrac{1}{2}(x_{\kappa i}/h_\kappa)^2)}{s_\kappa^{(2)} s_\kappa^{(0)} - (s_\kappa^{(1)})^2} \tag{2.70}$$

where

$$x_{\kappa i} = t_i - t_\kappa$$

$$s_\kappa^{(r)} = \sum_i^m \exp(-(x_{\kappa i}/h_\kappa)^2/2)\, x_{\kappa i}^r$$

The first derivative $\partial f(\mathbf{t})/\partial \mathbf{t}$ can be calculated as:

$$\frac{\partial f(t_\kappa)}{\partial t} = -\sum_{i=1}^{m}(g_{\kappa i} + g'_{\kappa i} + g''_{\kappa i})u_i + f(t_\kappa)(\sum_{i=1}^{m} g_{\kappa i} + \sum_{i=1}^{m} g'_{\kappa i} + \sum_{i=1}^{m} g''_{\kappa i}) \tag{2.71}$$

where

$$g_{\kappa i} = \frac{(s_\kappa^{(3)} / {h_\kappa}^2 - s_\kappa^{(1)} - (s_\kappa^{(2)} / {h_\kappa}^2 - s_\kappa^{(0)})x_{\kappa i})\exp(-\tfrac{1}{2}(x_{\kappa i} / h_\kappa)^2)}{s_\kappa^{(2)} s_\kappa^{(0)} - s_\kappa^{(1)} s_\kappa^{(1)}}$$

$$g'_{\kappa i} = \frac{x_{\kappa i} w_{\kappa i}}{{h_\kappa}^2} \tag{2.72}$$

$$g''_{\kappa i} = \frac{(s_\kappa^{(3)} / {h_\kappa}^2 - 2s_{\kappa 1})s_\kappa^{(0)} + s_\kappa^{(2)} s_\kappa^{(1)} / {h_\kappa}^2 - (2s_\kappa^{(1)}(s_\kappa^{(2)} / {h_\kappa}^2 - s_\kappa^{(0)}))}{s_\kappa^{(2)} s_\kappa^{(0)} - s_\kappa^{(1)} s_\kappa^{(1)}} w_{\kappa i}$$

An algorithm that combines the construction of the local kernel function and its first analytical derivative is given by the MATLAB function, Appendix A2.3.

2.4.2.2 Smoothing Splines

Smoothing splines is a nonparametric method that has similarities to the kernel regression method. Two criteria are used to select a nonparametric smoothing spline function:

(i) Goodness of fit

(ii) Smoothness of fit

A standard measure for the goodness of fit is the mean residual sum of squares:

$$\frac{1}{m}\sum_{i=1}^{m}(u_i - f(t_i))^2 \tag{2.73}$$

where m is the number of observation. A common measure of the smoothness of fit is the integrated squared second derivative:

$$\int_{-\infty}^{\infty}(\nabla^2 f(\mathbf{t}))^2 \,\partial \mathbf{t} \tag{2.74}$$

where $\nabla^2 f(\mathbf{t})\partial \mathbf{t}$ is the second derivative of $f(\mathbf{t})$ with respect to $\mathbf{t}$ and f belongs to the set of all continuously differentiable functions with square intergrable derivatives. A single criterion that combines the two criteria is given by:

$$\gamma\,(\mathbf{u}-f(\mathbf{t}))^{\mathbf{T}}(\mathbf{u}-f(\mathbf{t}))+(1-\gamma)\int_{-\infty}^{\infty}(\nabla^2 f(\mathbf{t}))^2\,\partial\mathbf{t} \tag{2.75}$$

where $\gamma \in (0, 1)$ is the smoothing parameter. The most common class of estimators uses a cubic polynomial for f (Simonoff, 1996), and is discussed in more detail in the subsequent section.

2.4.2.2.1 Cubic Smoothing Splines

This smoother is based on the solution to an optimization problem. Consider the following problem. Amongst all functions $f(\mathbf{t})$ with η continuous derivatives a solution that minimizes the penalized residual sum of squares exist:

$$\gamma\sum_{i=1}^{m}\{u_i - f(t_i)\}^2 + (1-\gamma)\int_a^b \{\nabla^\eta f(\mathbf{t})\}^2\,dt \tag{2.76}$$

where γ is a fixed constant and $a \le t_1 \le \cdots \le t_m \le b$. The integral boundary defined by a and b in Equation (2.76) is defined to capture the data. The most common form of η, is where $\eta = 2$ so that it becomes square integrable. It can be shown that Equation (2.76) has an unique minimizer that is a natural cubic spline with knots at unique values of t_i. The first term measures "closeness to the data", whilst the second term penalizes curvature in the function. It may appear that the spline is overparameterized since there are as many as $(m-2)$ interior knots. However, since the coefficients are estimated in a constrained manner by the second term, the effective dimension can be reduced considerably.

The parameter γ plays the same role as the bandwidth in kernel regression. Small values of γ produces smoother curves, whilst larger values are liable to result in overfitting. At one extreme as $\gamma \to 0$, the penalty term dominates, forcing $\nabla^2 f(\mathbf{t}) = 0$ everywhere, and thus the solution tends towards least squares (linear). At the other extreme, as $\gamma \to 1$ the penalty term becomes unimportant and the solution tends to an interpolating natural cubic spline, since it is constructed to have zero second and third derivatives at the boundaries (the natural boundary conditions).

2.4.2.2.2 Computational Aspects

Using the fact that the solution to Equation (2.76) is a natural cubic spline with $(m-2)$ interior knots, it can be represented in terms of a basis for this space. For computational convenience, the unconstrained B-spline basis is used. The m equations required to determine $f(t_1),\cdots,f(t_m)$ are obtained by minimizing Equation (2.76). Let $\mathbf{R}$ be the $((m-2)$ x $(m-2))$

symmetric tridiagonal matrix with diagonal elements $R_{i,i} = 2(t_{i+1} - t_{i-1})$ and off-diagonal elements $R_{i,i-1} = R_{i,i+1} = (t_{i+1} - t_i)$. Furthermore, let **Q** be the $((m-2)$ x $m)$ upper tridiagonal matrix with $Q_{i,i} = Q_{i-1,i+1} = 1/(t_{i+1} - t_i)$, $Q_{i,i+1} = -(1/(t_{i+1} - t_i) + 1/(t_{i+2} - t_{i+1}))$, with 0 elsewhere. It is shown (de Boor, Chapter XIV.6, 1978) that the m additional equations can be estimated as:

$$[6(1-\gamma)\mathbf{Q}\mathbf{Q}^{\mathrm{T}} + \gamma\mathbf{R}]\mathbf{v} = \partial\mathbf{u}/\partial\mathbf{t} \tag{2.77}$$

where the estimate $f(\mathbf{t})$ of $\mathbf{u}$ is given by

$$f(\mathbf{t}) = \mathbf{u} - 6(1-\gamma)\partial[0;\partial[0;\mathbf{v};0]/\partial\mathbf{t};0] \tag{2.78}$$

The cubic smoothing spline, because of the implicit way it is defined, does not use local weighted averaging as for kernel regression. However, it possesses local behaviour similar to that of kernels. It has been shown, Simonoff (1996), that the smoothing spline can be written as a kernel smoother. The cubic smoothing splines were fitted using the MATLAB program defined in Appendix A2.4.

2.4.3 Comparison

The effectiveness of a number of the function estimation approaches discussed in the previous section is demonstrated on 6 simulated examples, see also Cao *et al.*, (1994). The comparison was conducted for three noise levels and for three sample sizes. A selection of methods for function evaluation were compared. These methods were (Table 2):

Methods
1. Smoothing spline, smoothing parameter estimated using cross validation (CV).
2. Local linear kernel regression using optimal bandwidth and loess.
3. Local linear kernel regression, the smoothing parameter, *h*, estimated using CV.
4. Local linear kernel regression, using total least squares, *h*, estimated using CV.
5. Local cubic kernel regression, the smoothing parameter, *h*, estimated using CV.
6. Parametric regression, using the known function (reference).

Table 2. Methods for function evaluation.

Figure 2.4 shows six simulated functions with the true underlying function superimposed (red). The underlying functions were: (i) $\mathbf{u} = a_1 \sin(\mathbf{t}) + a_2\mathbf{t}$, (ii) $\mathbf{u} = a_1\mathbf{t}^2$, (iii) $\mathbf{u} = a_1 \exp(\mathbf{t})$, (iv) $\mathbf{u} = a_1 \tanh(\mathbf{t})$, (v) $\mathbf{u} = a_1 abs(\mathbf{t}) + a_2\mathbf{t}$ and (vi) $\mathbf{u} = a_1\mathbf{t}$. The univariate data consists of m = 25, 50 or 100 observations that were generated from these functions, for three signal to noise ratios

(SNR), i.e. low: SNR = 100, medium: SNR = 10 and high: SNR = 2. Additive Gaussian noise was added to both variables (**t** and **u**).

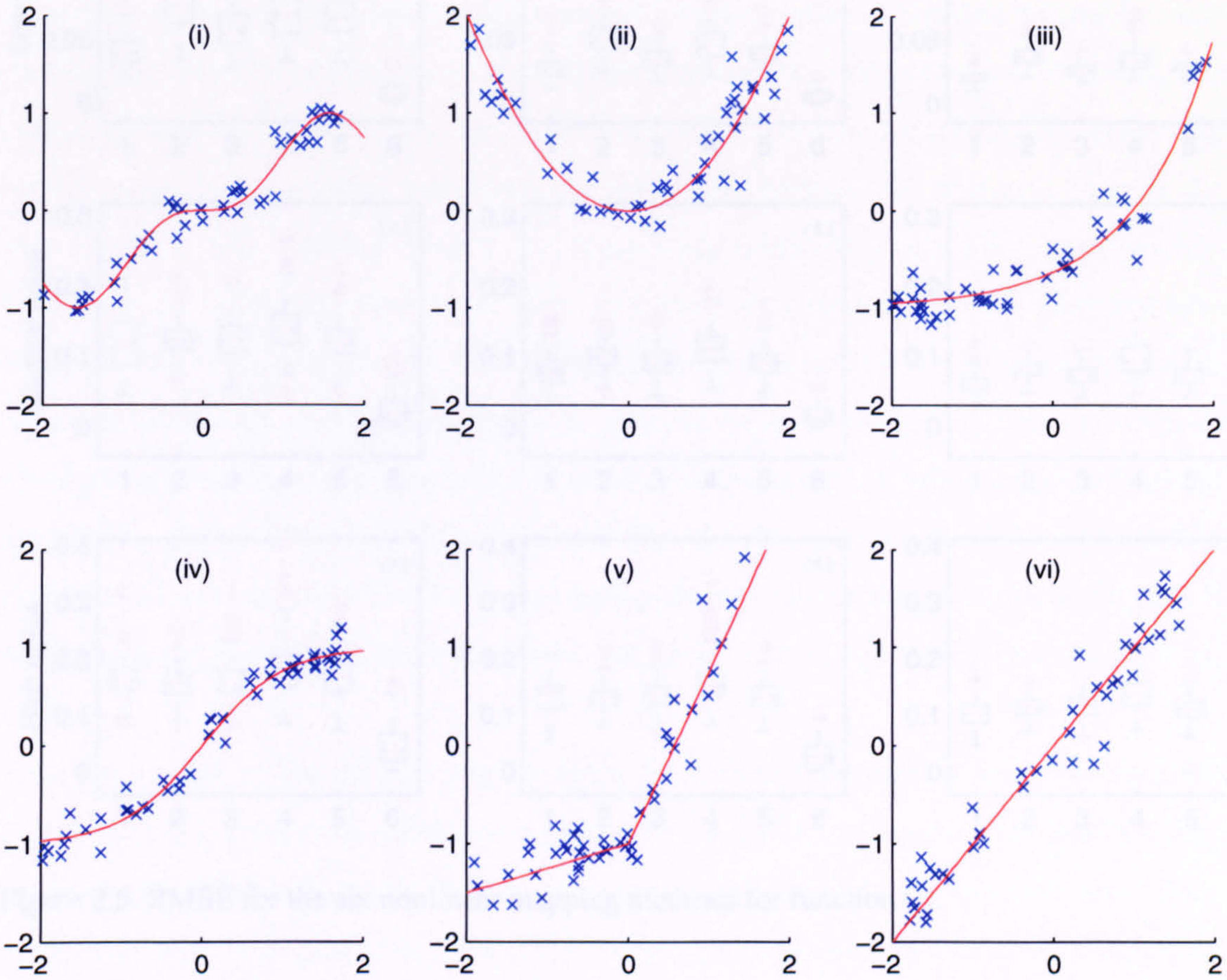

Figure 2.4. Six simulated examples, based on 50 samples and exploiting medium noise.

The underlying functions were then applied to the noisy data using the six different methods. For each case, the distribution of 500 repetitions was examined to avoid random instances when interpreting the outcome of the simulation. The results are presented in Figure 2.5. Since the true underlying functions were known from the simulation, the error was measured as the root mean squared error (RMSE) between the estimated function and the true function. Furthermore, by applying parametric regression derived from the reference function, the lower limit of RMSE is indicated.

There was a high level of agreement between the results for the six different functions illustrated, in Figure 2.4. Thus, only the results from the first curve (i) in Figure 2.4 are discussed, with the distribution results given in Figure 2.5. Based on the estimated Root Mean Squared Error (RMSE) values of 500 repetitions, the median, the first and fourth quartile, and the points not encapsulated by the first and fourth quartile were plotted in a box plot represented for each of the six function estimation methods.

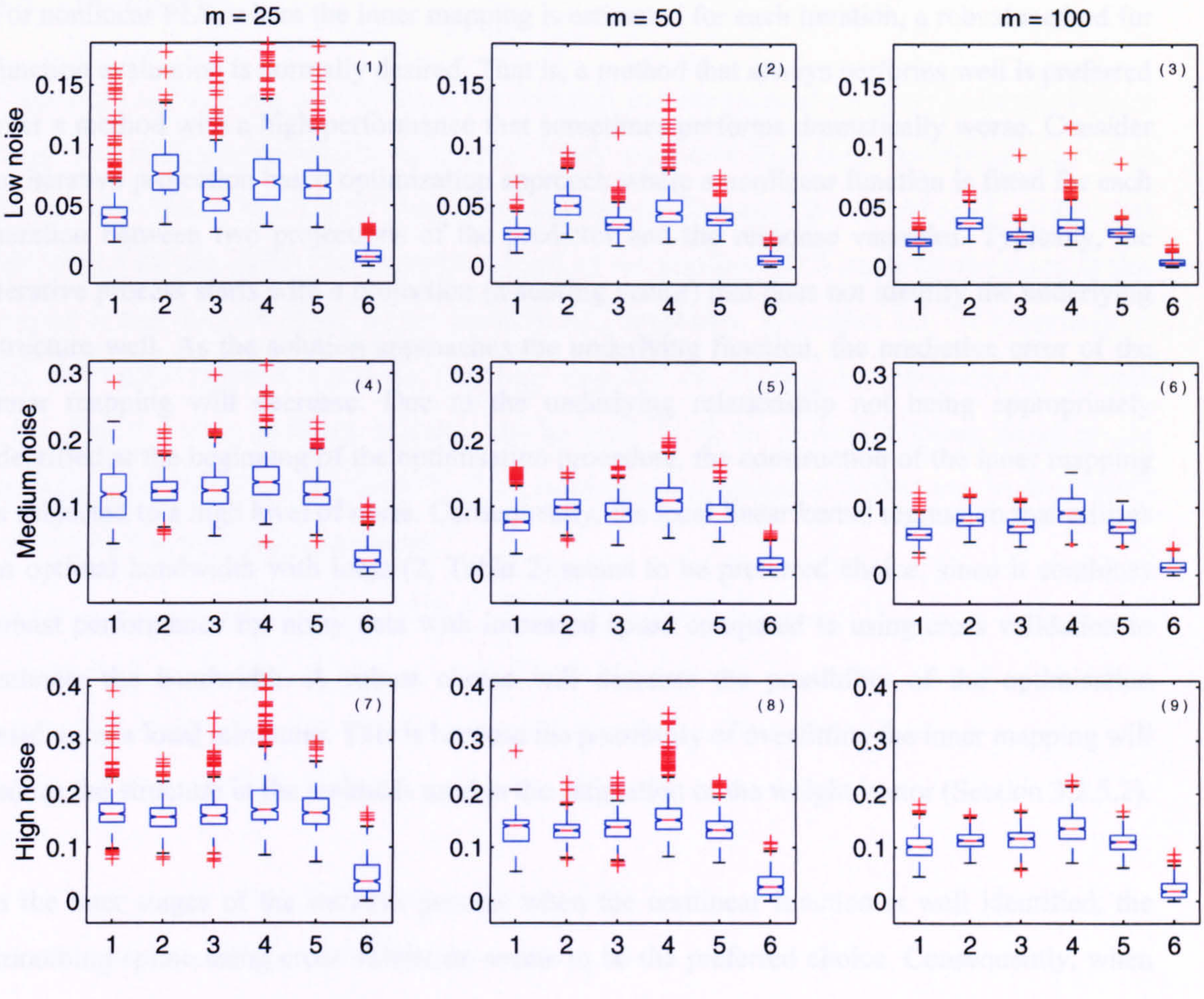

Figure 2.5. RMSE for the six nonlinear mapping methods for function (i).

From Figure 2.5, the features of the different methods are.

(i) If the function is known, parametric regression (6, Table 2) is the best method.

(ii) The difference between the performance of the nonparametric methods was not generally large, especially as the number of observations increased.

(iii) For low noise, cross validation was better than the optimal bandwidth approach.

(iv) For low noise, the smoothing spline (1, Table 2) performed the best of the nonparametric methods, and for large data sets was independent of noise level.

(v) For high noise and small and medium sized data sets, local linear kernel regression using the optimal bandwidth (2, Table 2) exhibited best performance for the nonparametric methods.

(vi) Local cubic kernel regression (5, Table 2) showed improved performance for an increasing number of observations, but local linear kernel regression generally performed better.

For nonlinear PLS, where the inner mapping is estimated for each iteration, a robust method for function evaluation is normally desired. That is, a method that always performs well is preferred over a method with a high performance that sometimes performs dramatically worse. Consider an iterative projection based optimization approach where a nonlinear function is fitted for each iteration between two projections of the predictor and the response variables. Typically, the iterative process starts with a projection (a starting vector) that does not identify the underlying structure well. As the solution approaches the underlying function, the predictive error of the inner mapping will decrease. Due to the underlying relationship not being appropriately identified at the beginning of the optimisation procedure, the construction of the inner mapping is subjected to a high level of noise. Consequently, the local linear kernel regression that utilizes an optimal bandwidth with loess (2, Table 2) seems to be preferred choice, since it combines robust performance for noisy data with increased speed compared to using cross validation to estimate the bandwidth. A robust choice will decrease the possibility of the optimisation residing in a local minimum. This is because the possibility of overfitting the inner mapping will reduce the structure in the residuals used in the estimation of the weight vector (Section 3.2.5.2).

In the later stages of the iterative process when the nonlinear function is well identified, the smoothing spline using cross validation seems to be the preferred choice. Consequently, when the parametric function is not known, the recommendation would be either to use local linear kernel regression or to start with local linear kernel regression and then switch to smoothing splines at the later stages of the optimization. Alternatively, if the parametric function is known, the recommendation would be to use that function for the inner mapping.

The difference between a smoother and a spline function is that although the final function appears continuous, there is no continuity restriction in fitting the local polynomials in kernel regression. In the smoothing spline, the first and second derivatives are assumed to be equal. However, kernel regression is easier to interpret, since it generalises the most commonly used statistical method, least squares, to allow local nonlinearity (Simonoff, 1996). Spline estimators place the smoothing problem in the framework of optimizing a penalized version of the likelihood. Historically, a strong argument for this roughness penalty approach over kernel estimators was the simplicity with which they could generalize from the least squares criterion to likelihood functions, Simonoff (1996). Ultimately, the choice between the two nonparametric methods comes down to the ease by which the smoothing parameter is found. This favours the plug-in method related to the local linear kernel regression since no cross validation is necessary, Bowmann and Azzalini (1997).

2.5 Discussion

A general overview has been given of the concepts and the tools of some of the more common linear and nonlinear regression techniques. Three nonlinear PLS methods were discussed in detail, Error Based PLS (Baffi *et al.*, 1999a), Steepest Descent PLS (Wold *et al.*, 1989), and Spline PLS (Wold, 1992). An important aspect of the nonlinear PLS methods discussed in this chapter is that they are based on the classical PLS algorithm, Algorithm 2.2. Of importance is that the weight estimation and the nonlinear mapping are constructed independently of each other, thus the same nonlinear mapping can be used for all the three algorithms. Based on the limitations described, the nonlinear PLS approaches are investigated further and improvements are proposed in the following chapter.

CHAPTER 3

DETERMINING THE WEIGHTS IN NONLINEAR PARTIAL LEAST SQUARES

3.1 Introduction

No particular nonlinear PLS method has dominated the field of chemometrics. Although shown to work for certain applications, they all have limitations. Furthermore, the issues associated with the different methods have not been explored in detail. Fitting a univariate nonlinear function $f(\mathbf{t})$ between the scores **t** and **u** is not difficult when the underlying structure is properly identified, thus the most important requirement in nonlinear PLS is to determine the weight vector **w** that identifies the underlying structure in an appropriate manner. The underlying structure may or may not be known. If it is not known the inner mapping can be verified using a separate validation set. It should be noted that the Y-score vector, $\mathbf{u} = \mathbf{Yq}$, is identified separately from the X-score vector, $\mathbf{t} = \mathbf{Xw}$, and is calculated equally for all the NIPALS based PLS methods discussed in the Thesis.

The aim of the work presented in this Chapter is to establish a general nonlinear PLS algorithm. In addition, theoretical and practical issues concerning nonlinear PLS algorithms are discussed. Specifically, two fundamentally different ideas as to how to implement nonlinear PLS are investigated, Framework 1 and 2. Both frameworks are independent of the type of nonlinear function that is used to model the inner relationship (between **t** and **u**), and differ only in terms of how the weight vector **w** is obtained.

3.1.1 Framework 1

In 1989, Wold *et al.* introduced the concept of nonlinear PLS whereby an objective function is minimized through an updating scheme, Steepest Descent PLS (SDPLS). This scheme minimises Equation (2.49) using the method of Steepest Descent (SD) that was discussed in Section 2.3.4.1. In 1999, Baffi *et al.* modified this algorithm by replacing the SD method by that of Gauss-Newton (Section 2.3.4.2), Error Based PLS (EBPLS). Based on these ideas, a new weight updating scheme is proposed, Nested PLS (NPLS). In NPLS, the weight updating is implemented by means of linear PLS with cross validation being used to minimize the expression in Equation (2.50), i.e. solving $\partial\mathbf{w}$ from the linear Taylor expansion $\mathbf{e} \approx -\mathbf{J}\partial\mathbf{w}$,

$$\partial \mathbf{w} = \mathrm{PLScv}(-\mathbf{J}, \mathbf{e}) \tag{3.1}$$

where $\partial \mathbf{w}$ represents the regression coefficients obtained from the inner PLS model. Fundamentally, steepest descent (SDPLS) can be seen as applying Equation (3.1) using only one latent variable whilst Gauss-Newton (EBPLS) can be seen as applying Equation (3.1) using all the latent variables, i.e. least squares. Nested PLS lies between these two extreme cases, with the number of latent variables selected using cross validation.

3.1.2 Framework 2

In 1992, Wold proposed a different approach based on the covariance criterion, Spline PLS (SPLS). The covariance criterion was adapted for use in the nonlinear case, by linearizing each variable through a nonlinear function fitted prior to calculating the covariance as described in Section 2.3.4.3. Although the method is appealing since it is closely related to the concept of linear PLS, it has a major limitation. This criterion does not focus on minimizing the error between $\mathbf{u}$ and $f(\mathbf{t})$, but on explaining both the variance in $\mathbf{X}$ and $\mathbf{Y}$. Consequently, the risk of introducing error when fitting the nonlinear function is larger than for Framework 1.

Based on the SPLS framework a new criterion is proposed, the reciprocal error variance criterion that results in Reciprocal Variance PLS (RVPLS). RVPLS calculates each weight independently as for the covariance criterion, but focuses on explaining the variance of the response variables only.

The advantage of linear PLS, i.e. the removal of information on a step-by-step basis until only noise is contained in the residuals, is not directly applicable in nonlinear PLS. It is essential that the underlying structure is identified, i.e. the selection of appropriate $\mathbf{w}$ and $\mathbf{q}$ vectors, that minimizes the error when fitting a nonlinear function between the $\mathbf{t}$ and $\mathbf{u}$ scores. By fitting a nonlinear function for each latent variable on a step-by-step basis, each nonlinear function will introduce an error that propagates to the next latent variable. It will then not be possible to model the resulting error using the predictor matrix $\mathbf{X}$, and this will influence the subsequent latent variable as the error caused by the inner mapping will be included in the residuals. Thus, it is desirable to minimize the number of latent variables used in the model by identifying the underlying nonlinear relationship in the best possible manner. Specifically, identifying the first latent variable is the most important, as this often determines the overall performance of the algorithm.

3.2 The Nested PLS Algorithm

The Nested PLS method is based on the Error Based Weight Updating (EBWU) procedure of Baffi *et al.* (1999a,b). The method minimizes the number of latent variables as discussed in the earlier section, but as it is a nonlinear least squares method, there will be issues when multicollinearity is present between the variables or when the system is underdetermined. One proposed solution to overcoming these problems is to use EBPLS as a nonlinear PCR method. That is, apply regression to the orthogonal score matrix **T**. However, this approach may not give a satisfactory model as discussed previously.

To overcome the collinearity limitation of the EBWU procedure, a new methodology is proposed, Nested PLS (NPLS), which comprises an inner and outer PLS algorithm (Li *et al.*, 2001). The objective of the outer algorithm is to extract those building blocks, such as latent variables and loadings, **t**, **u**, **p**, and **q** that will form the basis of the final application. The role of the inner algorithm is to overcome the multicollinearity problem of the EBWU procedure and to derive the weight vector **w** for the outer PLS algorithm. The concept of Nested PLS is illustrated in Figure 3.6.

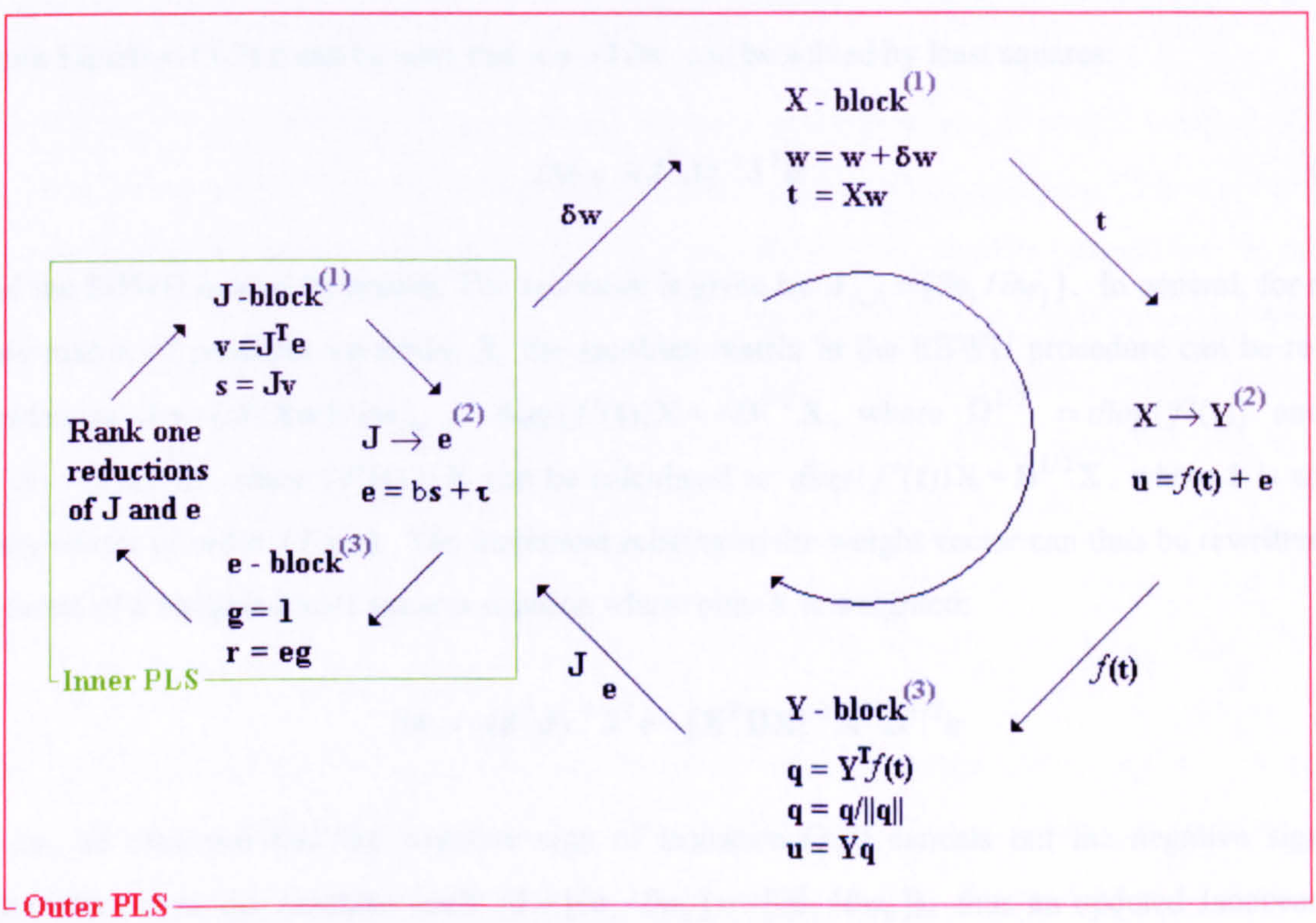

Figure 3.6. Simplified illustration of the Nested nonlinear PLS algorithm

Considering the Taylor series approximation defined in Equation (2.50) ($\mathbf{e} = -\mathbf{J}\partial\mathbf{w}$), the inner PLS algorithm is used to identify the regression coefficients, $\partial\mathbf{w}$, for the linear regression problem where **J** are the predictor variables and **e** is the response variable using linear PLS. The

number of latent variables is determined using cross validation. Step (1) in Figure 3.6, results in the calculation of the score vectors $\mathbf{t} = \mathbf{Xw}$ and $\mathbf{s} = \mathbf{Jv}$, for the outer and inner PLS algorithms, respectively. Step (2) identifies the nonlinear function, $f(\mathbf{t})$, and the linear function, $b\mathbf{s}$, between the score spaces, for the outer (**t**, **u**) and inner (**s**, **r**) score spaces respectively. Step (3) defines the loading vectors **q** and **g** for the responses **Y** and **e**, for the outer and inner PLS algorithms respectively. Since the error vector **e** represents an univariate response, i.e. $\mathbf{g} = 1$, the score vector **r** becomes **e**, consequently the inner PLS algorithm is non-iterative. The algorithm is described in more detail in the subsequent sections.

3.2.1 The Multicollinearity Problem

As discussed in Section 2.3.1, Gauss-Newton optimization is based on the Taylor series expansion of $f(\mathbf{Xw})$ about the current weight vector **w**. Consequently the mismatch, **e**, between $\mathbf{u} = \mathbf{Yq}$ and $\hat{\mathbf{u}} = f(\mathbf{Xw})$ can be defined as:

$$\mathbf{e} = \mathbf{u} - \hat{\mathbf{u}} \approx -\mathbf{J}\,\partial\mathbf{w} \tag{3.2}$$

From Equation (3.2) it can be seen that $\mathbf{e} \approx -\mathbf{J}\,\partial\mathbf{w}$ can be solved by least squares:

$$\partial\mathbf{w} = -(\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T\mathbf{e} \tag{3.3}$$

and the EBWU procedure results. The Jacobean is given by $\mathbf{J}_{(i,j)} = [\partial e_i / \partial w_j]$. In general, for a data matrix of predictor variables, **X**, the Jacobian matrix in the EBWU procedure can be re-written as $\mathbf{J} = -[\partial f(\mathbf{Xw})/\partial \mathbf{w}]_{\mathbf{w}} = -diag\{f'(\mathbf{t})\}\mathbf{X} = -\mathbf{D}^{1/2}\mathbf{X}$, where $\mathbf{D}^{1/2} = diag\{f'(\mathbf{t})\}$ and $f'(\mathbf{t}) = \partial f(\mathbf{t})/\partial \mathbf{t}$, since $(f'(\mathbf{t})\mathbf{1})\mathbf{X}$ can be calculated as $diag\{f'(\mathbf{t})\}\mathbf{X} = \mathbf{D}^{1/2}\mathbf{X}$, where **1** is an unity vector of order $(1 \times n)$. The increment relating to the weight vector can thus be rewritten in terms of a weighted least squares solution where only **X** is weighted:

$$\partial\mathbf{w} = -(\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T\mathbf{e} = [\mathbf{X}^T\mathbf{D}\mathbf{X}]^{-1}\mathbf{X}^T\mathbf{D}^{1/2}\mathbf{e} \tag{3.4}$$

It can be observed that the negative sign of Equation (3.3) cancels out the negative sign originating from the Jacobian itself ($\mathbf{J} = [\partial e_i / \partial w_k] = -[\partial \hat{u}_i / \partial w_k]$), thus an updated Jacobean can be defined, ($\dot{\mathbf{J}} = -\mathbf{J} = \mathbf{D}^{1/2}\mathbf{X}$) to simplify the following discussions.

The condition number is typically used to characterise the severity of the multicollinearity problem. From Equation (3.4), the condition number of the EBWU procedure is closely related to the condition number of **X**. Therefore, the EBWU procedure can be affected by

multicollinearity just as ordinary least squares. In particular, for the linear inner mapping, $f(\mathbf{s}) = b\mathbf{s}$ where b is a scalar, the condition number for problem (3.4) is equal to the condition number of $\mathbf{X}$.

If the matrix, $\mathbf{J}$ is rank deficient, the pseudo inverse $(\mathbf{J}^T\mathbf{J})^+$ of the matrix $(\mathbf{J}^T\mathbf{J})$ is required in Equation (3.4). It should be noted that replacing the inverse of matrix $(\mathbf{J}^T\mathbf{J})$ by a generalized inverse, for example, the Moore-Penrose inverse, $(\mathbf{J}^T\mathbf{J})^+$ will not work if $\mathbf{X}$ is ill-conditioned. It only works if $\mathbf{X}$ is of full column-rank since in this case, $(\mathbf{J}^T\mathbf{J})^- = (\mathbf{J}^T\mathbf{J})^{-1}$ for any generalized inverse of $(\mathbf{J}^T\mathbf{J})$. When $\mathbf{X}$ is not of column rank, the Moore-Penrose inverse can handle the problem of zero singular values of $\mathbf{X}$, but not those singular values that are very small but not strictly zero.

3.2.2 The Inner PLS Algorithm

Consider the following linear regression problem from Equation (3.2):

$$\mathbf{e} = \dot{\mathbf{J}}\partial\mathbf{w} + \boldsymbol{\tau} \tag{3.5}$$

where $\boldsymbol{\tau}$ is the inner error vector, $\partial\mathbf{w}$ is the unknown regression parameter vector, $\mathbf{e}$ and $\dot{\mathbf{J}}$, are the observation vectors for the response variable and design matrix respectively. This is a linear regression problem that can be solved using PLS, and denotes the inner PLS algorithm. In the inner PLS algorithm, standard PLS replaces the least squares solution in Equation (3.5), where $\dot{\mathbf{J}}$ is treated as the X-block and $\mathbf{e}$ the Y-block in linear PLS. Specifically, weight vectors $\mathbf{v}_\ell$ $(k = 1, \cdots, A)$ are sought for the inner PLS algorithm (Wold, 1966) such that the data matrices, $\dot{\mathbf{J}}$ and $\mathbf{e}$, can be decomposed into the following sum of outer products:

$$\dot{\mathbf{J}} = \sum_{\ell=1}^{A} \mathbf{s}_\ell \mathbf{h}_\ell^T + \dot{\mathbf{J}}_{A+1} \qquad \mathbf{e} = \sum_{\ell=1}^{A} \hat{b}_\ell \mathbf{s}_\ell + \mathbf{e}_{A+1} \tag{3.6}$$

$\dot{\mathbf{J}}_{A+1}$ and $\mathbf{e}_{A+1}$ are the residuals, A is determined by cross-validation and $\mathbf{s}_\ell = \dot{\mathbf{J}}_\ell \mathbf{v}_\ell$ are the latent variables of the inner PLS algorithm. The data matrices, $\dot{\mathbf{J}}$ and $\mathbf{e}$, are then updated as follows:

$$\begin{aligned} \dot{\mathbf{J}}_{\ell+1} &= \dot{\mathbf{J}}_\ell - \mathbf{s}_\ell \mathbf{h}_\ell^T \text{ with } \dot{\mathbf{J}}_1 = -\mathbf{J} \\ \mathbf{e}_{\ell+1} &= \mathbf{e}_\ell - \hat{b}_\ell \mathbf{s}_\ell \text{ with } \mathbf{e}_1 = \mathbf{e} \end{aligned} \tag{3.7}$$

where $\mathbf{h}_\ell = \dot{\mathbf{J}}_\ell^{\mathrm{T}}\mathbf{s}_\ell / (\mathbf{s}_\ell^{\mathrm{T}}\mathbf{s}_\ell)$ are the loading of the X-block and $\hat{b}_\ell = \mathbf{e}_\ell^{\mathrm{T}}\mathbf{s}_\ell / (\mathbf{s}_\ell^{\mathrm{T}}\mathbf{s}_\ell)$ is the estimate of the regression coefficient of the inner mapping between the X-block and the Y-block. The Y-block is a single column, resulting in the loading vector, $\mathbf{g}_\ell$ and latent variable vector, $\mathbf{r}_\ell$, of the Y-block being unity and $\mathbf{e}_\ell$ respectively. Thus the computation of extracting each of the latent variables, $\mathbf{v}_\ell$, is not iterative and thus is fast:

$$\mathbf{v}_\ell = \dot{\mathbf{J}}_\ell^{\mathrm{T}}\mathbf{e}_\ell / (\mathbf{e}_\ell^{\mathrm{T}}\dot{\mathbf{J}}_\ell\dot{\mathbf{J}}_\ell^{\mathrm{T}}\mathbf{e}_\ell)^{1/2} \tag{3.8}$$

The output of the inner PLS algorithm is a PLS based estimate (the regression coefficient) of the unknown parameter vector, $\partial\mathbf{w}$, given by $\partial\hat{\mathbf{w}} = \mathbf{V}(\mathbf{H}^{\mathrm{T}}\mathbf{V})^{-1}\mathbf{b}$, where $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_A]$, $\mathbf{H} = [\mathbf{h}_1, ..., \mathbf{h}_A]$, and $\mathbf{b} = [b_1, ..., b_A]^{\mathrm{T}}$, similar to that given in Equation (2.36).

3.2.3 The Outer PLS Algorithm

The basic building blocks of the outer PLS algorithm are calculated based on the weight vectors $\mathbf{w}$ calculated from the inner PLS algorithm, where $\mathbf{X}$ is treated as the X-block and $\mathbf{Y}$ the Y-block. More specifically, based on the updated weight vector, $\mathbf{w}_{l+1} = \mathbf{w}_l + \partial\mathbf{w}$ and $\mathbf{w} = \mathbf{w} / \|\mathbf{w}\|$, $\mathbf{X}$ and $\mathbf{Y}$ can be decomposed into the following sums of outer products:

$$\mathbf{X} = \sum_{j=1}^{A}\mathbf{t}_j\mathbf{p}_j^{\mathrm{T}} + \mathbf{X}_{A+1} \text{ and } \mathbf{Y} = \sum_{j=1}^{A} f(\mathbf{t}_j)\mathbf{q}_j^{\mathrm{T}} + \mathbf{Y}_{A+1} \tag{3.9}$$

where A may be determined by cross validation, the latent variables $\mathbf{t}_j$ and $\mathbf{u}_j$ are given by $\mathbf{t}_j = \mathbf{X}_j\mathbf{w}_j$, $\mathbf{u}_j = \mathbf{Y}_j\mathbf{q}_j$ and $\mathbf{p}_j = \mathbf{X}_j^{\mathrm{T}}\mathbf{t}_j / (\mathbf{t}_j^{\mathrm{T}}\mathbf{t}_j)$.

For each latent variable of the outer PLS algorithm, the error between the predicted and calculated latent variable, $\mathbf{e}_j = \mathbf{u}_j - \hat{\mathbf{u}}_j$, and the Jacobian matrix, $\dot{\mathbf{J}}_j = \partial f(\mathbf{X}_j\mathbf{w}_j)/\partial\mathbf{w}_j$, is obtained for each iteration of the outer PLS algorithm. This is then used in the inner PLS algorithm.

After convergence, rank one updating is undertaken according to Equation (3.9), and if required the subsequent latent variable is estimated. The only difference compared to EBPLS is that ordinary least squares in line (v) of Algorithm 2.4 is replaced by $\partial\mathbf{w}$ = PLScv($\dot{\mathbf{J}}$, $\mathbf{e}$), i.e. the updating vector, $\partial\mathbf{w}$, is calculated from the regression vector calculated from linear PLS where the number of latent variable is selected using cross validation (CV). Algorithm 3.1 gives the key algorithmic steps in the Nested PLS algorithm.

```
...                                                    ...
    while conv > limit,                                % Repeat until convergence
        uold = u;                                      % Retain the old Y score vector to restrain the convergence
        t = X*w;                                       % Calculate the X score vector
(i)     [û, dûdt] = f(t, u);                           % Calculate the nonlinear fit, and the 1st derivative
(ii)    q = Y'*û;                                      % Regress Y on û (not on u as in linear PLS)
        q = q/norm(q);                                 % Normalize to unit length
        u = Y*q;                                       % Update u using the linear combination q
(iii)   e = u - û;                                     % Calculate the error vector e
(iv)    J = X.*[dûdt(:, ones(1, xcol))];               % Calculate the Jacobian (sign cancels in (v))
(v)     dw = PLScv(J, e);                              % Calculate the weight update vector using PLS with CV
(vi)    w = w + dw;                                    % Improved weight vector
        w = w/norm(w);                                 % Normalize to unit length
        conv = norm(u-uold)/norm(u);                   % Convergence if s < limit (and no. of iter. > max iter.)
    end                                                % Inner loop of iterations (while loop)
```

Algorithm 3.1. The Nested PLS algorithm (NPLS) (MATLAB code).

3.2.4 Special Cases

3.2.4.1 The Error Based PLS Algorithm

When the number of latent variables is equal to the number of columns of the **X** matrix, the PLS algorithm gives the same solution as that of least squares. Thus, if the number of latent variables in the inner PLS model is equal to the number of columns in the Jacobian matrix, **J**, the inner PLS algorithm reduces to a least squares problem, Equation (3.3), and the Nested PLS approach collapses to the Error Based Weight Updating (EBWU) approach based on the nonlinear PLS algorithm of Baffi *et al.* (1999a). Hence, the Gauss-Newton approach in EBWU is a special case of the inner PLS algorithm in the Nested PLS algorithm. In general, if a data matrix **X** is not ill-conditioned, both algorithms will give the same performance.

3.2.4.2 The Steepest Descent PLS Algorithm

For the case where the number of latent variables retained in the inner PLS algorithm of the Nested PLS algorithm is unity, the algorithm is closely related to that developed by Wold *et al.* (1989). For the inner mapping:

$$\mathbf{u} = f(\mathbf{t}) + \mathbf{e} = f(\mathbf{X}\mathbf{w}) + \mathbf{e} \tag{3.10}$$

Expanding $\hat{\mathbf{u}} = f(\mathbf{X}\mathbf{w})$ using the Taylor's series expansion as in Equation (2.50):

$$\mathbf{u} \approx \hat{\mathbf{u}} - \mathbf{J}\,\partial\mathbf{w} \tag{3.11}$$

The increment of the weight vector, $\partial\mathbf{w} = \mathbf{w}_{i+1} - \mathbf{w}_i$, is calculated using linear PLS with one latent variable retained. This is the same as the Steepest Descent (SD) approach discussed in Section 2.3.1. Furthermore, the method of SD starts with rather rapid convergence but then slows down, whilst Gauss-Newton has just the opposite effect, i.e. it starts out slowly and ends up with rapid convergence. Applying a method such as Nested PLS, that is a balance between the two approaches, may result in more effective convergence.

3.2.4.3 Linear PLS

Consider the situation where there is a single response variable. When the inner mapping is a linear function, Nested PLS reduces to linear PLS. Consider the first iteration of the outer PLS algorithm. Theoretically, when the inner mapping in the outer PLS algorithm is a linear function, $f(\mathbf{t}) = b\,\mathbf{t}$, where b is a scalar, the Jacobian matrix in the inner PLS algorithm, $\mathbf{J}_1 = \partial f(\mathbf{X}_1\mathbf{w}_1)/\partial\mathbf{w}_1$, reduces to $\mathbf{J}_1 = b_1\,\mathbf{X} = b_1\,\mathbf{X}$, and the error, $\mathbf{e}_1 = \mathbf{u}_1 - \hat{\mathbf{u}}_1$, reduces to $\mathbf{Y}$ if the initial vector, $\mathbf{w}_0$, is taken as $\mathbf{0}$ (note that $\hat{\mathbf{u}}_1$ is a constant in this case). Then for the inner PLS model, the increment of the weight vector is given by, $\partial\mathbf{w}_1 = \mathbf{w}_1 - \mathbf{w}_0 = \mathbf{w}_1$. Since no further useful information remains after extracting the necessary latent variables, $\mathbf{s}_k$ $(k = 1, \cdots, A)$, in the inner PLS algorithm, the outer PLS algorithm will be terminated after the extraction of the first latent variable, $\mathbf{t}_1$. In this case, the weight vector, $\mathbf{w}_1$, is equal to the normalized regression coefficients of linear PLS, whilst the weight vectors, $\mathbf{V}_1 = [\mathbf{v}_1, ..., \mathbf{v}_A]$, of the inner PLS model are the same as the weight vectors of linear PLS.

3.2.5 Practicalities Regarding the Nested PLS Algorithm

This Section discusses some practical issues that influence the resulting model using the Nested PLS (NPLS) algorithm. As these practical aspects of the algorithm are important for the performance of the algorithm, the effects are examined in the next chapter, Chapter 4. Similar to the Steepest Descent PLS (SDPLS) algorithm of Wold *et al.* (1989) and the Error Based PLS (EBPLS) algorithm of Baffi *et al.* (1999a,b), the NPLS algorithm is based on an optimisation algorithm. All issues regarding nonlinear optimisation will therefore affect these algorithms.

As discussed in Section 2.3.1 in nonlinear least squares, the optimisation methods can be affected by local minima, noise, multicollinearity and the ratio between the number of observations to the number of variables in the data set. In particular, overfitting is likely to occur for underdetermined data sets.

3.2.5.1 Cross Validation

Cross validation (CV) is a method that enables the evaluation of a given model in terms of its predictive ability, and the determination of the appropriate number of latent variables to include in the model. Cross validation was originally developed by Mosteller and Wallace (1963). A number of key references related to the application of cross validation in PLS include Stone (1974, 1977) and Allen (1974), who pioneered systematic resampling within the calibration set. Wold (1978), Krzanowski (1987) and Jackson (1987) examined the use of cross validation in PLS. More recent references on the estimation of the numbers of latent variables include Ferre (1993), Dey *et al.* (1996), Runger and Alt (1996), and Henry *et al.* (1999). Finally, of special note is the investigation of Jonathan *et al.* (2000) into a number of variants of cross validation for the assessment of the performance of predictive models, in particular the two-deep fashion. In the two-deep fashion the samples used to calculate the cross validation error are separated from the selection of the calibration samples.

One approach to cross validation is to divide the data set into L subsets, where L lies between one and the total number of observations. The groups can be selected without replacement, so that no observation is present in more than one group. If the number of groups L is the same as the number of observations, i.e. leave-one-out cross validation, overfitting and an under-estimation of the true predictive error can occur (Martens and Dardenne, 1998). Simulations have shown that the optimal number of groups lies between 4 and 11 (Wold, 1978).

Alternatively, data groups may be selected with replacement, i.e. where an observation is included in more than one group. This is called Monte Carlo Cross Validation MCCV (or bootstrapping), initially considered by Picard and Cook (1984).

Selection of the more appropriate method depends on the type of data and the objective of the analysis. It has been argued that cross validation tends to overestimate the number of latent variables in PCA and PLS. A number of stopping rules have been proposed, see Wold (1978) or Krzanowski (1987), that penalise the addition of more latent variables.

3.2.5.2 Dependence on the Nonlinear Function Fitted

Since the error vector $\mathbf{e} = \mathbf{u} - \hat{\mathbf{u}}$ is used directly in the optimization procedure, any underfitting or overfitting of the inner mapping $\hat{\mathbf{u}} = f(\mathbf{t})$ between **t** and **u** will affect the updating procedure of the weights. Thus an appropriate choice of smoothing parameter is important (Section 2.4.3.1.1). This is best explained from Figure 3.7.

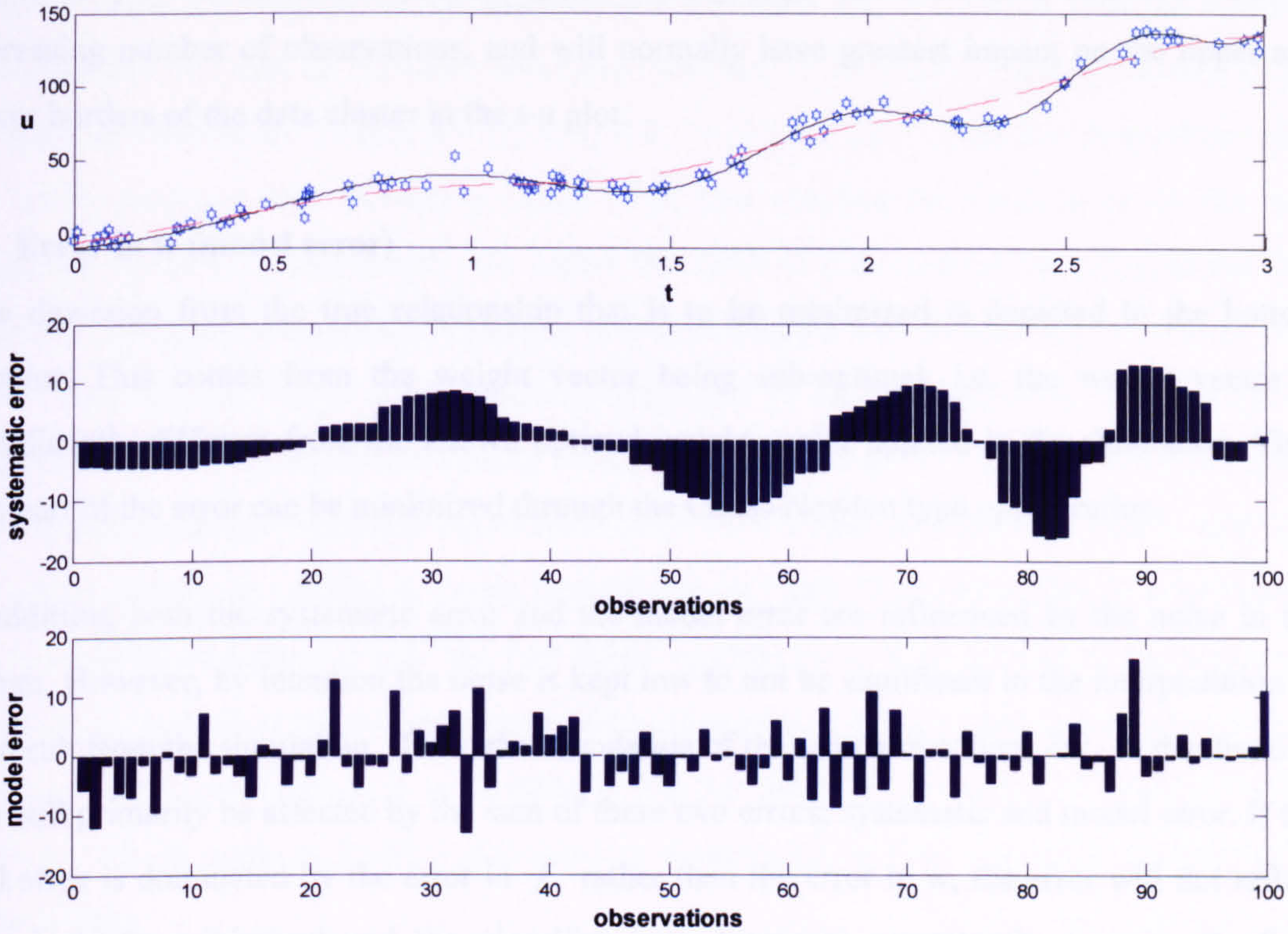

Figure 3.7. Systematic error introduced by underfitting the nonlinear function.

In the top subplot of Figure 3.7, a t-u plot is shown as blue stars with the known nonlinear relationship shown as a solid black line. The underlying function is known since the data is simulated, and the deviation from the underlying relationship comes primarily from the sub-optimal choice of weight vector, **w**. Consequently, there is little noise in **X** and **Y** in this illustrative simulation example. A function, f, is then fitted using local linear kernel regression to the data points and is shown as a dashed red line. This function uses a smoothing parameter that is too large, thus underfitting results compared to the known function. The discrepancy, shown in the middle plot, is greatest on the right of the curve. If this had been the situation in an iteration step for the Nested PLS, the question is what would the effect of such underfitting of the underlying relationship be?

The function, f, is underfitted compared to the known structure of the data. Thus the error vector ($\mathbf{e} = \mathbf{u} - \hat{\mathbf{u}}$) can be regarded as originating from two sources:

1. Error in f (Systematic error)

The difference between the known function and the fitted function, f, is shown in the middle subplot of Figure 3.7. This error originates mainly from the underfitting caused by the incorrect selection of the smoothing parameter. Secondly, it may originate from any casual structural deviation between the data points and the known function that affects the construction of the function, f. However, the probability of having large causal effects decrease with an increasing number of observations, and will normally have greatest impact on the upper and lower borders of the data cluster in the t-u plot.

2. Error in w (model error)

The deviation from the true relationship that is to be minimized is depicted in the bottom subplot. This comes from the weight vector being sub-optimal, i.e. the weight vector is significantly different from the known optimal weight vector applied in the simulation. Thus this part of the error can be minimized through the Gauss-Newton type optimization.

In addition, both the systematic error and the model error are influenced by the noise in the system. However, by intention the noise is kept low to not be significant in the interpretation of the result from the simulation. Thus, the calculation of the search direction, $\partial \mathbf{w}$, in the iteration step will primarily be affected by the sum of these two errors; systematic and model error. If the total error is dominated by the error in f, rather than the error in $\mathbf{w}$, the error will not reflect what is to be minimized and the algorithm may terminate prematurely as a result of an inappropriate search direction being estimated. This is due to increased noise in the error vector, $(\mathbf{e} = \mathbf{u} - \hat{\mathbf{u}})$, which is used as the predictor in the calculation of the search direction, $\partial \mathbf{w}$, e.g. for NPLS the error vector is used as the predictor variable in the inner PLS, Equation (3.1).

If a higher noise level is present, both the systematic error and the model error will be affected. However, essentially the same observations can be drawn, i.e. the systematic error from the underfitting of the inner mapping may be of the order of the model error that results, originating from an inappropriate choice of weight vector.

Alternatively, if the function f is overfitted, the errors will tend towards zero ($\| \mathbf{u} - \hat{\mathbf{u}} \|_2 \to 0$), and the risk of terminating in a local minimum is increased. This is because the model error caused by an inappropriate weight vector is reduced compared to the error due to noise in the system. The minimum value for the error vector occurs when the norm of the error vector becomes zero, i.e. when the function f goes through all the data points.

3.2.5.3 The Starting Criterion

As for all optimisation algorithms, the final model is dependent on the starting vector, i.e. the initial parameters used for the first function approximation. Ideally, a starting vector that is close to the final solution is sought. A natural choice is to use the same starting vector as derived from linear PLS, i.e. $\mathbf{w} = \mathbf{X}^T\mathbf{u}$ (Figure 3.8a). This will normally give acceptable results, but can be a problem when the majority of the variables suffer from a low correlation with the response. Consequently, to focus on the more important variables, an alternative starting criterion can be initiated, $\mathbf{w} = \text{cor}(\mathbf{X}, \mathbf{u})^p$, where $p = \{1, 3, 5, 7, 9, \cdots\}$ (Figure 3.8b). Here low correlation values are down weighted with increasing values of p, thus enabling the focus to be on the more important variables. Consequently, the higher the value of p the greater the focus on the more important variables. An alternative method is to give weights with an absolute correlation lower than a limit (c), e.g. 0.3, a value of zero as suggested by Martens and Næs, (1989). This will eliminate those variables that are weakly correlated with the response variable. When the data is slightly nonlinear, the regression vector from linear PLS, where the number of latent variables is identified using cross validation, can be applied (Figure 3.8d). This vector often will be closer to the desired solution than if only the first latent variable ($\mathbf{w} = \mathbf{X}^T\mathbf{u}$) is used. If the underlying structure is strongly nonlinear, the linear methods presented will not necessarily give the best starting vector. In this case, it is possible to use the Spline PLS method of Wold (1992) or related methods such as RVPLS discussed in Section 3.3 to find a suitable starting vector (Figure 3.8e).

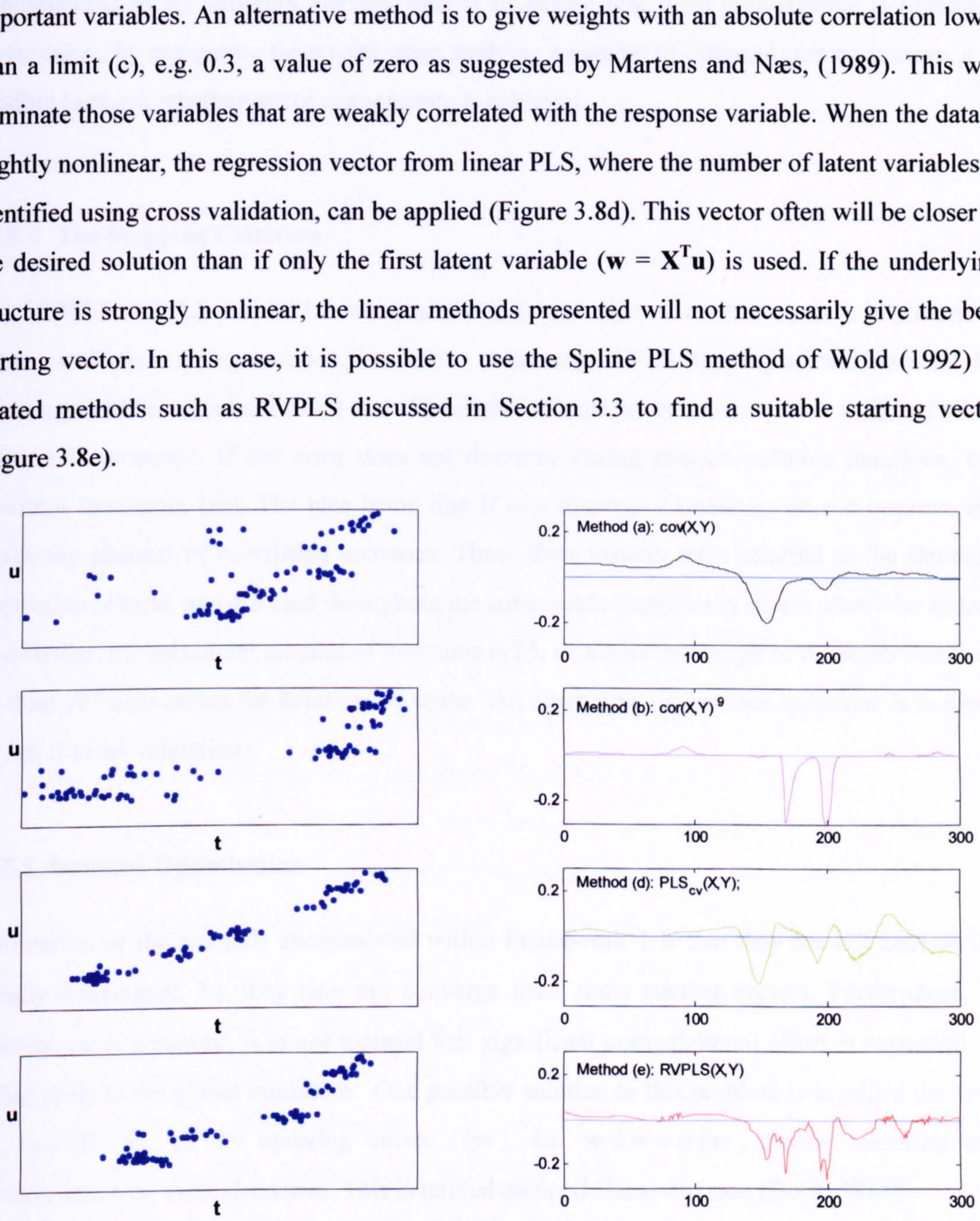

Figure 3.8. The influence of t-u relationship of four different starting vectors.

Figure 3.8 shows the effect of applying different starting vectors. The starting vectors shown are constructed by the four methods a, b, d and e, and are located on the right hand side with the corresponding t-u plot is shown on the left. Generally, the better the underlying structure identified by the starting vector, the better is the performance of the starting vector and thus the final model obtained. A good starting vector will reduce the chance of identifying a local minimum that is far from the global minimum, and since the magnitude of the error vector (||e||) will be smaller, the risk of overfitting is reduced. However, it should be kept in mind that the method of constructing the starting vector could cause overfitting.

Consequently, for noisy data or when uncertain about the degree of nonlinearity in the data, it is recommended to try different starting vectors to ensure that good convergence is obtained. Furthermore, by comparing the results when applying a number of different starting vectors, it is possible to check whether or not convergence is achieved.

3.2.5.4 The Stopping Criterion

Since EBPLS, SDPLS and NPLS are minimising a least squares expression, it is important to define a realistic stopping criterion. In addition to the normal PLS stopping criteria based on the maximum number of iterations (i) and the relative change in the score vector **t** (ii), a further criterion is proposed. If the error does not decrease during two consecutive iterations, the algorithm terminates (iii). The idea being that if two successive iterations do not improve the model, the chances of overfitting increases. These three criteria were selected as the standard termination criteria, and are used throughout the subsequent comparison unless otherwise stated. In particular, the maximum number of iterations is 25, or a relative change in the score vector of less than 10^{-8} also causes the iterations to cease. An alternative termination technique is to use a test set or cross validation.

3.2.5.5 Damped Optimisation

A limitation of the methods encapsulated within Framework 1 is that they are not necessarily globally convergent, i.e. they may not converge from some starting vectors. Furthermore, if convergence is achieved, it is not unusual that significant computational effort is expended in getting close to the global minimum. One possible solution to this problem is to adjust the step size, $\alpha \in \{0 \cdots 1\}$, of the updating vector ($\partial \mathbf{w}$), i.e. $\mathbf{w} = \mathbf{w} + \alpha \partial \mathbf{w}$, thereby ensuring the objective function value decreases. This is termed damped Gauss-Newton (Bock, 1981).

Since **w** is normalised after the increment is added, an alternative generalization is to include step sizes larger than unity, i.e. $\alpha > 0$. Then the weight updating, $\mathbf{w} = \mathbf{w} + \alpha\,\partial\mathbf{w}$, can alternatively be written as $\mathbf{w} = (1-\beta)\mathbf{w} + \beta\,\partial\mathbf{w}$, where $\beta \in \{0, 1\}$ denotes the step length and $\partial\mathbf{w}$ the step direction. For both cases, **w** is normalised to attain a norm of unity. Using a variable step size is common practise when applying most optimisation algorithms. Strictly, this is not dampening but the term is used even when α exceeds unity. The step length can be found using a separate optimization method, e.g. the golden search method (Lewis *et al.* 2000). Dampening can be applied for the methods encapsulated within Framework 1.

3.2.6 Summary of Nested PLS

The theoretical background of Nested PLS has been presented and related to the existing methods of SDPLS and EBPLS. The main contribution of NPLS is the use of linear PLS within the optimization framework that is common to these three nonlinear PLS methods. The strengths of NPLS, are its ability to handle underdetermined and multcollinear data. The weakness of NPLS relates to overfitting, but can be reduced as discussed in Section 3.2.5.

3.3 The Reciprocal Error Variance Criterion

An alternative criterion for the derivation of the weight vector is proposed, the Reciprocal Variance criterion. The resulting nonlinear PLS algorithm, Reciprocal Variance PLS (RVPLS), builds on the Spline PLS algorithm, (Wold, 1992). The proposed concept is motivated by the need to reduce the number of latent variables, by focusing more on the response variance compared to the Spline PLS approach. This is because a reduced number of latent variables reduces the risk of overfitting. The concept is based on the weighted average (Taylor, 1997). It is first developed theoretically for a simple multicollinear data set where there is a single underlying phenomenon in **X** related to the response **y**, and the resulting error is distributed as a multivariate normal distribution. An example of an underlying phenomenon could be the peak in a spectrum (typically FT-IR) that is correlated with the response variable through Beers law ($A \propto c$), i.e. the Absorbance (A) is proportional to the concentration (c) and thus the signal itself. In practice the model is shown to also work when there are only a limited number of underlying phenomena, particularly for low rank spectral data. The approach is not dependent on the dimensionality of the data set and generally works well when there are fewer observations than variables. However, the uncertainty of the estimation of the weights and the inner mapping increases with a decreasing number of observations.

3.3.1 The Weighted Average

Consider an unknown quantity, y, being measured by a system comprising two sensors, each recording a single measurement, z_k $(k=1, 2)$, in the presence of random, independent unbiased measurement error, ε_i $(i=1, 2)$:

$$y = z_1 + \varepsilon_1 \qquad \text{and} \qquad y = z_2 + \varepsilon_2 \tag{3.12}$$

In the absence of other information, an optimal estimate of y, that is a linear combination of the measurements, is sought:

$$\hat{y} = w_1 z_1 + w_2 z_2 \tag{3.13}$$

It is assumed that the weights w_1 and w_2 are independent of y and hence the expression $(y - \hat{y})$ is unbiased:

$$E[y - \hat{y}] = E[w_1(y - \varepsilon_1) + w_2(y - \varepsilon_2) - y] = 0 \tag{3.14}$$

Now letting $E[e_1] = E[e_2] = 0$ and $E[y] = y$ and $w_2 = 1 - w_1$, the mean squared error is given by:

$$E[(y - \hat{y})^2] = w_1^2 \sigma_1^2 + (1 - w_1)^2 \sigma_2^2 \tag{3.15}$$

where σ_1^2 and σ_2^2 denote the variance of ε_1 and ε_2, respectively. Differentiating Equation (3.15) with respect to w_1 and setting the result equal to zero gives:

$$2w_1\sigma_1^2 - 2(1 - w_1)\sigma_2^2 = 0 \tag{3.16}$$

resulting in the weight:

$$w_1 = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2} = \frac{1/\sigma_1^2}{1/\sigma_1^2 + 1/\sigma_2^2} \tag{3.17}$$

The general case can be written as (Taylor, 1997),

$$w_k = \frac{1/\sigma_k^2}{\sum_{i=1}^{n}(1/\sigma_i^2)} = \frac{1/\sigma_k^2}{\sum_{i=1}^{n}(1/\sigma_i^2)} \tag{3.18}$$

and the corresponding minimum mean square error is as follows,

$$\mathrm{E}[(y-\hat{y})^2]=\left(\sum_{i=1}^{n}\frac{1}{\sigma_i^{\,2}}\right)^{-1} \tag{3.19}$$

that is, the optimal relative weighting is the inverse of the error variance.

3.3.2 The Reciprocal Variance Criterion

The mathematical concept of the weighted average is extended to the case of regression, where a trend is to be modelled by n variables as opposed to a constant. Assuming $\mathbf{y}$ (m x 1) can be modelled by two variables $\mathbf{x}_1$ and $\mathbf{x}_2$, and that the noise is independent:

$$\underbrace{\mathbf{y}=\mathbf{x}_1+\boldsymbol{\varepsilon}_1}_{\varepsilon_1\sim N(0,\sigma_{E_1})} \quad \underbrace{\mathbf{y}=\mathbf{x}_2+\boldsymbol{\varepsilon}_2}_{\varepsilon_2\sim N(0,\sigma_{E_2})} \tag{3.20}$$

Then in a similar manner to the PLS covariance criterion, the model is defined by the weights w_1 and w_2 where $\mathbf{w}$ is normalised to unit length. The linear PLS model for the first latent variable is given by:

$$\hat{\mathbf{y}}=d(\mathbf{X}\mathbf{w})=dw_1\mathbf{x}_1+dw_2\mathbf{x}_2 \tag{3.21}$$

where d is a scaling constant that includes the ratio between $\|\mathbf{w}\|$ and Σw_i. If the noise is Gaussian and independent, then:

$$E[(\mathbf{y}-\hat{\mathbf{y}})^2]=d^2\,w_1^{\,2}\sigma_1^{\,2}+d^2\,(1-w_1)^2\sigma_2^{\,2} \tag{3.22}$$

Minimising Equation (3.22) gives:

$$\begin{gathered}2d^2\,w_1\sigma_1^{\,2}-2d^2\,(1-w_1)\sigma_2^{\,2}=0\\ \Updownarrow\\ w_1\sigma_1^{\,2}-(1-w_1)\sigma_2^{\,2}=0\end{gathered} \tag{3.23}$$

This is the same result as in Equation (3.18) with the weight vector being given by $\tilde{w}_1=1/\sigma_1^{\,2}$ and $\tilde{w}_2=1/\sigma_2^{\,2}$. The weight vector, $\tilde{\mathbf{w}}$, is then normalised using the Euclidean norm and not the sum as was used for normalising the weighted average. However, the difference

between these two scaling methods is included in the regression coefficient of **u** on **t**, thus the difference in normalisation factor does not affect the final model.

Consider now the calculation of the regression coefficient between $\mathbf{u}$ and $\mathbf{x}_k$. The error vector $\boldsymbol{\varepsilon}_k$, is given by $\boldsymbol{\varepsilon}_k = \mathbf{u} - \hat{\mathbf{u}}_k = \mathbf{u} - \mathbf{x}_k(\mathbf{u}^T\mathbf{x}_k)/(\mathbf{x}_k{}^T\mathbf{x}_k)$. Letting $\tilde{\mathbf{w}}$ represent the weight before normalisation, the relative weight then becomes

$$\tilde{w}_k = sign(\mathbf{u}^T\mathbf{x}_k)/\sigma_{\varepsilon_k}^2 \tag{3.24}$$

where the sign of the weight is found from the sign of the covariance between **u** and $\mathbf{x}_k$. This weight criterion has the effect of giving all weights a 'minimum' value as seen from the calculation of $\boldsymbol{\varepsilon}_k$. That is, if $\mathbf{x}_k$ cannot model **u**, i.e. if $\hat{\mathbf{u}}_k = \mathbf{0}$ the zero vector, the weight of that variable approaches $\sigma_{\mathbf{u}}^{-2}$ since $\boldsymbol{\varepsilon}_k = \mathbf{u} - \hat{\mathbf{u}}_k = \mathbf{u}$, Equation (3.24). This value will be the "threshold value" and those variables attaining this value are given zero weight. Thus:

$$\tilde{w}_k = sign(\mathbf{u}^T\mathbf{x}_k)(1/\sigma_{\varepsilon_k}^2 - 1/\sigma_{\mathbf{u}}^2) \tag{3.25}$$

3.3.3 Nonlinear Partial Least Squares

Extending the weight vector based on the reciprocal error variance to the nonlinear case is not straightforward. Given the nonlinear model:

$$\mathbf{u} = f(\mathbf{t}) + \mathbf{e} = f(w_1\mathbf{x}_1 + w_2\mathbf{x}_2 + \cdots + w_n\mathbf{x}_n) + \mathbf{e} \tag{3.26}$$

w_k has to be determined independently by fitting each $\mathbf{x}_k$ to the scores vector to **u** through the function, f. Thus the basic theory of the weighted average cannot be applied directly. From the work of Wold (1992), a methodology was proposed for the extension of the weight vector from the linear to the nonlinear case. Here $\mathbf{x}_k$ is first scaled to have the same variance as **t**, i.e. $v_k = sign(\sigma_t/\sigma_{\mathbf{x}_k})$, where σ_t is the standard deviation of the score vector and $\sigma_{\mathbf{x}_k}$ is the standard deviation of variable *k*. The sign $s_k = \{-1,1\}$ is found from the sign of $(\mathbf{t}^T\mathbf{x}_k)$. The model of **u** ($\hat{\mathbf{u}}_k$) was then estimated as $f(\mathbf{x}_k v_k)$ and thus the linear criterion from Equation (2.54) was extended to the nonlinear case:

$$\tilde{w}_k = COR(f(\mathbf{x}_k v_k), \mathbf{u})\sigma_{\mathbf{x}_k} \tag{3.27}$$

where v_k and $\tilde{w}_k$ are given the appropriate sign and the function, f, is given by the fit between **t** and **u**.

The same framework is used when estimating the weight, $\mathbf{w}_\varepsilon$, by the reciprocal variance criteria. First, the error between the fit of the given variable and **u** must be calculated:

$$\boldsymbol{\varepsilon}_k = \mathbf{u} - f(\mathbf{x}_k v_k) \tag{3.28}$$

and then

$$\tilde{w}_k = sign(\mathbf{t}^{\mathrm{T}}\mathbf{x}_k)(1/\sigma_{\varepsilon_k}^{\ 2} - 1/\sigma_{\mathbf{u}}^{\ 2}) \tag{3.29}$$

The first step in the calculation of the weight vector is to obtain the sign of v_k. One approach for calculating the sign was proposed by Wold (1992). The sign for both $f(\mathbf{x}_k v_k)$ and $f(-\mathbf{x}_k v_k)$ are calculated and the one that gives the best fit of **u**, calculated in terms of the squared error is selected.

The next stage is to estimate the variance. This stage can be enhanced by using a robust estimate (Hoaglin *et al.*, 1983) of the standard deviation. For example the mean absolute deviation, the median absolute deviation, the fourth-spread or the estimator based on the bi-weight estimator of location. For the bi-weight estimator, the observations are given different weights according to their distance from the median using the bi-weight function, i.e. the greater the distance from the median, the smaller the weight. The bi-weight estimator is efficient under a number of distributional assumptions.

Criterion	Weight Criteria	Explanation
SPLS	$s_k = \mathrm{sign}(\mathbf{x}_k{}^{\mathrm{T}}\mathbf{t})$	Obtain the sign of the covariance
	$v_k = \mathrm{s}_k[\sigma_{tk}/\sigma_{\mathbf{x}k}]$	Find the correct scaling of X_k
	$w_k = \mathrm{s}_k[\mathrm{COR}(\mathbf{u}, f(v_k\mathbf{x}_k))\sigma_{\mathbf{x}_k}]$	Calculate the weight for the given variable
RVPLS	$s_k = \mathrm{sign}(\mathbf{x}_k{}^{\mathrm{T}}\mathbf{t})$	Obtain the sign of the covariance
	$v_k = \mathrm{s}_k[\sigma_{tk}/\sigma_{\mathbf{x}k}]$	Find the correct scaling of $\mathbf{x}_k$
	$\boldsymbol{\varepsilon}_k = \mathbf{u} - f(v_k\,\mathbf{x}_k)$	Calculate the k^{th} error vector, $\boldsymbol{\varepsilon}_k$
	$w_k = \mathrm{s}_k[1/\mathrm{VAR}(\boldsymbol{\varepsilon}_k) - 1/\mathrm{VAR}(\mathbf{u})]$	Calculate the weight for the given variable

Table 3. Calculation of the relative weight using the different weight criteria.

Table 3 describes the difference between the calculation of the weight vector between SPLS and RVPLS, elsewhere the algorithms are identical. The MATLAB algorithm describing the RVPLS is given in Algorithm 3.2:

```
...                                                    ...
    while conv > limit,                                % Repeat until convergence
        uold = u;                                      % Retain the old Y score vector to control the convergence
        t = X*w;                                       % Calculate the X score vector
(i)     û = f(t, u);                                   % Calculate the nonlinear fit
(ii)    q = Y'* û;                                     % Regress Y on û (not on u as in linear PLS)
        q = q/norm(q);                                 % Normalize to unit length
        u = Y*q;                                       % Improve u using the linear combination q
        for i = 1 : n,                                 % For each of the weights/variables
(iii)       x = X(:,i);                                % Select the i^th variable
(iv)        s = sign(t'*x);                            % Find if x is negative or positive correlated to t
(v)         sx = std(x);                               % Calculate the standard deviation of the ith variable
(vi)        v = std(t)/sx;                             % Calculate the scaling constant
(vii)       e = u - f(s*v*x));                         % Calculate each weight according to criterion
(viii)      w(i) = s/(std(e).^2)- s/(std(u).^2);       % Weight loop
        end                                            % Normalize to unit length
        w = w/norm(w);                                 % Convergence if s < limit (and no. of iter. > max iter.)
        conv = norm(u-uold)/norm(u);                   % Inner while loop
    end
```

Algorithm 3.2. The reciprocal error variance PLS algorithm (MATLAB code).

For the work presented in this Thesis, the median absolute deviation estimator was selected as it performs well and is a simple metric. Table 3 summarises the different weight criteria whilst Algorithm 3.2 gives the key algorithmic steps in the nonlinear algorithm.

3.3.4 A Simple Example

The purpose of this example is to illustrate the fundamental differences between the covariance criterion used in Spline PLS and the reciprocal variance criterion used in RVPLS. Assume $\mathbf{y}$ (m x 1) can be modelled by two variables $\mathbf{x}_1$ and $\mathbf{x}_2$, and that the noise vectors $\boldsymbol{\varepsilon}_1$ and $\boldsymbol{\varepsilon}_2$ are independent. From Equation (3.20):

$$\mathbf{y} = f(\mathbf{x}_1) + \boldsymbol{\varepsilon}_1 \quad \mathbf{y} = f(\mathbf{x}_2) + \boldsymbol{\varepsilon}_2 \tag{3.30}$$

A simulation model is constructed based on Equation 3.30, where $\mathbf{X}$ and $\mathbf{y}$ are autoscaled to mean zero and a variance of unity. The results obtained using Wold's weight criterion are

shown in the upper three plots of Figure 3.9(a-c), whilst the results obtained using the reciprocal variance criterion are shown in the lower three plots, Figure 3.9(d-f). The model is identical to the model in Equation (3.26), i.e. $\mathbf{u} = f(\mathbf{t}) + \mathbf{e} = f(w_1\mathbf{x}_1 + w_2\mathbf{x}_2) + \mathbf{e}$, where the weights, w_1 and w_2, are calculated based on the methods of SPLS and RVPLS.

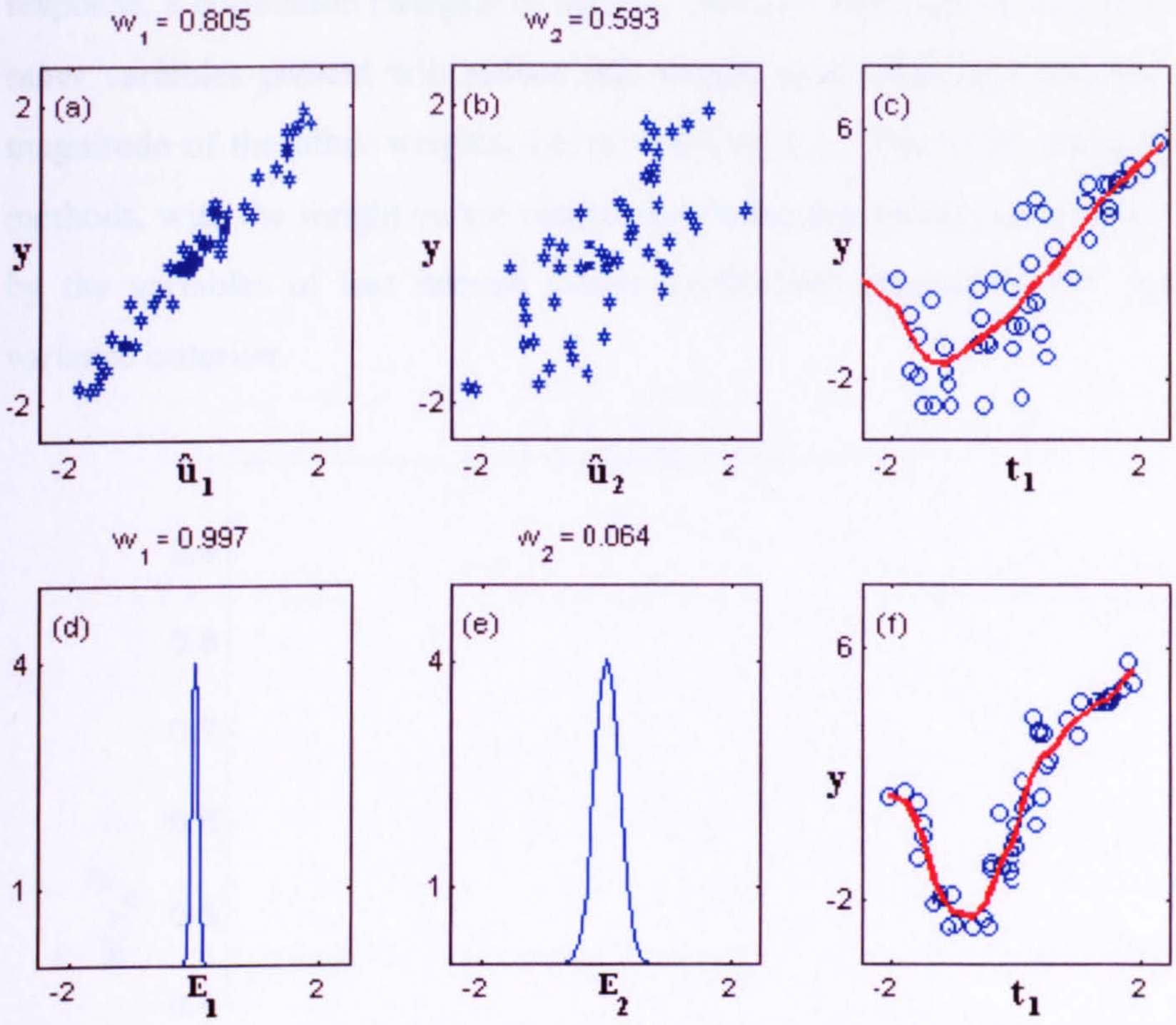

Figure 3.9. Comparison of the covariance and the reciprocal variance criterion

Only the first latent variable is discussed since including more latent variables did not improve the predictive ability of either method. In the Spline PLS algorithm, for autoscaled data, both weights are constructed using the correlation between $\mathbf{y}$ and the corresponding fit $\hat{\mathbf{u}}_k = f(\mathbf{x}_k v_k)$, where k = (1, 2). Thus the construction of the two weights w_1 and w_2 can be observed directly from Figure 3.9(a and b). The correlation in plot (a) is greater than in plot (b) reflecting the difference in the value for the weights ($w_1 > w_2$) calculated from Equation (3.27).

The reciprocal variance criterion is calculated from the error distributions, Equation (3.29), shown in Figure 3.9(d and e). It can be observed that the reciprocal variance criterion gives greater weight to the first variable ($w_1 = 0.997$, Figure 3.9d) than that obtained using the covariance criterion ($w_1 = 0.805$, Figure 3.9a). The reason is that the reciprocal variance criterion is constructed using the inverse of the variance for the error, $1/VAR(\mathbf{u} - \hat{\mathbf{u}}_k)$. Consequently, the weight of the k^{th} variable will tend towards infinity as the error tends towards zero, as seen from Equation (3.24). After normalization, the k^{th} weight will then tend towards

the value of unity whilst the other weights will tend towards the value of zero. This is the desired behaviour and is commensurate with the idea behind the weighted average.

For the covariance criterion of Wold, if a single variable explains 100% of the variance of the response, a correlation (weight) of unity is obtained. But after normalisation, the weights of the other variables present will reduce this weight to a value less than unity depending on the magnitude of the other weights, i.e. $\mathbf{w} = \mathbf{w}/\|\mathbf{w}\| = 1$. This is the main difference between the methods, with the weight vector obtained applying the covariance criterion being more affected by the variables of less interest (lower correlation) compared with applying the reciprocal variance criterion.

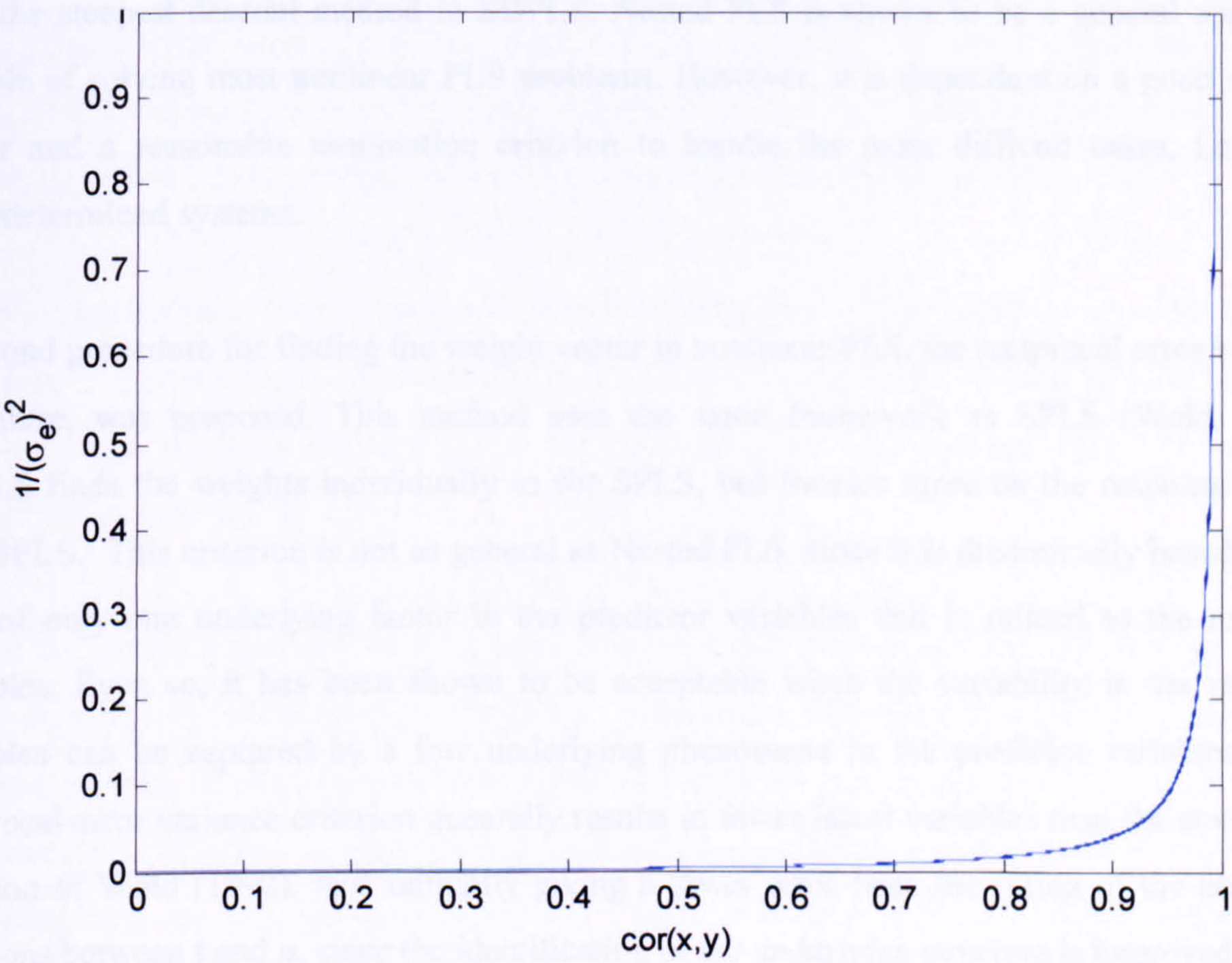

Figure 3.10. Relationship between the covariance and the reciprocal variance criterion

Figure 3.10 illustrates the relationship between the individual weight values calculated, w_k, by the two criteria, covariance and reciprocal variance, before normalization of the weight vector. Thus, the deviation between the two approaches increases with increasing correlation, thus after normalization, the reciprocal variance gives significantly higher weights to those variables of greater importance compared to the covariance criterion.

3.4 Conclusions

In this Chapter, the issues that arise from two existing weight updating schemes for nonlinear PLS were described. The two frameworks were the error based weight updating scheme of Baffi *et al.*, (1999a) based on the work of Wold et al., (1989) and the covariance criterion adapted for process nonlinearity (Wold, 1992).

A new procedure was proposed for calculating the weight vector, Nested PLS, which avoids the multicollinear problem of the EBWU procedure, by applying PLS, with cross validation, to estimate the linear Taylor approximation in the Gauss-Newton updating scheme. The Nested PLS approach also decreases the possibility of the global minimum not being attained compared with the steepest descent method in SDPLS. Nested PLS is shown to be a general approach, capable of solving most nonlinear PLS problems. However, it is dependent on a good starting vector and a reasonable termination criterion to handle the more difficult cases, i.e. noisy underdetermined systems.

A second procedure for finding the weight vector in nonlinear PLS, the reciprocal error variance procedure, was proposed. This method uses the same framework as SPLS (Wold, 1992). RVPLS finds the weights individually as for SPLS, but focuses more on the response values than SPLS. This criterion is not as general as Nested PLS, since it is theoretically based on the case of only one underlying factor in the predictor variables that is related to the response variables. Even so, it has been shown to be acceptable when the variability in the response variables can be captured by a few underlying phenomena in the predictor variables. The reciprocal error variance criterion generally results in fewer latent variables than the covariance criterion of Wold (1992), thus normally giving a lower error from the fitting of the nonlinear functions between **t** and **u**, since the identification of the underlying structure is improved.

The main objective of this Chapter was to present the two different criteria, both based on previously published ideas. The two concepts are very different and are not easy to compare. The NPLS approach and the methods of SDPLS and EBPLS which it is based on, attempt to minimize the error between the scores **t** and **u**, through a nonlinear function $f(\mathbf{t})$. This is achieved through three different optimisation methods. Since the methods focus on minimising the error between the fitted nonlinear function $\hat{\mathbf{u}}$ and the Y-score **u**, the error of the inner mapping will generally be reduced. Thus the main problems are those associated with local minima and that caused by overfitting either as a result of having an underdetermined system of

equations or due to the presence of multicollinearity. NPLS uses an inner partial least squares algorithm in the optimisation stage that counters both problems.

RVPLS and SPLS are more true to the concept of the ordinary PLS algorithm. The weights are calculated independently as a function relating the corresponding **X** variable to the Y-score **u**. In particular the framework aims to find each weight independently of the other weights. Furthermore, the model is constrained since the weight, w_k, is required have the same sign as the correlation between $\mathbf{x}_k$ and **t**. As a result, this framework will generally reduce the potential for overfitting, but will be affected by the error introduced when fitting the inner mapping between **t** and **u**. Because these methods are not focusing on minimizing the error between **u** and $f(\mathbf{t})$, the outcome may be that the underlying structure is not properly identified, thereby introducing a large error when the function, $f(\mathbf{t})$, is fitted. The error introduced will be propagated to the next latent variable and will affect the accuracy of the resulting model, as any smoothing defect can not generally be captured by subsequent latent variables.

In linear PLS the same result materializes when the orthogonal latent variables are regressed one by one or if **Y** is regressed on the whole of the **T** matrix. This is not the case when a nonlinear mapping exists between the scores. For the linear case, the intermediate regression vector is found from $\mathbf{b} = (\mathbf{T}^T\mathbf{T})^{-1}\mathbf{T}^T\mathbf{Y}$.

Due to the orthogonality of the **T** matrix, $\mathbf{T}^T\mathbf{T}$ is a diagonal matrix, and the inverse can be found directly as $1/\mathrm{diag}([\mathbf{t}_1{}^T\mathbf{t}_1, \cdots, \mathbf{t}_A{}^T\mathbf{t}_A])$. Thus, the independent regression coefficient becomes $\mathbf{b}_j = (\mathbf{t}_j{}^T\mathbf{Y})/(\mathbf{t}_j{}^T\mathbf{t}_j)$, and the regression analysis may be done sequentially. For the nonlinear case $\hat{\mathbf{U}} = [\hat{\mathbf{u}}_1, \hat{\mathbf{u}}_2, \cdots, \hat{\mathbf{u}}_k] = [f(\mathbf{t}_1), f(\mathbf{t}_2), \cdots, f(\mathbf{t}_k)]$ is the matrix of sequential approximations to **Y**. Even if $\mathbf{t}_j{}^T\mathbf{t}_i = 0$, $\hat{\mathbf{u}}_j{}^T\hat{\mathbf{u}}_i \neq 0$, carrying out the regression either sequentially or simultaneously will produce different results. Consequently, even if the objective of SPLS and RVPLS are closer to that of linear PLS, going from the linear to the nonlinear case is not straight forward. A modified objective further away from the linear PLS algorithm, such as including a weight updating scheme, could be beneficial if the general performance is improved. In the next chapter, the performance of SDPLS, EBPLS, NPLS, SPLS and RVPLS are compared.

CHAPTER 4

APPLICATION STUDIES OF NONLINEAR PARTIAL LEAST SQUARES

4.1 Introduction

In this Chapter the two new nonlinear PLS algorithms of Reciprocal Variance PLS and Nested PLS are compared with the nonlinear PLS methods from which they were developed, i.e. Spline PLS, Steepest Descent PLS and Error Based PLS. In addition to the nonlinear PLS methods, two reference methods are discussed. In those studies where the underlying data is known to be approximately linear through process understanding, ordinary linear PLS is included in the comparison. Secondly, for a single response, local linear kernel regression was applied between the predicted response obtained from the linear PLS model and the measured response to model any underlying nonlinear behaviour (Martens and Næs, 1989). That is, the regression vector (b) obtained from linear PLS is used directly as the first and only weight vector in the nonlinear PLS framework. This method is denoted BPLS. The different methods compared are summarized in Table 4.

Methods		Framework	Denoted
A.	Reciprocal Variance PLS	2	RVPLS
B.	Spline-PLS	2	SPLS
C.	Error Based PLS procedure	1	EBPLS
D.	Nested PLS procedure	1	NPLS
E.	Steepest Descent PLS procedure	1	SDPLS
F.	Ordinary Linear PLS	3	PLS
G.	Linear PLS + nonlinear mapping	3	BPLS

Table 4. PLS algorithms included in the comparison.

The performance of the different nonlinear PLS algorithms is influenced by a number of parameters, including the starting and stopping criteria. To address these issues the models were first compared using standard parameter settings prior to examining the effect of the parameters separately. The modelling results from the reference methods (F-G) were included in the study to help identify whether the nonlinear model is better than the linear reference models. The standard settings used for all the nonlinear PLS methods were as follows:

1. Local linear kernel regression using the plug-in bandwidth of Bowmann and Azzalini (1997) was used as the nonlinear function between **t** and **u**. In addition, a variable plug-in bandwidth was included (Section 2.5.2.1). This method was chosen as it is universal, generally exhibits good performance and is faster than cross validation to estimate the smoothing parameter (Appendix A2.2).
2. The starting Y-score vector **u** was selected to be the first column of the **Y** matrix.
3. The standard starting vector was selected as $\mathbf{w} = \text{cor}(\mathbf{X}, \mathbf{u})^9$. Applying the power of 9 gives greater emphasis to the most important variables, compared with a power of unity. Nine was chosen from experience as it gave good overall performance, but any power from the series $n = \{3, 5, 7, 9, \ldots\}$ may be applied. The power, *p*, could also be found using a separate optimization algorithm.
4. The termination criteria were based on; (1) a relative change in the score vector of less than 10^{-8}, (2) maximum number of iterations of 25, or (3) if two subsequent iterations did not result in a decrease in the calibration error.
5. For NPLS, the number of groups when applying cross validation to the inner PLS was taken to be two, due to the ease of calculation.
6. For EBPLS, NPLS and SDPLS (Framework 1), no form of dampening (or variable step length) was used in the optimisation.

Thus the only difference between the various nonlinear PLS algorithms was how the weight vector was calculated. The benefit of this is that for the comparison, only the estimation of the weights differs between the approaches. The drawback of this is that a particular setting may bias the result in favour of one of the methods, e.g. the steepest descent method tends to converge more slowly than Gauss-Newton. Therefore, the effects of items 3-6 on the nonlinear PLS algorithms were investigated separately. Ten data sets were investigated to reduce the risk of drawing incorrect conclusions. The general nonlinear PLS algorithm, common to all the approaches, is shown as MATLAB code in Appendix A2.5.

4.1.1 The Data Sets

The nonlinear PLS algorithms were compared by investigating their prediction performance on ten data sets. Of the ten, three data sets were thoroughly examined and are reported in the main body of the Thesis, Table 5 (1-3). The first two data sets are underdetermined and contain highly multicollinear data, whilst the third data set is overdetermined with a low level of collinearity existing between the variables. These three data sets were selected to enable a comparison between the different aspects of the algorithms to be undertaken. In particular, the

effect of the start vector, the termination criterion, the number of subsets used in the application of cross validation in the inner PLS algorithm, and the use of damped optimisation were examined (Section 3.2.5.5). The modelling results from the additional seven data sets described in Table 5 (4-10) are included in Appendix A1, and discussed in Section 4.5.

Identification	Comments	Matrix size (X)
1 Polymer Density	In-line near infrared spectroscopy data recorded on a high-density polyethylene process, modelling density	(87 x 301)
2 Alkylation Product	At-line near infrared spectroscopy data recorded for a pharmaceutical alkylation process, modelling both product and by-product concentrations	(45 x 401)
3 Melt Index I	Process data recorded for a high-density polyethylene process, modelling Melt Index.	(300 x 87)
4 Melt Index II	Nuclear magnetic resonance spectroscopy data recorded on a polypropylene plant, modelling Melt Index	(454 x 28)
5 Melt Index III	Process data recorded from a polyethylene pilot reactor study, modelling four melt indices.	(50 x 15)
6 Xylene products	Three Xylene concentrations (metha, ortho, para) measured by ultra violet (UV) spectroscopy	(196 x 30)
7 Moisture in fibre	Near infrared data set (Blanco *et al.*, 2000), modelling the moisture in acrylonitrile-vinyl acetate polymer	(60 x 700)
8 Rise Time	Rise time of a servo motor (Ulrich, 1986), modelled by two gain settings and five categorical variables	(99 x 12)
9 Simulation I	Simulation of a pH process (Henson and Seborg, 1994), used in Baffi *et al.* (1999b), modelling pH.	(700 x 4)
10 Simulation II	Simulation used to investigate the impact of noise level and the data set size, one response.	various

Table 5. Summary of data sets included in the comparison.

4.1.2 Comparison of the Nonlinear PLS Methods

Each data set is divided into a calibration and a validation data set. Performance of the nonlinear PLS methods was compared, by first developing models using the calibration data set, and then independently checking the models on a validation data set. The performance of the models is reported in terms of the Root Mean Squared Error of Calibration (RMSEC) and Root Mean

Squared Error of Prediction (RMSEP). Furthermore, the variance explained by the **X** and **Y** matrices and the number of latent variables included in the models are summarized. The impact of the different starting vectors, the effect of dampened optimization, and the influence of the termination criteria for the methods in Framework 1 (Table 4, C-E) are also investigated. Finally, the effect of the number of groups used in cross validation in the inner PLS loop of the Nested PLS algorithm is examined.

The plots used in the comparison are those of the inner score space for each pair of latent variables (i.e. t-u plots), and the corresponding weight vectors. The t-u plot is included as it is important to examine the relationship between the scores **t** and **u**, especially for the first latent variable. A good fit will reduce the likelihood of an error being introduced when fitting the nonlinear function between the scores. The 'goodness of fit' in terms of the underlying structure is observed in the t-u plots, and the configuration of the weight vectors provide an indication as to whether the methodology is prone to overfitting. For spectral data, a noisy weight vector indicates overfitting in contrast to a smooth trajectory. Thus it is possible to discuss and compare the performance of the methods from these representations. The calibration data set is plotted as green circles (o) and the validation data is denoted by crosses (+). A solid line (—) denotes the nonlinear function fitted.

4.2 Density of Polymer using Near Infrared Spectroscopy.

The first data set analysed is based on a near infrared data set collected on a high-density polyethylene (HDPE) plant between 1999 and 2001. The density is regulated to control a number of physical properties by varying the quantity of comonomer in the reactor. The nonlinear behaviour is a consequence of the density/concentration relationship that is related to the crystallinity of the product. A small amount of comonomer reduces the crystallinity of the product dramatically as a result of the comonomer side-chains inability to be arranged in a crystal structure. This effect decreases as the level of comonomer increases, approaching the density of amorphous polyethylene, hence the presence of nonlinear behaviour. In addition the comonomer can be distributed either in a homogeneous or heterogeneous manner in the polymer.

The near infrared spectra were recorded on the polymer melt using an in-line probe attached to an extruder, Figure 4.11. The polymer density was measured using a density column. The data set consists of 301 predictor variables, representing different wavelengths in the near infrared

spectrum. The calibration data set comprises 87 observations and the validation data set comprises 91 observations. The single response variable is density.

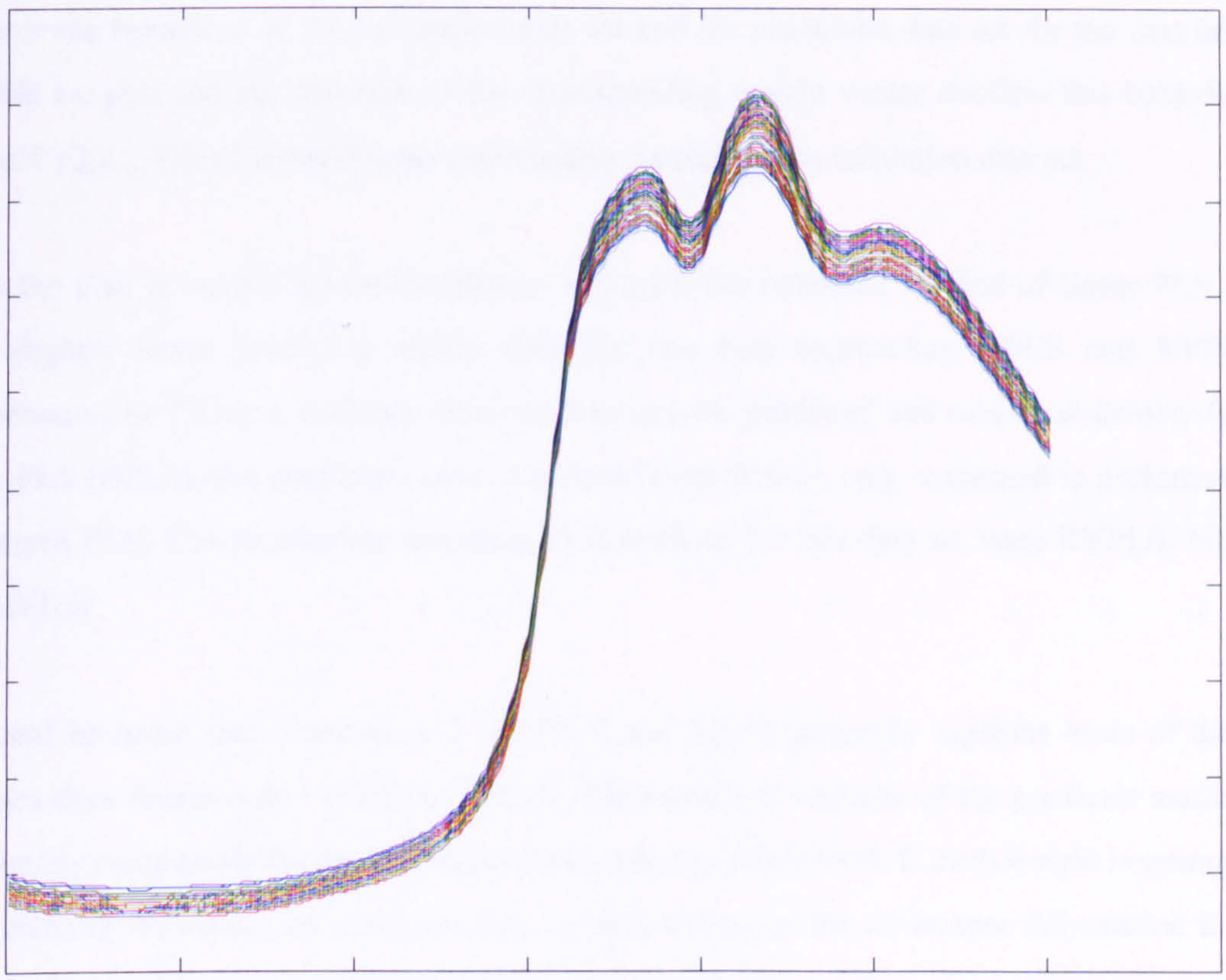

Figure 4.11. Plot of the original spectra for the density data (labels omitted due to confidentiality).

Method applied	No. of latent variables	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
A: RVPLS	4	60.33	99.90	0.00096	0.00152
B: SPLS	4	88.84	99.80	0.00192	0.00292
C: EBPLS	3	16.42	100.00	9.86e-6	0.00437
D: NPLS	1	14.90	99.91	0.00084	0.00132
E: SDPLS	6	61.03	99.91	0.00085	0.00151
F: PLS	8	99.55	99.86	0.00120	0.00162
G: BPLS	8	13.97	99.89	0.00109	0.00138

Table 6. Comparison between the methods for the density data.

The main results from applying the different methods to this data set are presented in Table 6, including percentage variability explained for the calibration set, the root mean squared error of calibration (RMSEC) and prediction error (RMSEP). For simplicity, the number of latent variables included is defined from that model with the lowest RMSEP.

This multicollinear data set, comprising more variables than observations, is a problem for the EBPLS algorithm, due to the need to calculate the pseudo-inverse in Equation (2.53). This results in overfitting, i.e. a low RMSEC value and a high RMSEP, Table 6. The difference between the behaviour of the calibration data set and the prediction data set for the first latent variable t-u plot and the structure of the corresponding weight vector confirm this behaviour, Figure 4.12(C). The observations are more widely spread for the validation data set.

Since the data is weakly nonlinear (Figure 4.12A-E), the reference method of linear PLS has only slightly lower prediction ability than the two best approaches, NPLS and RVPLS. Furthermore, by fitting a nonlinear function between the predicted and measured density from linear PLS (BPLS), the prediction error is reduced even further, only surpassed in performance by Nested PLS. The satisfactory nonlinear PLS methods for this data set were RVPLS, NLPS and SDPLS.

It should be noted that Framework 2 (RVPLS and SPLS) generally explains more of the **X** variance than Framework 1 (EBPLS, NPLS). The explained variance of the predictor matrix is not directly comparable for the two frameworks, since in Framework 2, each weight is estimated independently between each corresponding variable and **u**, i.e. the covariance information is not used. In Framework 1, the resulting weight vector can be seen more as a regression vector since it represents the least squares minimization results of the objective function, Equation (2.49), for each of the three methods is applied. For the same reasons, the number of latent variables utilized is not directly comparable between the two frameworks. However, within the individual frameworks the results are comparable, i.e. RVPLS explains less variance of the **X** matrix than SPLS, as it focuses more on explaining the response than SPLS. Furthermore, for this framework, the number of latent variables and the variance captured of **X** can be compared with linear PLS. For Framework 1, the **X** variance explained can be compared with that by BPLS, where the regression coefficient is used to estimate the variance of **X** described. The variance explained for BPLS (14%) is comparable to EBPLS (16%) and NPLS (15%), whilst SDPLS (61%) has a much higher value due to convergence issues.

The final SDPLS model includes six latent variables and has the highest amount of **X** variance of Framework 2, but the RMSEP value is satisfactory compared with the reference methods. Although the Nested PLS algorithm describes the smallest amount of the **X** variance and only uses one latent variable, it has the best prediction ability. The RVPLS algorithm demonstrates similar prediction performance to the reference method of BPLS, and improved performance over SPLS which utilizes the same framework and the same number of latent variables. SPLS

explains higher variance of the predictor matrix, but as the prediction results show it gives an inferior model. Thus, for the nonlinear case, the variance captured by **X** is not correlated with the prediction results.

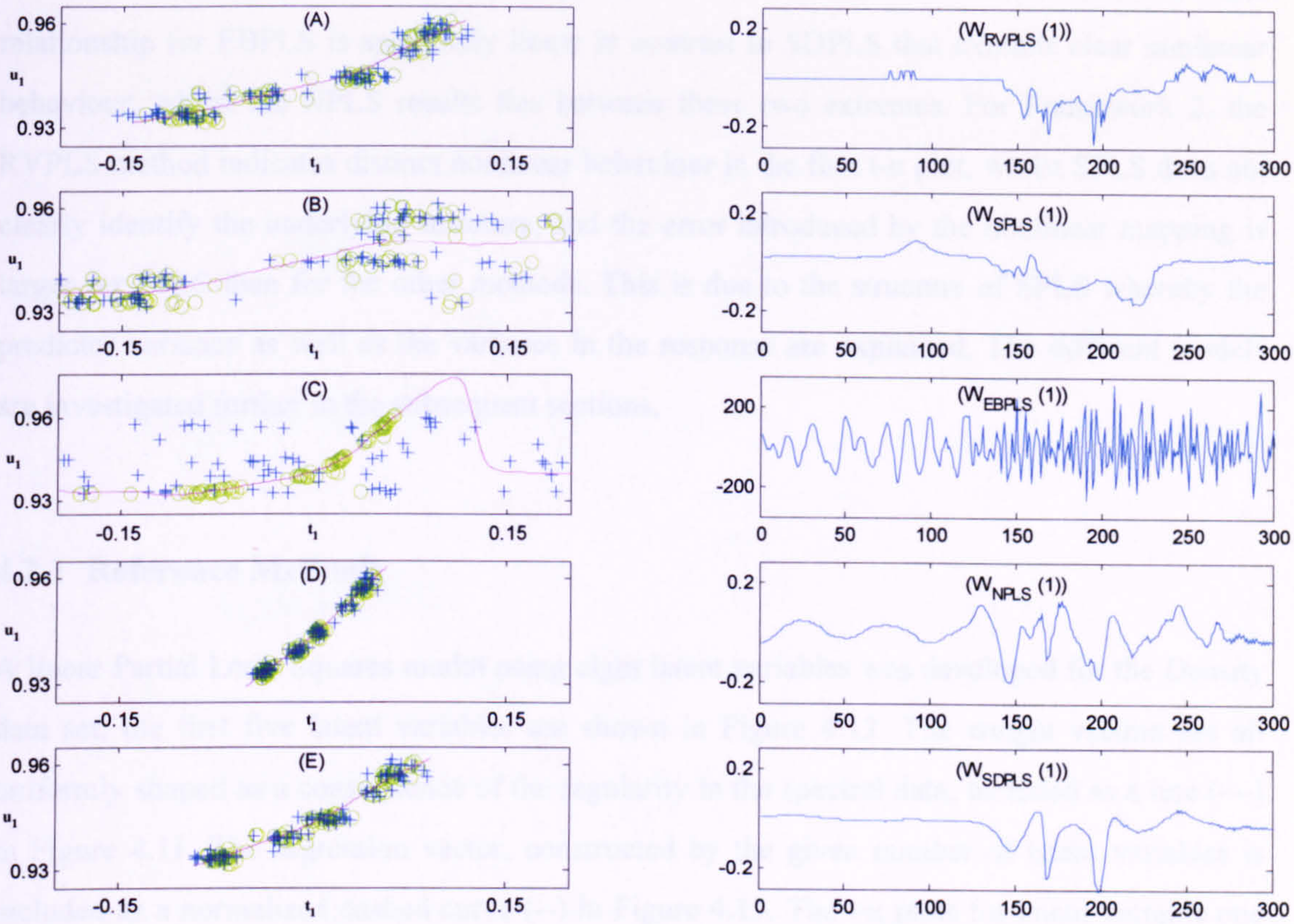

Figure 4.12. Plot of t_1-u_1 scores and the corresponding weight vector for the density data

In Figure 4.12, the first pair of latent variables are plotted for the various nonlinear PLS methods. The corresponding weight vectors are plotted to the right of the t-u plots. A high level of correspondence between the prediction ability of the first latent variable and the final model was established. That is, the method that identified the underlying structure best using the first latent variable (t-u plot) generally had the best prediction ability for the final model, regardless of the number of latent variables included in the model. Therefore, it can be concluded that the first latent variable can provide information about the performance of the methods. Consequently, for the satisfactory methods, RVPLS, NPLS and SDPLS the underlying structure is acceptable modelled, Figure 4.12. In particular, the complexity of the first weight vector calculated using the Nested PLS approach is a balance between the weights for the Error Based PLS and the Steepest Descent PLS approaches, Figure 4.12. The weight vector obtained applying EBPLS is clearly overfitting as observed from the noisy structure, resulting in a large difference between the fit of the calibration and the validation set. The weight vector created by the Reciprocal Variance PLS method exhibits some similarity to the weight vector obtained using SDPLS.

The different methods describe different levels of nonlinearity in the t-u plot for the first latent variable. This difference in nonlinearity originates mainly from possible underfitting of the nonlinear mapping, $f(\mathbf{t})$, briefly discussed in Section 3.2.5.2. In Framework 1, the inner relationship for EBPLS is apparently linear in contrast to SDPLS that exhibits clear nonlinear behaviour, whilst the NPLS results lies between these two extremes. For Framework 2, the RVPLS method indicates distinct nonlinear behaviour in the first t-u plot, whilst SPLS does not clearly identify the underlying structure, and the error introduced by the nonlinear mapping is larger for SPLS than for the other methods. This is due to the structure of SPLS whereby the predictor variance as well as the variance in the response are explained. The different models are investigated further in the subsequent sections.

4.2.1 Reference Methods

A linear Partial Least Squares model using eight latent variables was developed for the Density data set, the first five latent variables are shown in Figure 4.13. The weight vectors are all uniformly shaped as a consequence of the regularity in the spectral data, included as a line (—) in Figure 4.11. The regression vector, constructed by the given number of latent variables is included as a normalized dashed curve (--) in Figure 4.13. The t-u plots for latent variable one and two indicates nonlinearity, for the subsequent it is not so apparent.

LV's	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	40.59	78.68	0.00423	0.00428
2	92.15	89.91	0.00291	0.00313
3	97.87	93.31	0.00237	0.00265
4	98.83	97.28	0.00177	0.00215
5	99.13	98.51	0.00145	0.00177
6	99.36	98.81	0.00136	0.00175
7	99.48	99.00	0.00130	0.00170
8	**99.55**	**99.86**	**0.00120**	**0.00162**
9	99.61	99.91	0.00117	0.00167

Table 7. Results of PLS applied to density data

Due to the relatively weak nonlinearity present in this data set, it is believed that PLS would perform well, confirmed from the consistency between the calibration and the validation data points in the t-u plot. An RMSEC = 0.0012 and a RMSEP = 0.0016 from Table 7 confirms this belief. The RMSEC values are lower than the RMSEP values, with the difference for the final

model being 26%, i.e. the level of overfitting does not appear significant. The high number of 8 latent variables is needed to create the best model. It has been observed that when the data is nonlinear, PLS need a high number of latent variables to be included (Martens and Næs, 1989).

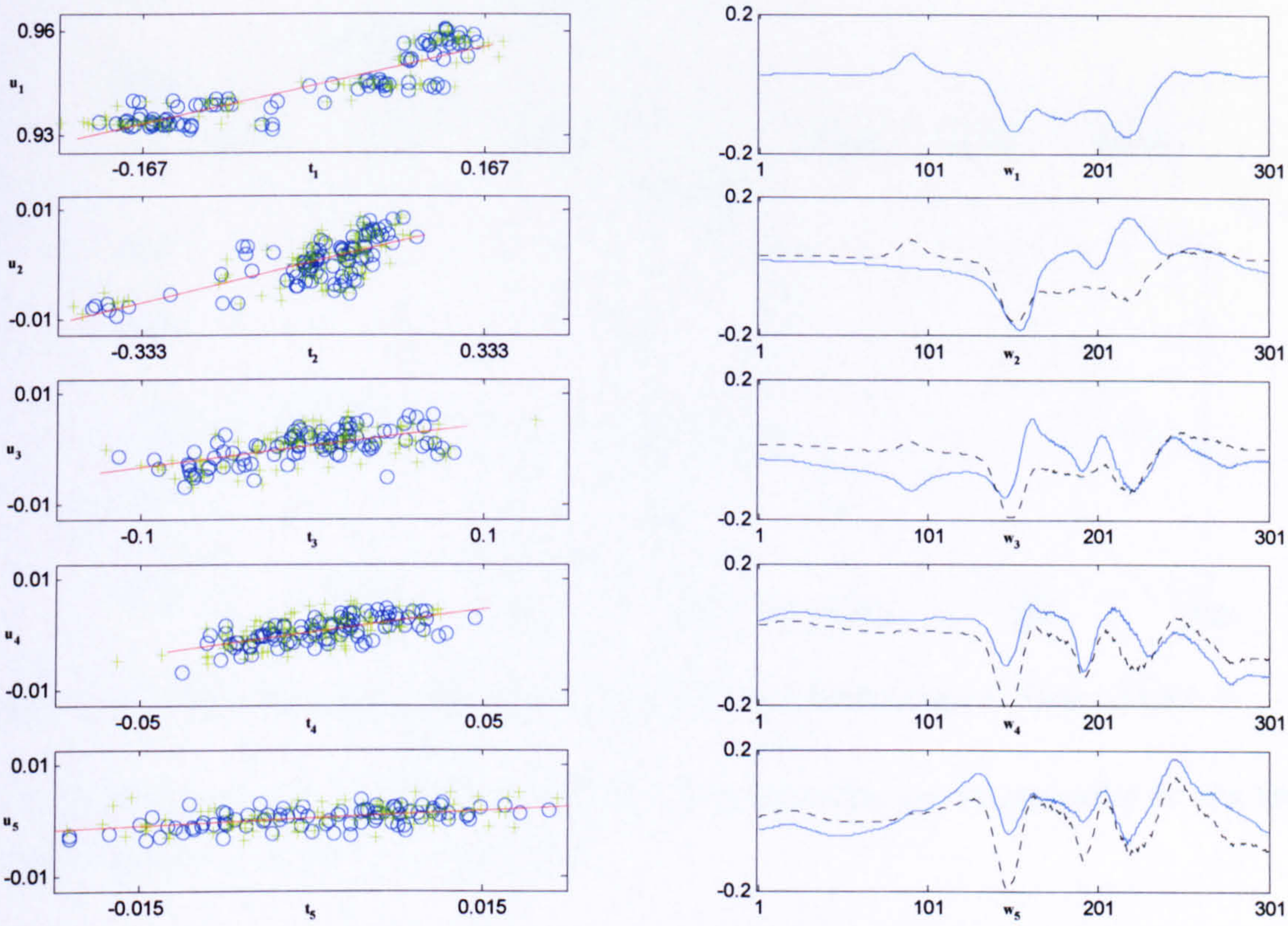

Figure 4.13. The t-u, weights and regression vectors for PLS for the first five latent variables.

If nonlinear behaviour is present in the data, this can often be observed by plotting the measured response versus the predicted response from PLS, that is the measured density is plotted agains predicted density. BPLS uses the regression vector obtained from linear PLS model developed from eight latent variables, Figure 4.14. This regression vector is then used as the first and only weight vector in the general nonlinear PLS algorithm, Appendix A2.5. Thus the nonlinear mapping between **u** and **t** = **Xb**, is constructed using standard local linear kernel regression, included as a line (—) in Figure 4.14, upper plot). The regression vector developed from eight latent variables is shown in the lower plot, Figure 4.14.

By including a nonlinear model between the predicted and the measured response (BPLS), the model error decreases by 15% compared to linear PLS, resulting in a RMSEC = 0.0011 and a RMSEP = 0.0014, Table 6(G). In particular, the prediction ability at the extremes of the t-u plot was improved, compared with applying a linear relation in the t-u plot (PLS), Figure 4.14, upper plot.

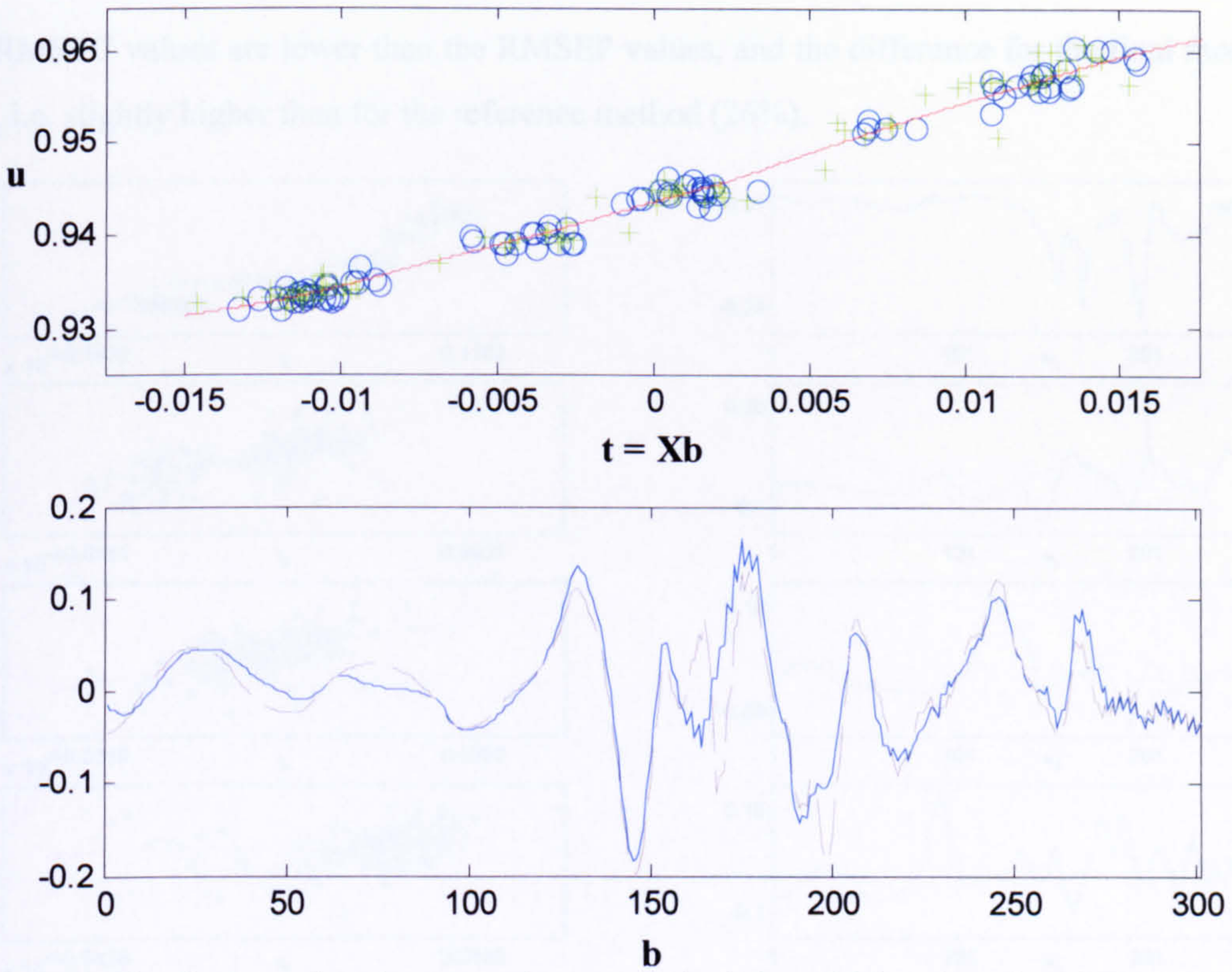

Figure 4.14. The final t-u plot, the regression vector (—) from PLS and w_1 from NPLS (--).

Of particular note, the shape of the regression vector resembles the weight vector for the first latent variable of Nested PLS, Figure 4.14.

4.2.2 Reciprocal Variance PLS

This method identifies the nonlinear relationship primarily through the first pair of latent variables, consequently subsequent pairs of latent variables exhibit an approximately linear relationship, Figure 4.15. However, there is still structure present as observed from the latent variables two and three and the reduction in the Root Mean Square Error of Prediction (RMSEP) values. The weight vectors are relatively smooth and exploit different parts of the spectrum. The variances captured and the prediction performance captured by the first six latent variables is summarized in Table 8.

LV's	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	18.19	99.77	0.00220	0.00241
2	28.52	99.86	0.00136	0.00192
3	33.84	99.89	0.00102	0.00158
4	**60.33**	**99.90**	**0.00096**	**0.00152**
5	63.21	99.90	0.00093	0.00153
6	63.45	99.91	0.00089	0.00153

Table 8. Results of RVPLS applied to density data.

The RMSEC values are lower than the RMSEP values, and the difference for the final model is 37%, i.e. slightly higher than for the reference method (26%).

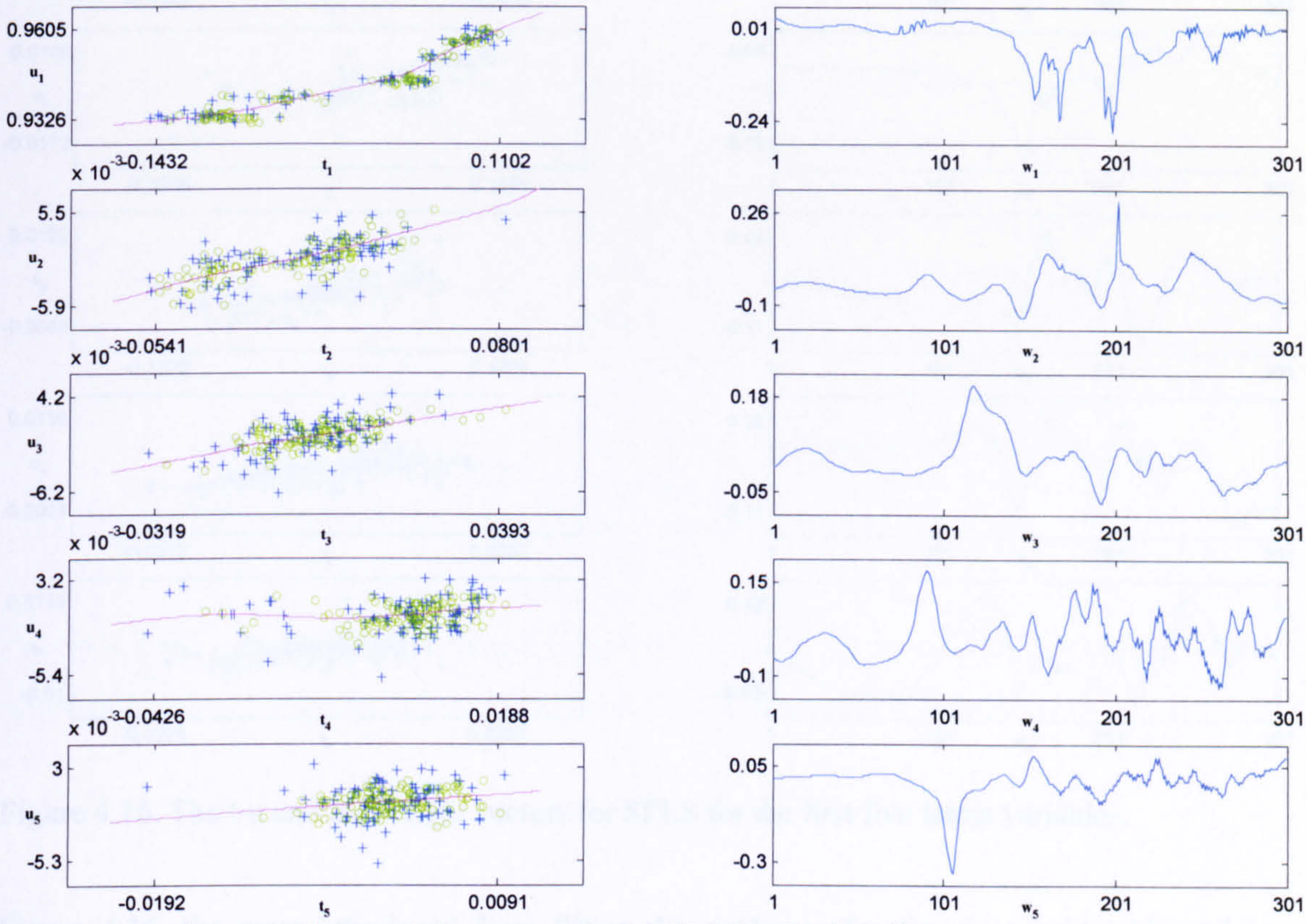

Figure 4.15. The t-u and the weight vectors for RVPLS for the first five latent variables.

4.2.3 Spline PLS

This method does not identify the underlying nonlinear relationship from the first pair of latent variables as for RVPLS, due to the badly chosen weight vector. Furthermore, the nonlinear relationship between subsequent pairs of latent variables are difficult to model due to the error introduced from the rank one model of **Y**, introduced by the first nonlinear mapping function.

LV's	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	33.54	99.39	0.00572	0.00565
2	71.38	99.69	0.00293	0.00386
3	85.61	99.76	0.00228	0.00320
4	**88.84**	**99.80**	**0.00192**	**0.00292**
5	90.36	99.84	0.00155	0.00303
6	91.13	99.85	0.00143	0.00295

Table 9. Result of SPLS applied to density data

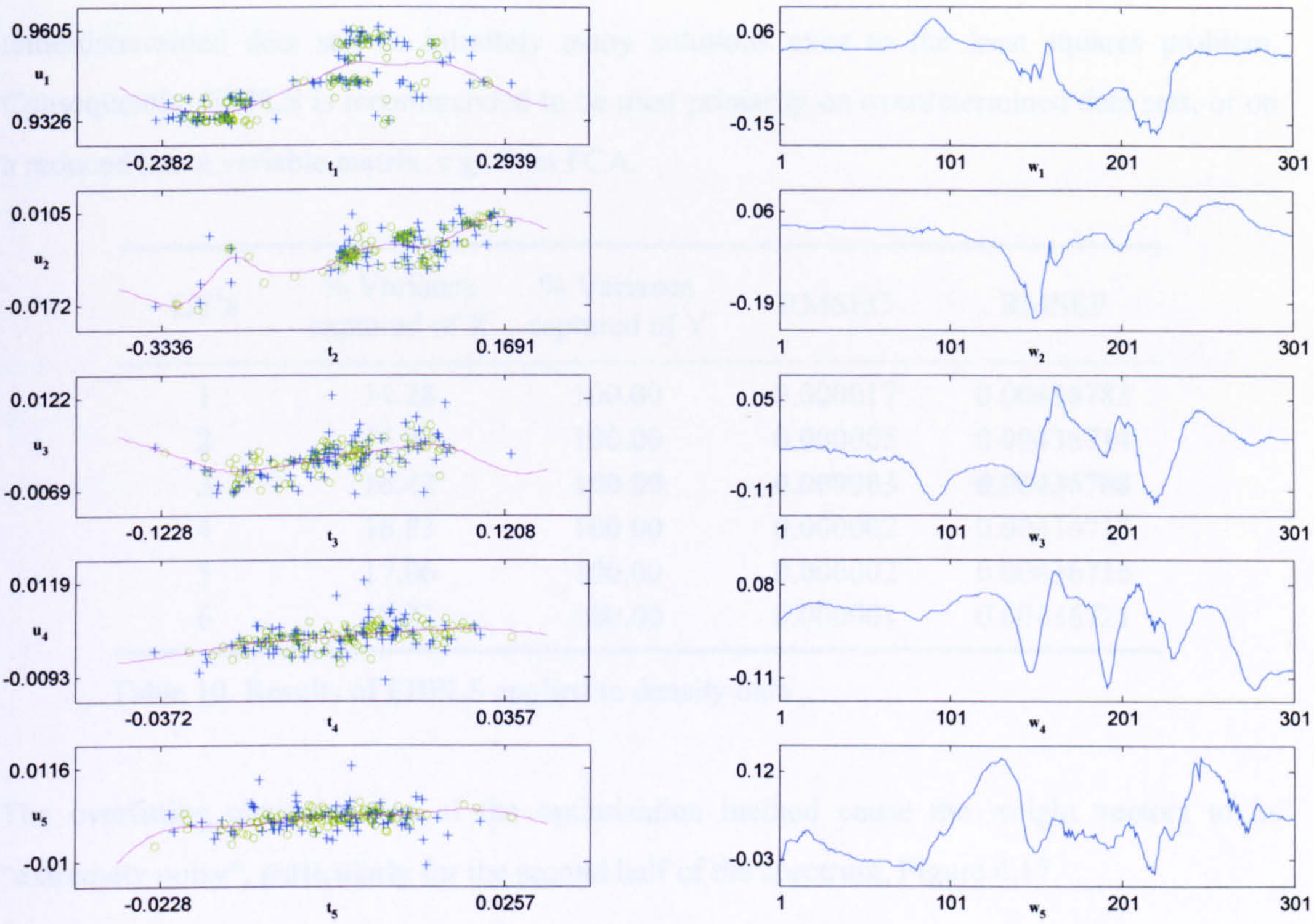

Figure 4.16. The t-u and the weight vectors for SPLS for the first five latent variables.

Figure 4.16, the error introduced from fitting the nonlinear function is considerable and is propagated through to subsequent latent variables, thus impacting on the potential of the method. In particular, the second pair of latent variables ($\mathbf{t}_2$-$\mathbf{u}_2$) does not identify the underlying structure. However, the difference between the RMSEC and RMSEP values is 35% for the 4 latent variable model, Table 9, and the weight vectors are smooth. The main limitation of the method is the impact of the error originating from the inner mapping on the overall model, due to the focus of simultaneously modelling both the variance in **X** and **Y** when constructing the weight vector. Consequently, the first latent variable captures almost double the X-variance, 34%, compared with that of RVPLS, 18%.

4.2.4 Error Based PLS

The method of EBPLS clearly overfits this underdetermined data set. Although the calibration data set is modelled well, the validation data set is not. The difference between the RMSEC and the RMSEP values is high, 99%, for a three latent variable model. This is also observed from the t-u plot for the first latent variable, Figure 4.17. Even though the third latent variable has the smallest RMSEP value, very little is gained compared to the case where only one latent variable is used in the development of the method. The reason for the overfitting is due to the

underdetermined data set, as infinitely many solutions exist to the least squares problem. Consequently, EBPLS is recommended to be used primarily on overdetermined data sets, or on a reduced latent variable matrix, e.g. from PCA.

LV's	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	14.28	100.00	0.000017	0.00436783
2	15.93	100.00	0.000005	0.00436714
3	**16.42**	**100.00**	**0.000003**	**0.00436708**
4	16.83	100.00	0.000002	0.00436711
5	17.06	100.00	0.000002	0.00436716
6	17.23	100.00	0.000001	0.00436721

Table 10. Results of EBPLS applied to density data

The overfitting characteristics of the optimization method cause the weight vectors to be "extremely noisy", particularly for the second half of the spectrum, Figure 4.17.

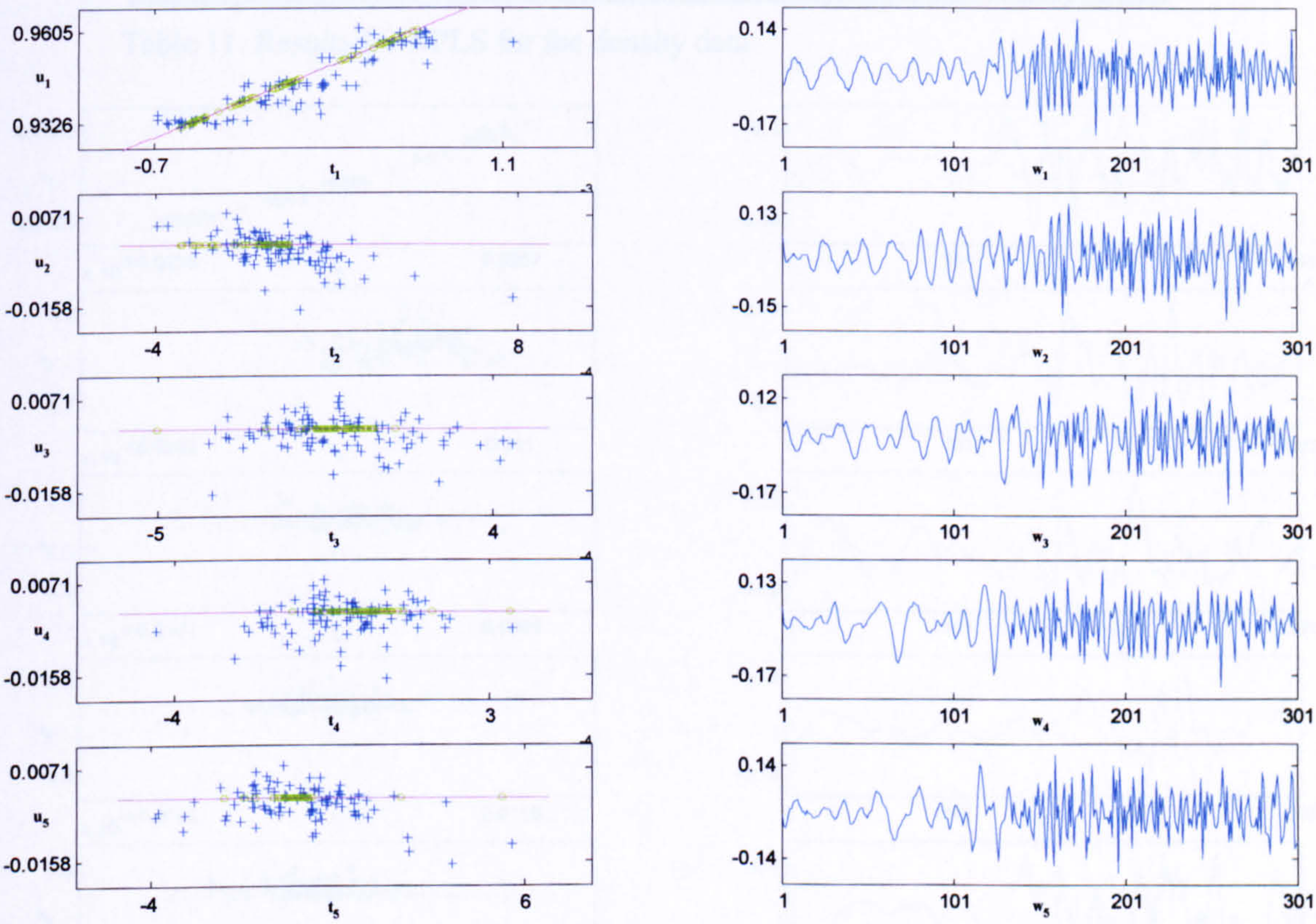

Figure 4.17. The t-u and the weight vectors for EBPLS for the first five latent variables.

4.2.5 Nested PLS

Nested PLS generates a weight vector that identifies an acceptable underlying structure between the first pair of latent variables, that bears a strong resemblance with the regression vector from PLS, Figure 4.14. For the methods included in Framework 1, this is desirable when the underlying structure is close to being linear as for this data set. It indicates that the nonlinear extension of PLS performs similar to linear PLS. The difference between the RMSEC and the RMSEP values is similar to that for RVPLS and SPLS (36% lower RMSEC) but higher than for the reference method of PLS (26%), i.e. thus some overfitting is indicated.

LV's	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	**14.90**	**99.91**	**0.00084**	**0.00132**
2	18.99	99.92	0.00071	0.00174
3	22.02	99.93	0.00066	0.00176
4	29.59	99.94	0.00059	0.00175
5	37.88	99.94	0.00054	0.00171
6	44.10	99.95	0.00046	0.00171

Table 11. Results for NPLS for the density data

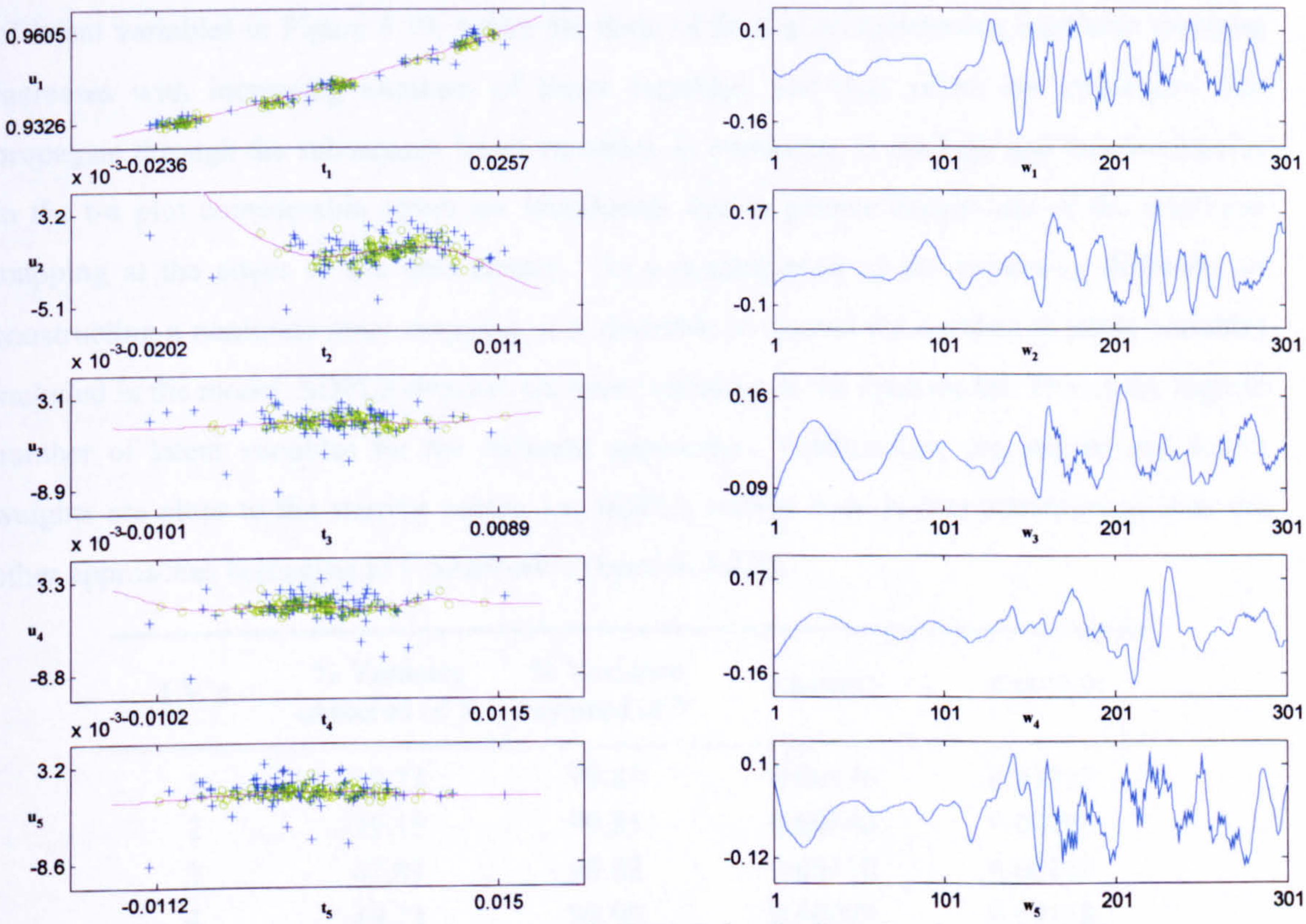

Figure 4.18. The t-u and the weight vectors for NPLS for the first five latent variables.

From Figure 4.18, the second latent variable displays no real structure with the calibration error being only slightly smaller when included in the model. For this two latent variable model, the

prediction error is strongly affected by 4 observations as observed in the second t-u plot (t_2-u_2) for latent variable two. Examining the plots of the weight vectors on the right hand side of Figure 4.18, the weight vectors are generally smooth and do not appear to be notably affected by the noise in the data set. The noise observed in the weight vectors is of the same order as for the RVPLS method that models each weight vector independently. Thus it can be concluded that the NPLS method produces weight vectors with reasonable smoothness, signifying that the approach is not subjected to significant overfitting for this data set.

4.2.6 Steepest Descent PLS

For this data set SDPLS appears to overfit compared with NPLS as quantified by the deviation between the RMSEC and the RMSEP (44% lower for RMSEC), due to the greater number of latent variables required to explain the underlying structure. The weight vectors are smooth and have a more simple structure than for NPLS, due to premature termination since no covariance information is used in the construction of the step direction. Even though the underlying structure is determined by the first pair of latent variables, prediction ability is poorer than for the second reference method, BPLS. This can be understood by examining the subsequent pair of latent variables in Figure 4.19, where the issue of finding an appropriate nonlinear mapping increases with increasing numbers of latent variables, and thus errors are introduced that propagate through the subsequent latent variables. In particular, at the high and low boundaries in the t-u plot considerable errors are introduced, due to greater uncertainty of the nonlinear mapping at the edges of the data cluster. As a consequence of the increasing difficulty of constructing a nonlinear inner mapping, it is desirable to restrict the number of latent variables included in the model. SDPLS requires six latent variables in the final model. This is the highest number of latent variables for the different approaches. Furthermore, the second and fourth weights are close to the starting vector, i.e. SDPLS suffers from poorer convergence than the other approaches belonging to Framework 1 (Section 4.2.7).

LV's	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	15.73	99.83	0.00156	0.00212
2	39.15	99.85	0.00140	0.00195
3	42.03	99.88	0.00110	0.00160
4	49.23	99.90	0.00099	0.00158
5	56.89	99.90	0.00092	0.00158
6	**61.03**	**99.91**	**0.00085**	**0.00151**
7	64.97	99.91	0.00082	0.00154

Table 12. SDPLS on density data

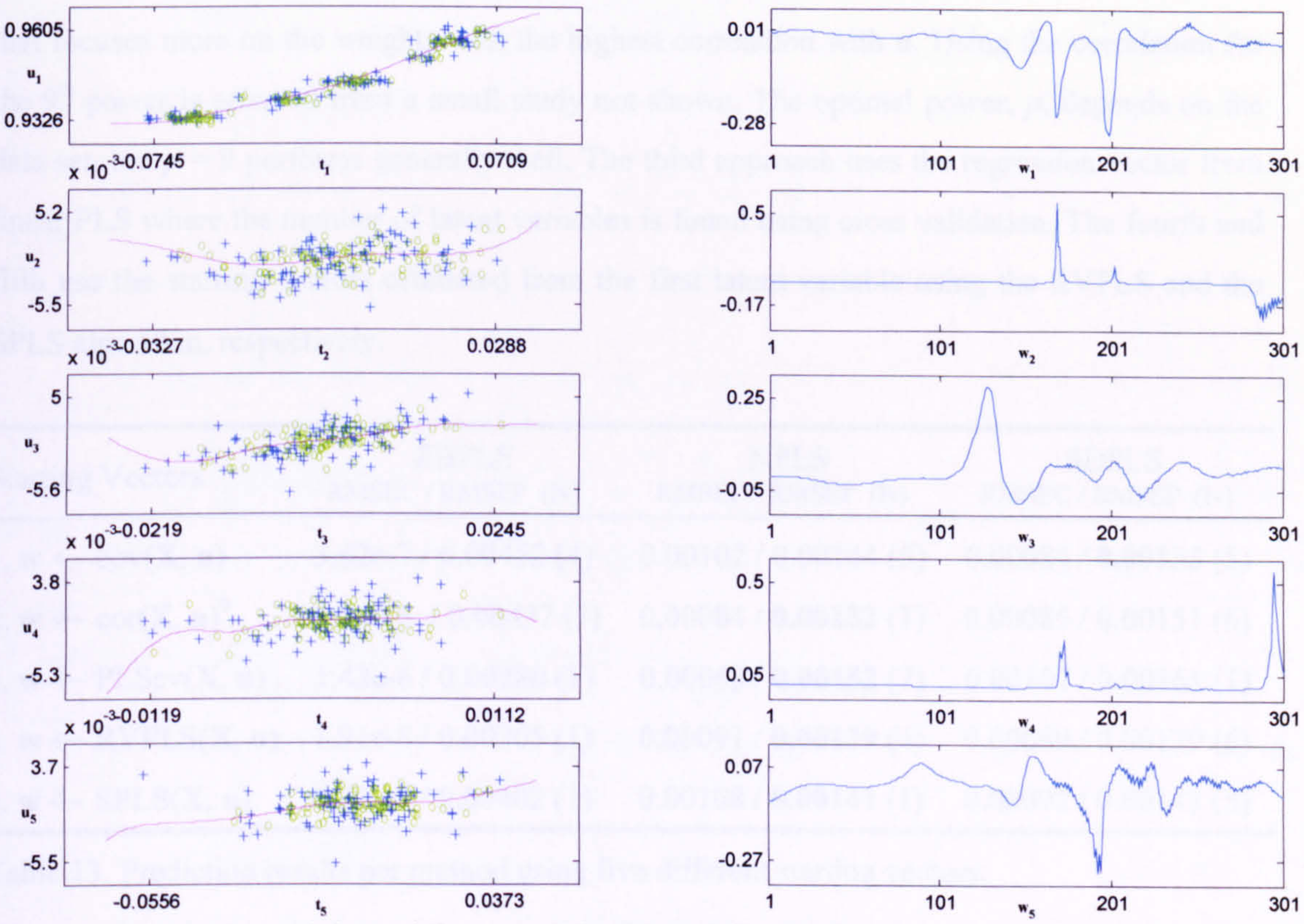

Figure 4.19. The t-u and the weight vectors for SDPLS for the first five latent variables.

4.2.7 Influence of the Starting Vector

The influence of the starting vector was investigated for the methods included in Framework 1, since the different starting vectors were found to have little influence on Framework 2, i.e. the methods of SPLS and RVPLS. This is due to the methodology of Framework 2, in particular the individual calculations of the estimates of the weights not using an updating vector.

The prediction ability of the methods and the spread of the first weight vector were investigated, by applying five different starting vectors. The spread is evaluated by comparing the deviation between the five weight vectors obtained from applying the five different starting vectors, for each of the nonlinear PLS method investigated. In particular, the weight vector, **w**, is that calculated, from the given starting vector, to give the first score vector **t**, such that $\mathbf{t} = \mathbf{Xw}$. Large deviations between the weight vectors for the five starting vectors will indicate a tendency for the method to terminate in local minima far away from the global minimum, whilst a small deviation between the final models will indicate robustness towards early termination in a local minimum.

In Table 13, the first starting vector is the standard starting vector used in linear PLS and is identical to the first weight vector when there is only one response variable for linear PLS. The second is the standard choice used in the comparison. It is a variation of the first starting vector

that focuses more on the weights with the highest correlation with **u**. Using the correlation for the 9th power is selected from a small study not shown. The optimal power, p, depends on the data set, but $p = 9$ performs generally well. The third approach uses the regression vector from linear PLS where the number of latent variables is found using cross validation. The fourth and fifth use the starting vectors estimated from the first latent variable using the RVPLS and the SPLS algorithm, respectively.

Starting Vectors	EBPLS RMSEC / RMSEP (lv)	NPLS RMSEC / RMSEP (lv)	SDPLS RMSEC / RMSEP (lv)
1. **w** ← cov(**X**, **u**)	3.62e-7 / 0.00452 (4)	0.00107 / 0.00144 (5)	0.00084 / **0.00135** (5)
2. **w** ← cor(**X**, **u**)^9	9.86e-6 / 0.00437 (3)	0.00084 / **0.00132** (1)	0.00085 / 0.00151 (6)
3. **w** ← PLScv(**X**, **u**)	1.42e-6 / 0.00280 (1)	0.00063 / **0.00152** (7)	0.00107 / 0.00161 (1)
4. **w** ← RVPLS(**X**, **u**)	1.91e-8 / 0.00305 (1)	0.00091 / **0.00139** (4)	0.00089 / 0.00139 (6)
5. **w** ← SPLS(**X**, **u**)	6.22e-7 / 0.00402 (1)	0.00108 / **0.00141** (1)	0.00092 / 0.00143 (5)

Table 13. Prediction results per method using five different starting vectors.

The different starting vectors and the prediction results are presented in Table 13. The overall ranking of the prediction ability for the three methods using the five different starting vectors are independent of the choice of starting vector. However, SDPLS performed best for the first staring vector. Furthermore, the number of latent variables varies considerably, although the first latent variable explains the largest amount of response variance.

SDPLS appears to have greatest dependency on the starting vector, whilst EBPLS is least reliant on the starting vector, calculated from the average correlation, i.e. $\sum cor(w_k, w_i)/10$ where $k \neq i \in \{1,2,\cdots,5\}$, between the five weight vectors. In particular, the average correlation between the five obtained weight vectors in Figure 4.20 are 0.97 ± 0.03 for the EBPLS method, 0.92 ± 0.06 for the NPLS method, and 0.64 ± 0.24 for the SDPLS method. The performance of the starting vector estimated by **w** = PLScv(**X**, **u**) differs from the other starting vectors. It gives the lowest average correlation with the other vector methods for all three PLS methods (EBPLS: 0.94 ± 0.02, NPLS: 0.87 ± 0.04, SDPLS: 0.40 ± 0.03). One interpretation could be that by using this starting vector the possibility to terminate in a local minimum is increased, as this starting vector represents a linear solution (the regression vector) to a nonlinear situation.

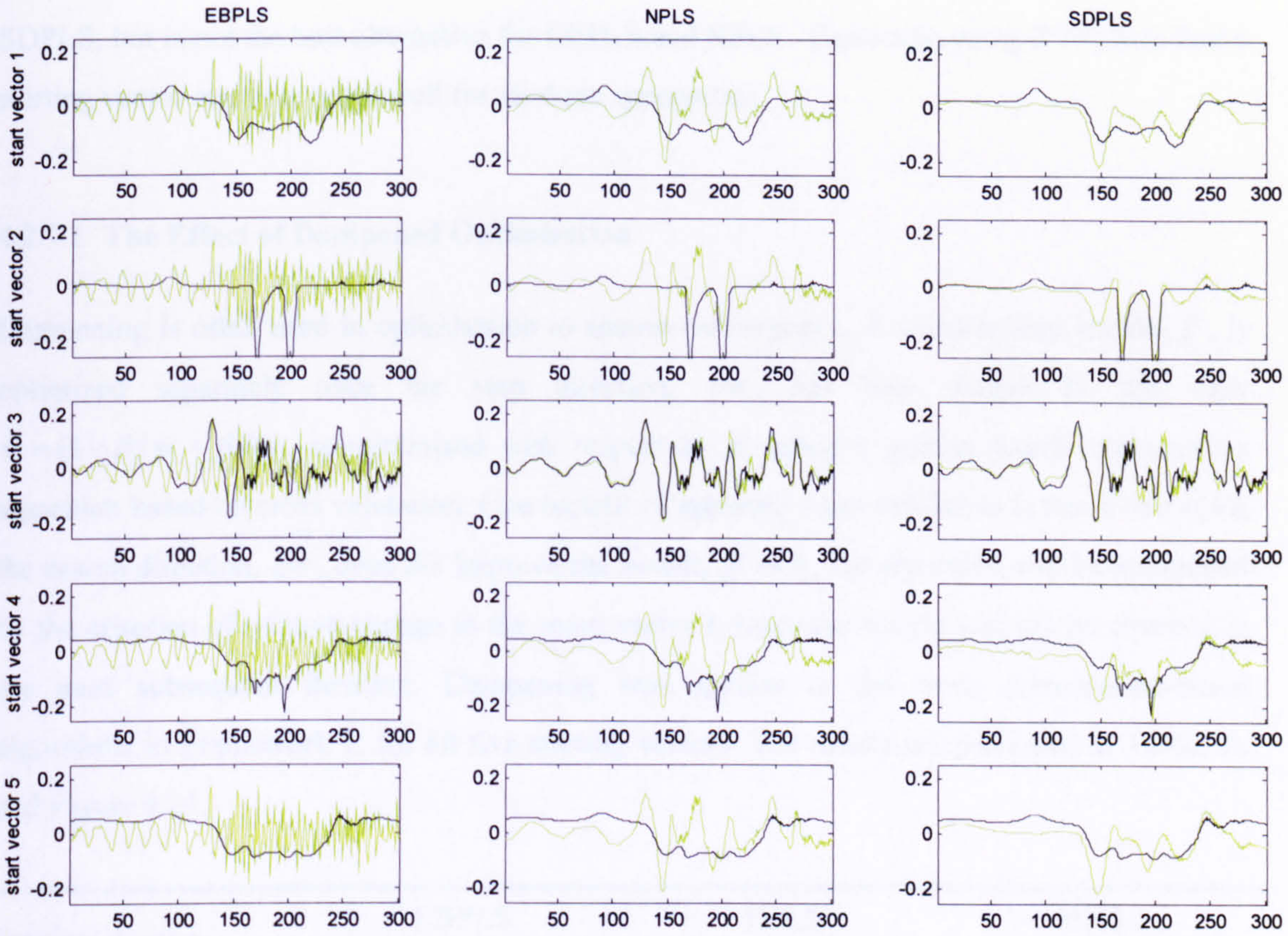

Figure 4.20. The starting vector (black) and the final weights (green) for the first latent variable.

To summarise, starting vector 5 (SPLS) is similar to starting vector 1 (cov(**X**, **u**)), whilst the first weight vector from Nested PLS is similar to the starting vector 3 (PLScv(**X**, **u**)), Figure 4.20. Furthermore, the solution obtained using NPLS is situated between that of EBPLS and SDPLS. Overfitting occur when applying EBPLS, whilst termination in local minima often occur when applying SDPLS.

NPLS obtains a weight vector, independent of the starting vector, that is similar to the starting vector obtained using linear PLS with cross validation, Figure 4.20. It is desirable to achieve a model using nonlinear PLS that does not diverge severely from the linear PLS model, when the underlying structure is approximately linear. Finally, it should be noted that the first and fifth starting vector in Figure 4.20, are similar with a correlation of 0.97 between them due to the relatively weak nonlinearity in the data.

It is difficult to draw any conclusions about which starting vectors performs better than the others. The starting vector obtained using linear PLS with cross validation is similar to the first weight vector using Nested PLS, but using it as a starting vector gave the worst prediction performance for both NPLS and SDPLS. Applying starting vector 1 gives the best model for

SDPLS, but is not the best alternative for EBPLS and NPLS. Generally, using RVPLS to find a starting vector seems to work well for all three approaches.

4.2.7.1 The Effect of Dampened Optimisation

Dampening is often used in optimisation to ensure convergence. A variable step length, β, is optimized separately once the step direction, $\partial \mathbf{w}$, has been found. In this case $\mathbf{w} = (1-\beta)\mathbf{w} + \beta\partial\mathbf{w}$, is minimised with respect to β using a golden search optimisation algorithm based on cross validation. One benefit of applying cross validation is that if including the search direction, $\partial \mathbf{w}$, does not improve the model, $\beta \to 0$, the algorithm will be terminated by the criterion of relative change in the score vector **t**, since the weight will not be changed in the next subsequent iteration. Dampening was applied to the three optimisation-based algorithms in Framework 1, for all five starting vectors. The results are presented in Table 14, and Figure 4.21.

Starting Vectors	EBPLS RMSEC / RMSEP (lv)	NPLS RMSEC / RMSEP (lv)	SDPLS RMSEC / RMSEP (lv)
1. $\mathbf{w} \leftarrow \text{cov}(\mathbf{X}, \mathbf{u})$	0.00195 / 0.00241 (2)	0.00077 / 0.00133 (5)	0.00078 / **0.00131 (7)**
2. $\mathbf{w} \leftarrow \text{cor}(\mathbf{X}, \mathbf{u})$[9]	0.00002 / 0.00383 (7)	0.00077 / **0.00124 (3)**	0.00061 / 0.00148 (9)
3. $\mathbf{w} \leftarrow$ PLScv(**X**, **u**)	0.00000 / 0.00216 (5)	0.00097 / **0.00141 (1)**	0.00102 / 0.00147 (1)
4. $\mathbf{w} \leftarrow$ RVPLS(**X**, **u**)	0.00343 / 0.00337 (1)	0.00069 / **0.00127 (5)**	0.00071 / 0.00138 (9)
5. $\mathbf{w} \leftarrow$ SPLS(**X**, **u**)	0.00175 / 0.00251 (3)	0.00060 / **0.00130 (6)**	0.00103 / 0.00144 (3)

Table 14. Prediction results using different starting vectors when utilizing dampening.

For this data set using damped optimization basically improved all the models, but not significantly. Closer investigation showed that dampening resulted in the more rapid termination of the algorithms. In particular, the EBPLS algorithm terminated after a few iterations (2-3 compared to 25), resulting in a weight vector that retained the basic structure of the starting vector, Figure 4.21. This comes from the use of cross validation in the golden search method used to calculate the step size. If the lowest cross validation error is achieved for $\beta = 0$, when adding the step direction, $\beta\partial\mathbf{w}$, the algorithm will terminate since the weight vector and thereby the score vector, **t**, is not changed. This is a consequence of that one of the termination criteria is given by a relative change in the score vector of less than 10^{-8}, Section 4.1

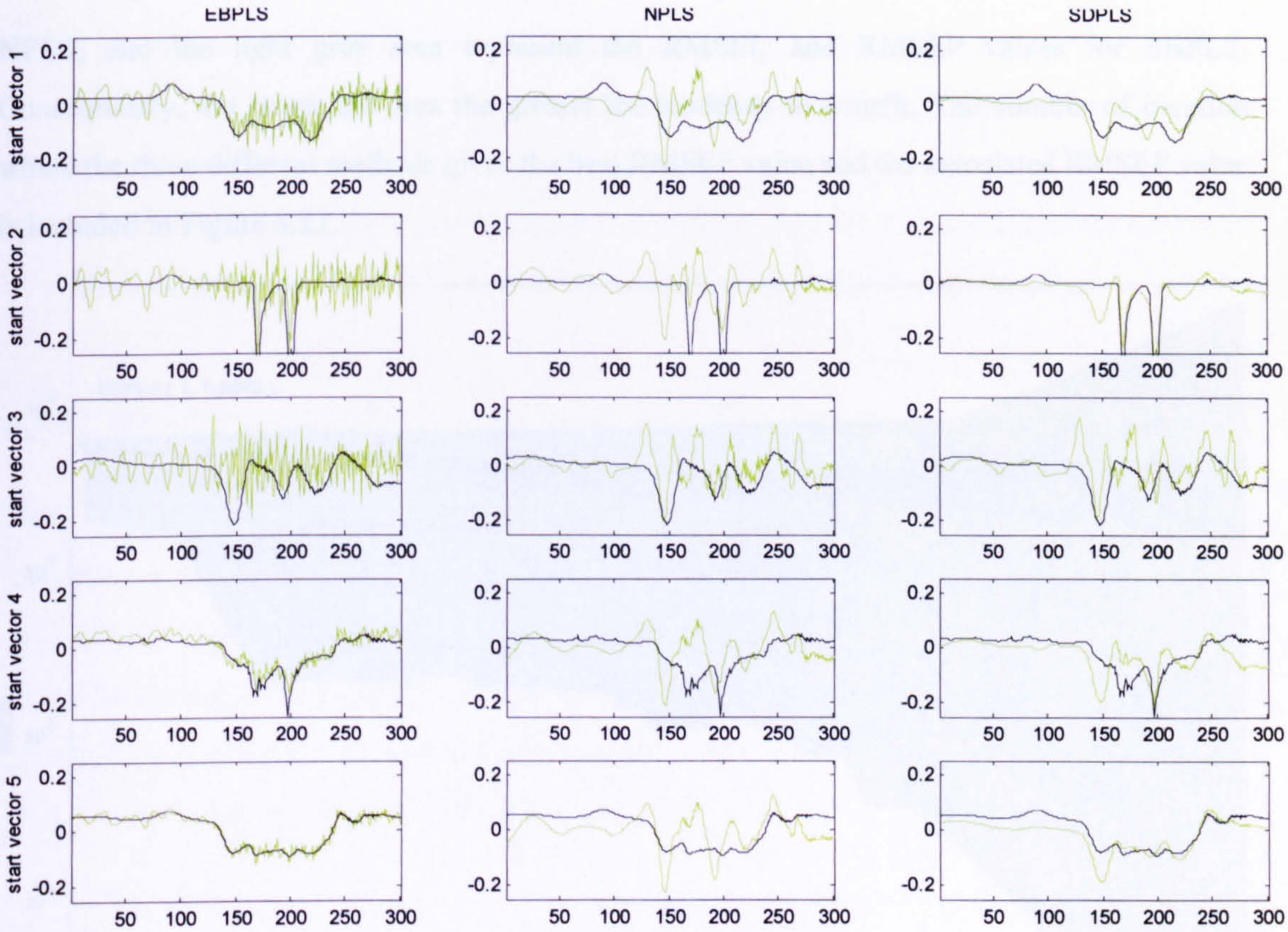

Figure 4.21. The starting vector (black) and the final weights (green) for the first latent variable.

The first weight vector obtained for the five starting vectors shows slightly smother and simpler behaviour compared with when damped optimisation was not used, but the spread between the five weight vectors is larger. The average correlation between the obtained five weight vectors in Figure 4.21 are 0.45 ± 0.35 for EBPLS, 0.87 ± 0.08 for NPLS, and 0.63 ± 0.21 for SDPLS. For the density data set, dampening ensured earlier termination and decreased the tendency of overfitting. However, applying dampened optimisation does not alter the ranking of the algorithms when it comes to prediction performance.

4.2.8 Impact of the Termination Criteria

To investigate the influence of the termination criteria for those methods belonging to Framework 1, the RMSEC and RMSEP values were examined by recording the values for each iteration, for the first latent variable. Except for varying the number of iterations and not using any termination criteria, the standard preferences were retained. The Root Mean Squared Error (RMSE) values are presented in Figure 4.22, with the difference between the RMSEC and the RMSEP values plotted as an area to highlight any tendency to overfit. The darkest area plotted represents the difference in RMSE between the calibration set (RMSEC) and the validation set (RMSEP) for SDPLS, the medium grey area represents the RMSEC and RMSEP values for

NPLS, and the light grey area represent the RMSEC and RMSEP values for EBPLS. Consequently, the larger the area the greater the tendency to overfit. The number of iteration where the three different methods gives the best RMSEP value and the associated RMSEP value is included in Figure 4.22.

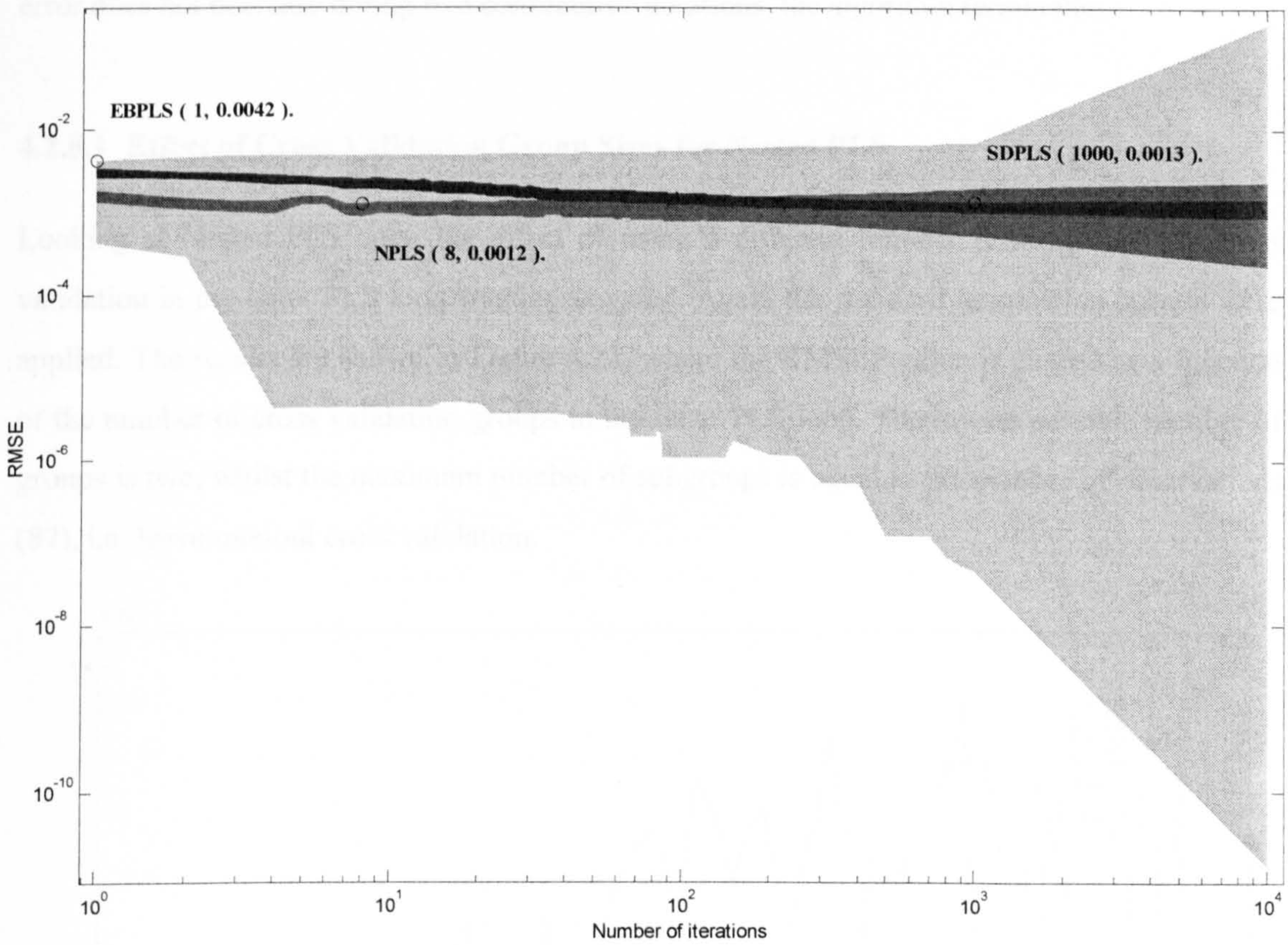

Figure 4.22. Difference between RMSEC and RMSEP for EBPLS(▒), NPLS(■) and SDPLS(■).

All three methods tend to overfit, but this occurs at different stages of the optimisation and with varying consequences, Figure 4.22. For this data set, EBPLS tends to overfit from the first iteration and the difference between the values of RMSEC and RMSEP increases rapidly with the number of iterations. For Nested PLS, overfitting starts after approximately ten iterations, whilst for SDPLS overfitting happens after approximately one thousand iterations. But for NPLS and SDPLS the overfitting is less pronounced, as observed from the small area between the RMSEC and the RMSEP values.

For NPLS, generally a higher number of latent variables is selected in the inner PLS at the beginning of the iterations than later on. For this data set, after ten iterations only one latent variable was selected in the inner PLS. Thus, from that point on NPLS and SDPLS would have performed equally.

NPLS has the overall lowest prediction error of all the methods independent of the number of iterations, but only slightly better than SDPLS. In particular, SDPLS is most affected using the standard termination criteria, defined as the basis of the study (Section 4.1). However, the standard termination criteria seems to prevent overfitting for NPLS, in particular the rule if the error does not decrease during two consecutive iterations, the algorithm terminates.

4.2.8.1 Effect of Cross Validation Group Sizes for Nested PLS

Looking at Nested PLS only, the effect of using a different number of subgroups for cross validation in the inner PLS loop was investigated. Again the standard termination criteria were applied. The results are shown in Figure 4.23, where the RMSEP value is plotted as a function of the number of cross validation groups in the inner PLS loop. The lowest possible number of groups is two, whilst the maximum number of subgroups is equal to the number of observations (87), i.e. leave-one-out cross validation.

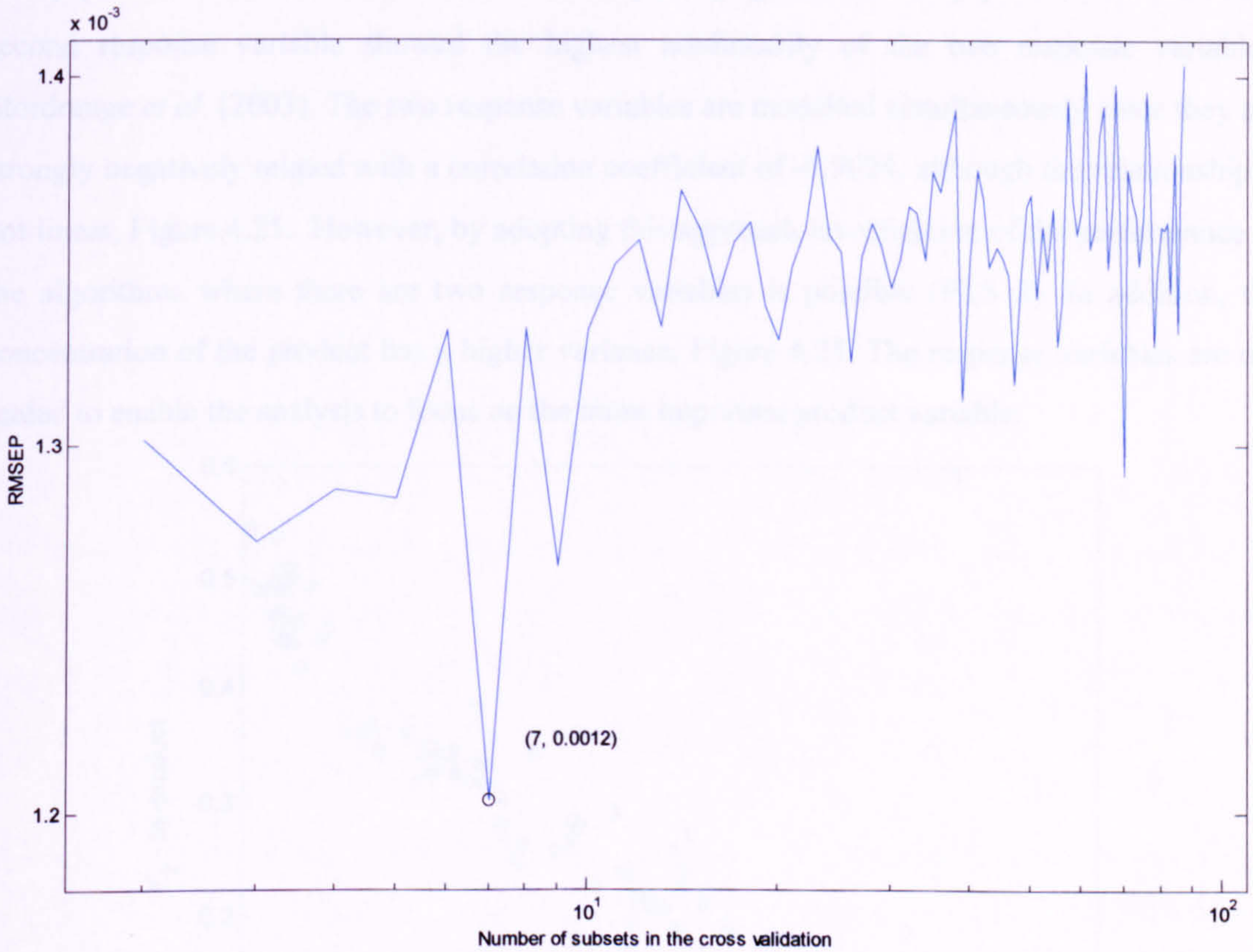

Figure 4.23. RMSEP for NPLS for different cross validation groups sizes.

The lowest average RMSEP value was obtained using 7 subsets. In general the results suggest that using a small number of subsets is beneficial. This is in agreement with Wold (1978) who reported that the optimal number of subsets lies between 4 and 11. In a separate test applying dampening within the NPLS algorithm, not shown here, the lowest RMSEP (0.0012) was obtained using 4 subsets in the cross validation.

4.3 Pharmaceutical Process Data using Near Infrared Spectroscopy.

Process data from a pharmaceutical batch synthesis is investigated in this Section. The main reactant reacts with the alkylating agent, R_3, to form the product. Furthermore, a by-product can be formed by further reaction with the alkylating agent, Figure 4.24.

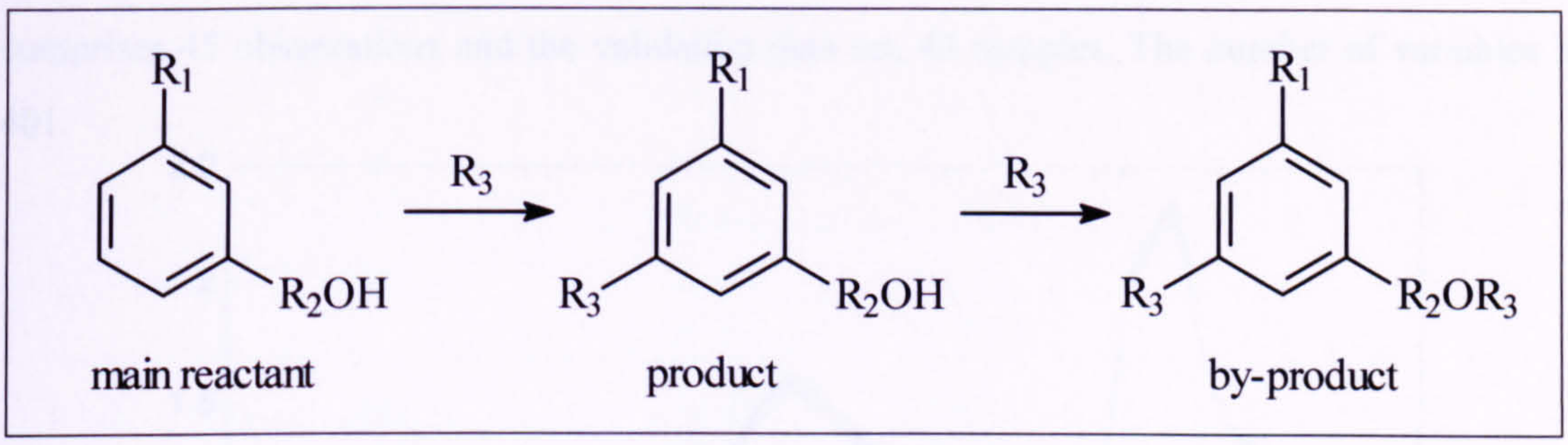

Figure 4.24. The Alkylation Process.

The nonlinearity materializes from the relationship between the concentration of the product and the by-product with the concentration of the alkylating agent, R_3. The by-product defined as the second response variable showed the highest nonlinearity of the two response variables, Stordrange *et al*. (2003). The two response variables are modelled simultaneously since they are strongly negatively related with a correlation coefficient of -0.9024, although the relationship is not linear, Figure 4.25. However, by adopting this approach investigation of the performance of the algorithms where there are two response variables is possible (PLS 2). In addition, the concentration of the product has a higher variance, Figure 4.25. The response variables are not scaled to enable the analysis to focus on the more important product variable.

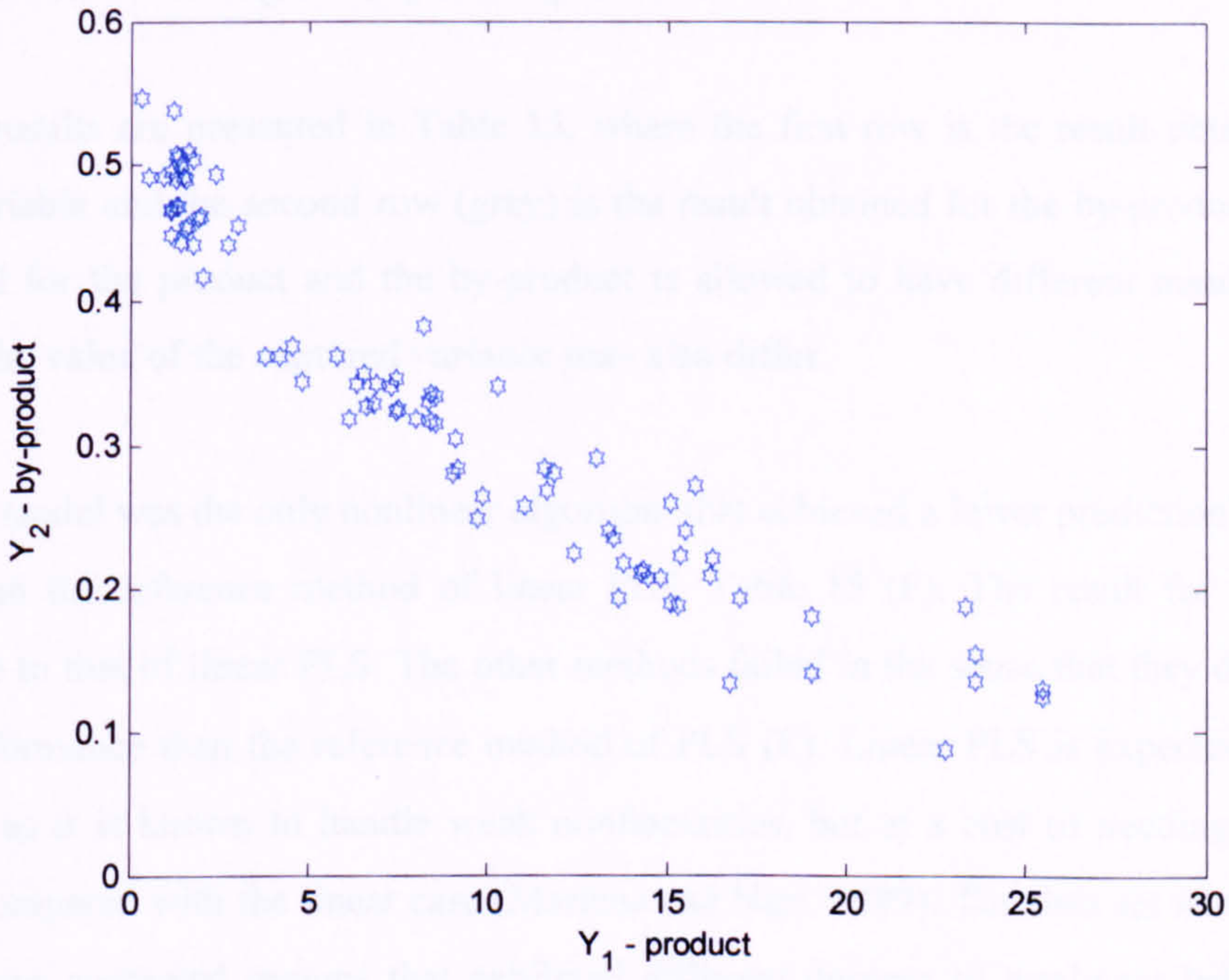

Figure 4.25. Scatter plot of product versus by-product.

The near infrared spectra were recorded using a transreflectance probe, whilst the concentration of the product and the by-product were determined using High Pressure Liquid Chromatography (HPLC), Figure 4.26. The near infrared spectra were pre-processed using Multiplicative Scatter Correction (MSC) as this pre-processing technique generally performs well (Martens and Næs, 1989). The data set is underdetermined and is highly collinear. The calibration data set comprises 45 observations and the validation data set, 43 samples. The number of variables is 401.

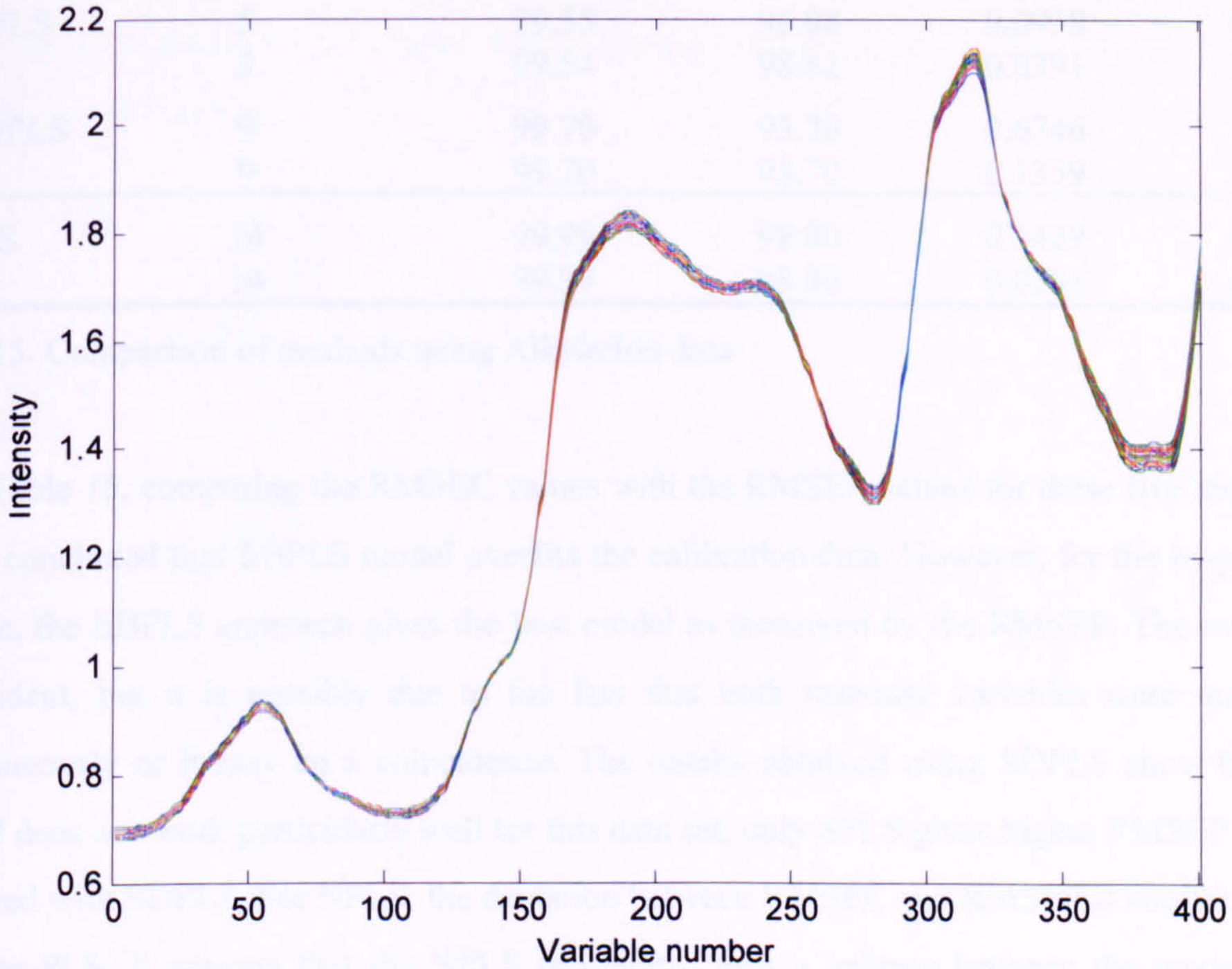

Figure 4.26. Plot of the original alkylation spectra.

The main results are presented in Table 15, where the first row is the result obtained for the product variable and the second row (grey) is the result obtained for the by-product. Since the final model for the product and the by-product is allowed to have different number of latent variables, the value of the captured variance may also differ.

The NPLS model was the only nonlinear algorithm that achieved a lower prediction error for the product than the reference method of linear PLS, Table 15 (F). The result for EBPLS was comparable to that of linear PLS. The other methods failed in the sense that they demonstrated poorer performance than the reference method of PLS (F). Linear PLS is expected to perform acceptably as it is known to handle weak nonlinearities, but at a cost of needing more latent variables compared with the linear case (Martens and Næs, 1989). The data set was noteworthy as the spectra contained regions that exhibited different degrees of nonlinear behaviour with respect to the responses, Stordrange *et al*. (2003).

Method applied	No. of latent variables	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
A: RVPLS	8	99.97	97.14	0.3006	0.8781
	6	99.91	95.89	0.0880	0.0949
B: SPLS	10	99.99	93.38	0.6785	1.5366
	11	99.99	94.57	0.2483	0.2440
C: EBPLS	1	97.26	97.25	0.0063	0.4077
	5	99.44	100.00	0.0002	0.0338
D: NPLS	5	99.55	96.08	0.0938	0.3187
	3	99.54	98.62	0.0391	0.0359
E: SDPLS	9	99.70	93.70	0.6746	1.0545
	9	99.70	93.70	0.1359	0.1676
F: PLS	14	99.99	98.80	0.1429	0.3758
	14	99.99	98.80	0.0383	0.0395

Table 15. Comparison of methods using Alkylation data

From Table 15, comparing the RMSEC values with the RMSEP values for these five models it can be concluded that EBPLS model overfits the calibration data. However, for the by-product variable, the EBPLS approach gives the best model as measured by the RMSEP. The reason is not evident, but it is possibly due to the fact that both response variables were modelled simultaneously or it may be a coincidence. The results obtained using SDPLS show that the method does not work particularly well for this data set, only SPLS gives higher RMSEP values compared with SDPLS. For NPLS, the deviation between RMSEC and RMSEP is similar to that of linear PLS. It appears that the NPLS procedure finds a balance between the tendency to overfit associated with EBPLS and the inability to avoid local minimum synonymous with SDPLS.

RVPLS performs better than SPLS. However, both methods are less efficient than the reference method of linear PLS. An explanation of this behaviour can be deduced based on the first t-u plot, Figure 4.27 (A-B). As a result of poorly chosen weight vectors, the construction of the nonlinear function will introduce an error that is propagated through to the subsequent latent variables. Even if the first weight vector of the RVPLS model appears to be similar to the weight vector of NPLS, the complexity of the data prevents the identification of the appropriate t-u relationship. That is, the model needs information about the covariance of the data matrix to appropriately identify the inner relationship. Since, for each iteration, SDPLS does not use the covariance information, this argument may also be used to explain the poor results for SDPLS. However, it must be seen together with the problem of local minima as this method uses an iterative optimisation technique to minimize the error of the inner mapping. Explicitly, for this

data set, the covariance information must be used to get a good step direction, $\partial \mathbf{w}$, to avoid the problem of local minima.

Examining the weights for the first latent variable (Figure 4.27) for RVPLS (A), EBPLS (C) and NPLS (D), they appears to give a similar shape, with EBPLS being the noisiest. For SPLS (B) the weight vector appears to focus on the same areas, but the structure differ to the other 3 approaches, RVPLS, EBPLS and NPLS. The first weight vector obtained by applying SDPLS (E) is again different to the others, but is closely related to the starting vector, indicating convergence difficulties.

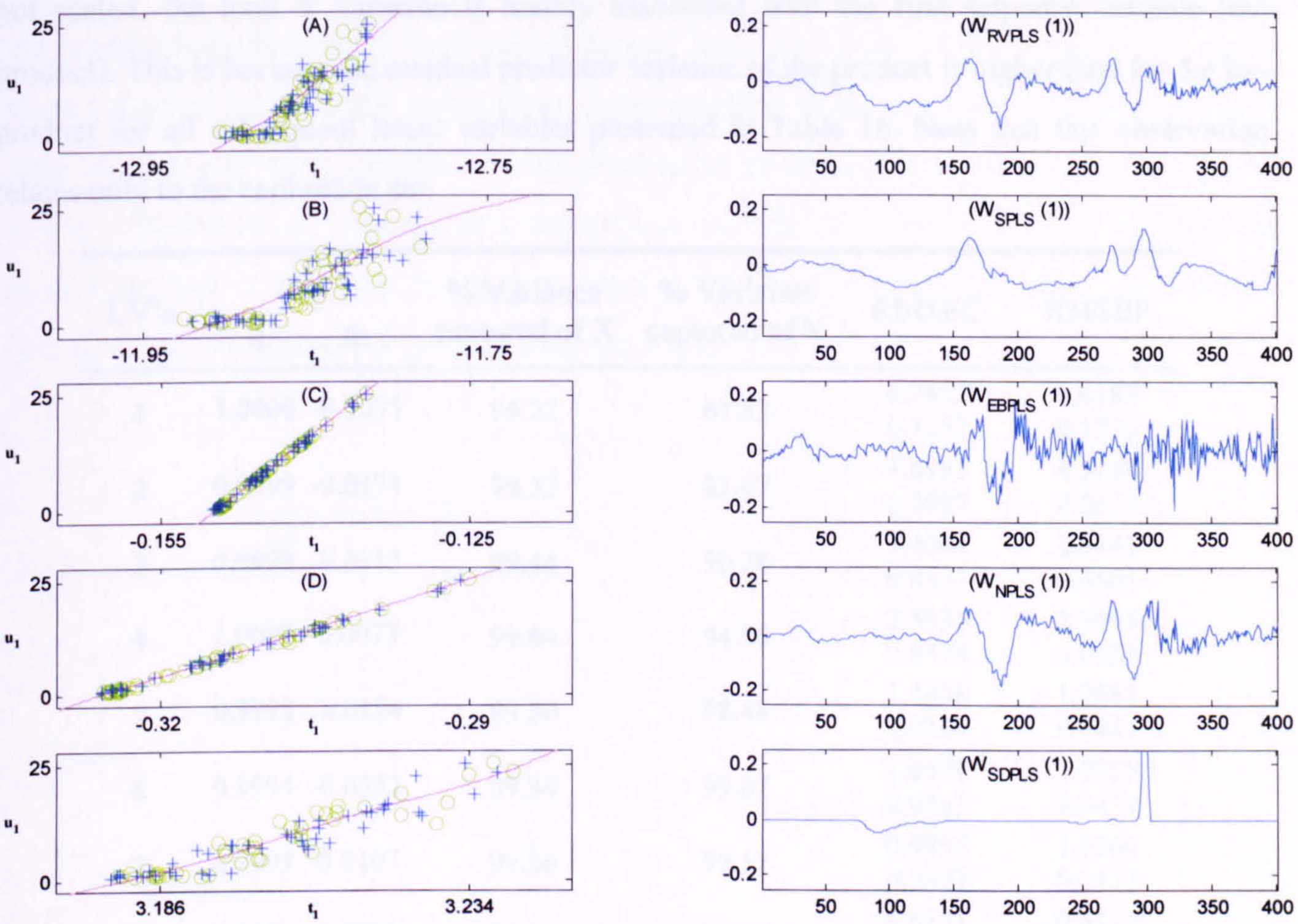

Figure 4.27. Plot of t_1-u_1 and corresponding weight vector for the alkylating data

The nonlinearity in the data is not easily observed from the t_1-u_1 plots, Figure 4.27. This is because the dominant nonlinear behaviour is mainly present in the second response variable, whereas the Y-loading ($\mathbf{q}_1$) implies that the first latent variable primarily explains the first response value, i.e. the product. This issue is discussed further in the next sections.

4.3.1 PLS Reference Method

Linear Partial Least Squares using fourteen latent variables was applied to the data set, as it gave the model with the lowest RMSEP value. The first five latent variables did not capture sufficiently the underlying structure (Figure 4.28) and more latent variables must be included to obtain a satisfactory model (Table 16). Due to the orthogonality between the modelled **X** and **Y** variance in linear PLS, applying a large number of latent variables is of less concern compared with this situation in the nonlinear case. Furthermore, it is known that PLS handles weak nonlinearity by including additional latent variables (Martens and Næs, 1989). Table 16 summarizes the result from applying PLS on the alkylating process data. Since the responses are not scaled, the total **Y** variance is mainly associated with the first response variable (the product). This is because the residual predictor variance of the product is higher than for the by-product for all subsequent latent variables presented in Table 16. Note that this observation relates only to the calibration set.

LV's	Q q_1	Q q_2	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	1.0000	-0.0035	99.22	61.83	6.7452 0.1252	6.6183 0.1225
2	0.9999	-0.0171	99.32	81.47	4.6993 0.0907	4.3147 0.0824
3	0.9998	-0.0213	99.44	90.28	3.4044 0.0637	2.9441 0.0601
4	1.0000	0.0073	99.64	94.36	2.5935 0.0578	2.5994 0.0553
5	0.9998	-0.0174	99.80	98.48	1.3456 0.0364	1.0865 0.0441
6	0.9994	-0.0352	99.84	99.07	1.0539 0.0463	1.0355 0.0439
7	0.9999	0.0107	99.86	99.17	0.9955 0.0457	1.0364 0.0432
8	1.0000	0.0003	99.91	99.68	0.6191 0.0456	0.6115 0.0428
9	1.0000	0.0031	99.96	99.80	0.4877 0.0452	0.4535 0.0436
10	0.9999	-0.0104	99.98	99.91	0.3219 0.0435	0.4436 0.0443
11	1.0000	0.0050	99.99	99.93	0.2819 0.0433	0.3958 0.0445
12	0.9999	-0.0159	99.99	99.96	0.2243 0.0424	0.3934 0.0437
13	0.9999	- 0.0163	99.99	99.98	0.1621 0.0414	0.3777 0.0401
14	0.9820	-0.1891	99.99	99.98	0.1429 **0.0383**	**0.3758** 0.0395
15	1.0000	0.0000	99.99	99.99	0.0985 0.0383	0.3935 0.0396

Table 16. Results of PLS applied to alkylation data

The **Y** loading, **Q**, gives a higher weight to the product than to the by-product for all the latent variables shown. The first latent variable explains almost all the variance in the **X** matrix, whilst only 62% of the **Y** variance is explained. Including subsequent latent variables explains a decreasing amount of variance, until almost all the variance in **Y** is captured. From latent variable nine, the difference between the residual error of the calibration and the validation set increases, thus increasing overfitting occurs. Even though the RMSEP value decreases after latent variable nine, the values flatten out and a threshold value is attained whereby 99.98 % of the variance in the response matrix is explained.

In Figure 4.28, the plots of the weight vectors (—) are supplemented by the regression vector of the product (--), constructed from the referenced latent variables. Thus the regression vector plotted together with weight vector three, is the resulting regression vector, $\mathbf{b}_1^3$ (normalized), obtained after regressing variable one (the product) on the three first score vectors, such that $\mathbf{Y}_1 = \mathbf{X}\mathbf{b}_1^3 + \mathbf{E}$, and so on. Thus, the first weight vector and the first regression vector are identical. In PLS, the task is to explain the highest combined **X** and **Y** variance per latent variable, thus the t-u plot shows different degrees of structure, Figure 4.28.

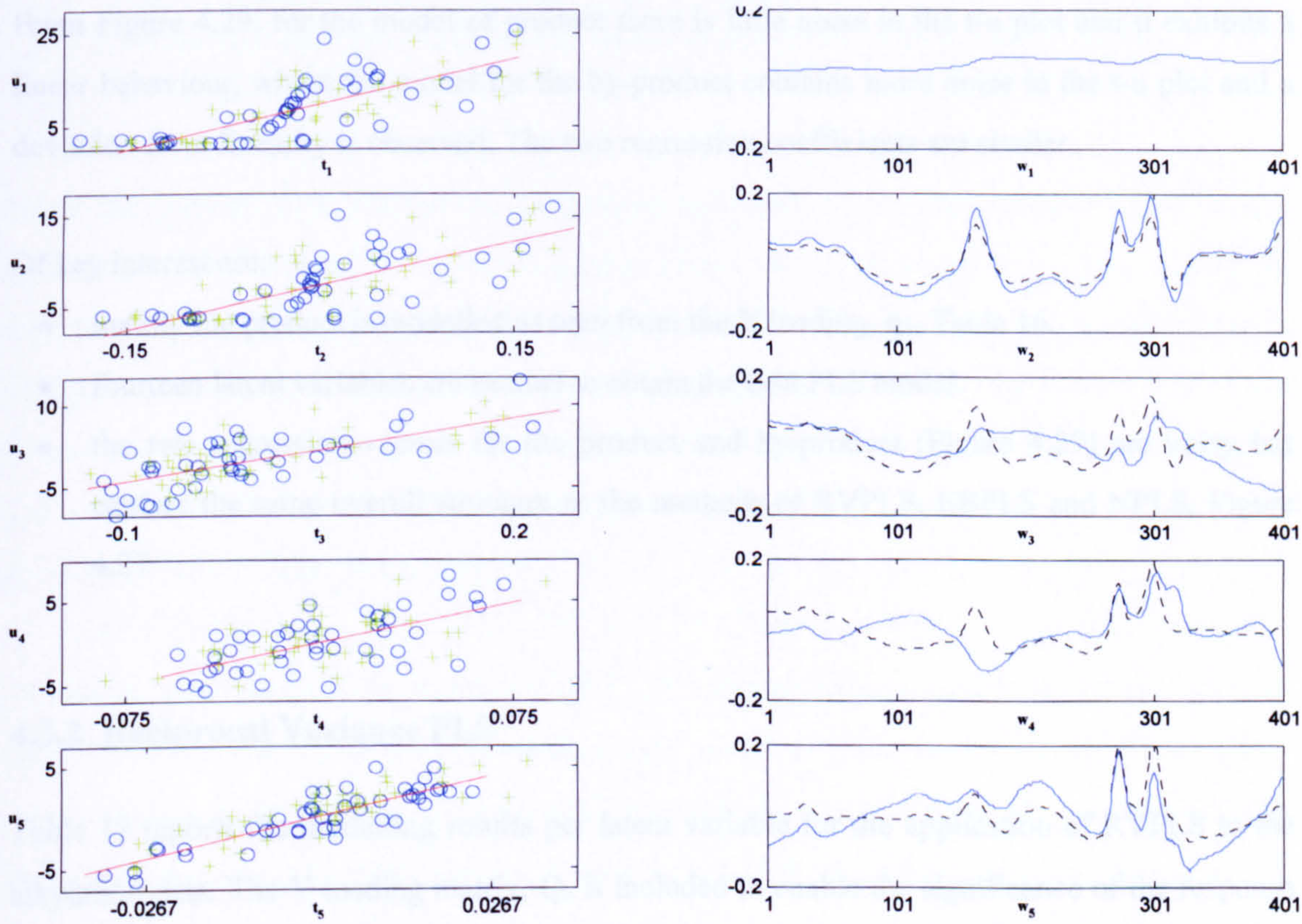

Figure 4.28. The t-u plots, weights and regression vectors for PLS, first five latent variables.

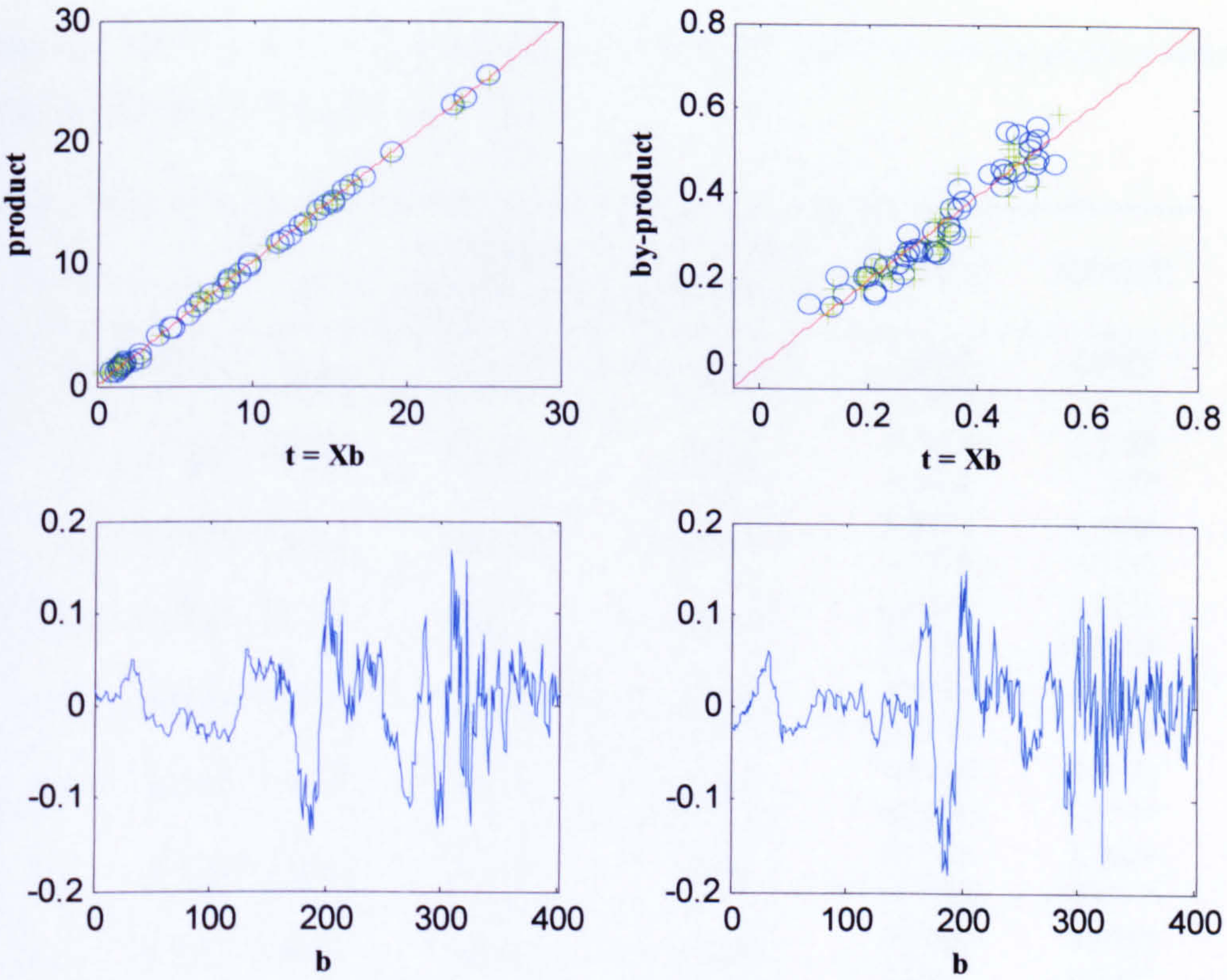

Figure 4.29. The final t-u plot and the regression vector applying fourteen latent variables.

From Figure 4.29, for the model of product there is little noise in the t-u plot and it exhibits a linear behaviour, whilst the model for the by-product contains more noise in the t-u plot and a deviation from linearity is observed. The two regression coefficients are similar.

Of key interest are:

- mainly the product is modelled as seen from the **Y** loading, $\mathbf{q}_1$, Table 16.
- fourteen latent variables are needed to obtain the best PLS model.
- the two regression vectors for the product and by-product (Figure 4.29) are noisy but capture the same overall structure as the methods of RVPLS, EBPLS and NPLS, Figure 4.27

4.3.2 Reciprocal Variance PLS

Table 17 reports the modelling results per latent variable for the application of RVPLS to the alkylation data. The Y-loading matrix, **Q,** is included to enable the significance of the response variables per latent variable to be assessed. Since the variance of the response matrix **Y** is dominated by the first response variable, the product, it was expected that the first latent variable would have a high loading for the product variable. For RVPLS, the impact of the second response value is not observed until the fourth and the sixth latent variable, observed

from the **Q** values, Table 17. Consequently, the RMSEC value of the by-product decreases primarily for the fourth and sixth variables.

LV's	Q q_1	Q q_2	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	0.9998	0.0205	99.51	73.10	2.9242 0.2808	2.8692 0.2734
2	0.9985	-0.0540	99.56	83.70	1.7618 0.2538	2.1133 0.2485
3	0.9996	-0.0284	99.59	91.64	0.8776 0.2498	1.3816 0.2442
4	0.9463	0.3233	99.72	94.13	0.6267 0.1315	1.0143 0.1058
5	0.9997	0.0241	99.75	99.56	0.4665 0.1311	0.8940 0.1054
6	0.8404	0.5419	99.91	95.89	0.4468 **0.0880**	0.9570 **0.0949**
7	0.9997	-0.0230	99.93	96.58	0.3628 0.0874	0.9019 0.0955
8	0.9960	-0.0895	99.97	97.14	**0.3006** 0.0848	**0.8781** 0.0985
9	0.9939	0.1106	99.98	97.77	0.2291 0.0821	0.8875 0.0984
10	0.9998	-0.0178	99.98	98.51	0.1406 0.0820	0.9165 0.0984

Table 17. Application of RVPLS to alkylating data.

Since the method does not use the covariance information, the underlying structure is not accurately identified and the inner mapping introduces a relatively large error, which is incorporated into the subsequent sub-models. In particular, the RMSEC and RMSEP values obtained for the first latent variable are similar, but including more latent variables increases the deviation between the two measures, i.e. 17%, 37%, 38% and 48% for the second, third, fourth and the fifth latent variable. The overfitting increases to 66% for the final model, that was developed from eight latent variables.

Compared with linear PLS, that required 14 latent variables, a poorer prediction error is observed. Since the relationship is approximately linear, the error introduced by allowing the inner mapping to be nonlinear will be small for each latent variable added. Nevertheless, the error introduced per latent variable is cumulative, and will therefore increase as the number of latent variables increases. Thus after 8 latent variables, this error was significant compared with the residual left to be modelled.

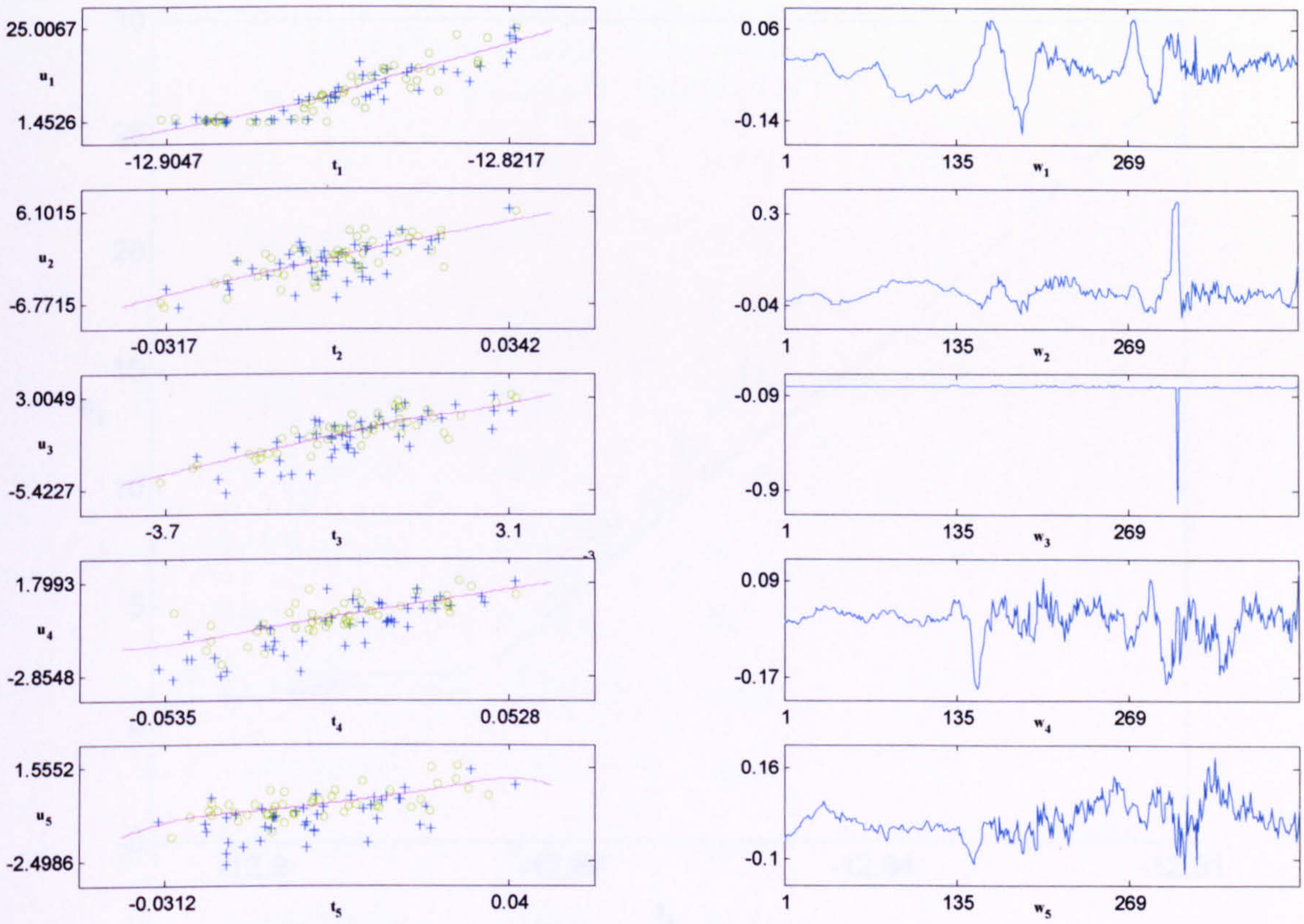

Figure 4.30. The t-u plots and the weight vectors for RVPLS for the first five latent variables.

From Figure 4.30, the issue of fitting the nonlinear function is evident from the first t-u plot. The inner mapping function does not capture the nonlinear behaviour, due to the plug-in bandwidth being estimated too high. The reason is that the loess bandwidth is estimated as a single average value for the whole data set (**t**, **u**) and then varied locally by the distance between the k-nearest neighbours of the **t** vector, Appendix A2.2. Thus, for the lower t-values, where the error variance is less than the average, the bandwidth will be too high and underfitting by the nonlinear function will occur. The error introduced by this mismatch for the lower area of the data cluster (small t-values) is comparable to the variation of the residuals ($\mathbf{u}_2$, Figure 4.30). The RMSEP for the first latent variable could be lowered to a value of 1.8, if the structure of the lower part in the first t-u plot had been modelled better. That is, by manually adjusting the bandwidth for the lower area, Figure 4.31. In particular, the first thirteen smallest t-values were given a bandwidth of 0.0075, compared with the average value of these points of 0.0187 from loess bandwidth.

From Figure 4.30, the weight vectors become gradually more noisy with increasing number of latent variables. For the third latent variable, only a few **X** variables are used, and the region corresponds with the area of interest for the previous modelled latent variable (around variable 310).

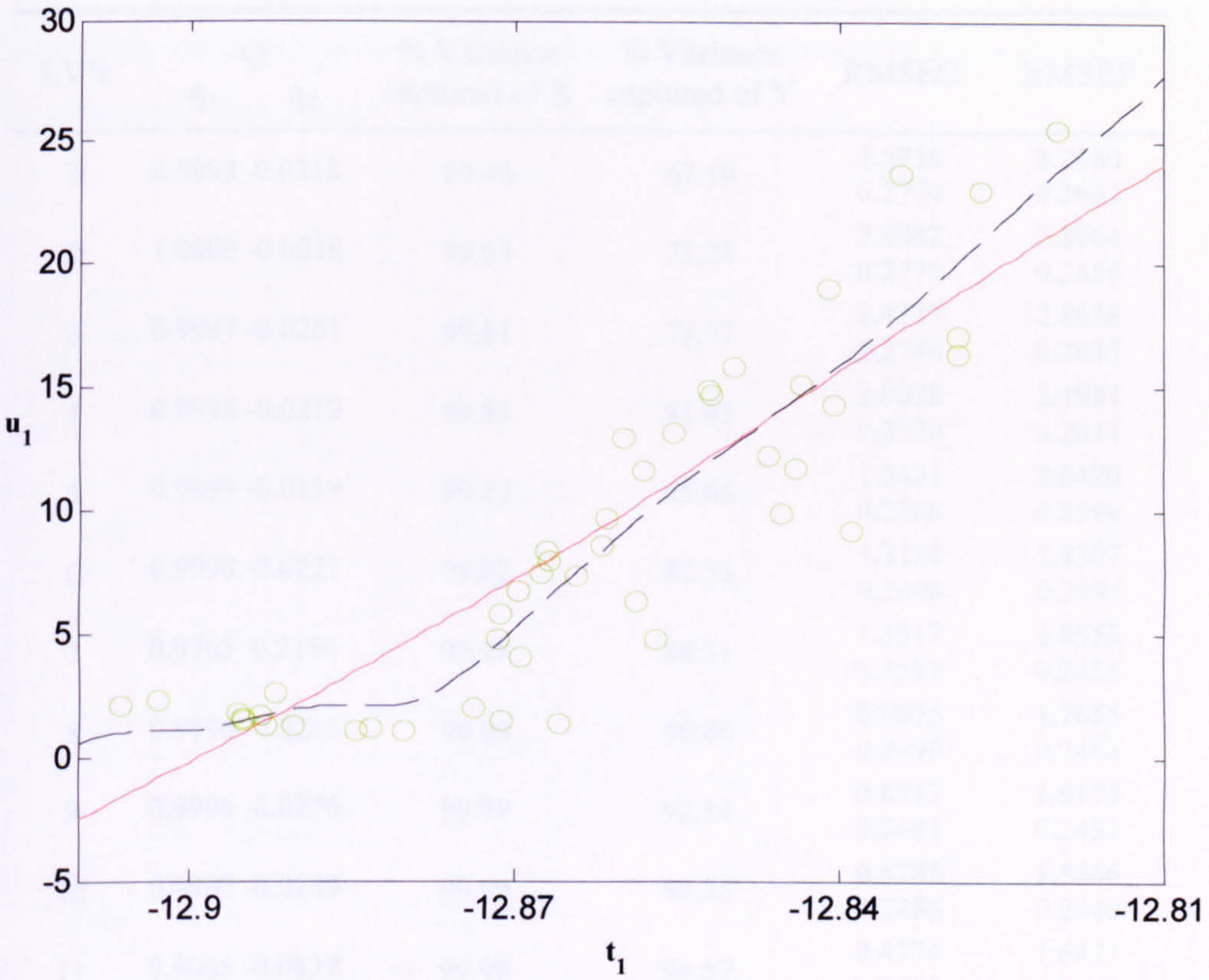

Figure 4.31. The first t-u plot, locally altering the bandwidth of the lowest t-values.

In a study into the effect of the nonlinear function fitted, the inner mapping was forced to be linear by applying a large bandwidth for the local linear kernel regression, i.e. the bandwidth was given a value of 10^8. The best model for the product was achieved when 11 latent variables were included in the model, resulting in a RMSEC of 0.33 and a RMSEP of 0.36. The best model for the by-product was achieved when applying 14 variables, resulting in a RMSEC of 0.039 and a RMSEP of 0.037. Thus, the performance of the methods was comparable to that of linear PLS. Furthermore, the overfitting, measured by the ratio between the RMSEC and RMSEP values, was eliminated. Consequently, the cause of the overfitting mainly resulted from the construction of the inner mapping.

4.3.3 Spline PLS

Spline PLS has the same set of issues as RVPLS. In particular, the method locates inappropriate nonlinear latent spaces as a consequence of focusing on explaining a high level of variance for both the predictor and the response matrices. The inaccuracy of the inner mapping affects the modelling of the subsequent latent variables, Figure 4.32. In particular, the problem is more complicated than with RVPLS since additional latent variables are required to construct the best model.

LV's	Q q_1	Q q_2	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	0.9998	0.0218	99.46	67.19	3.5716 0.2774	3.2060 0.2663
2	1.0000	-0.0018	99.63	73.24	2.9082 0.2772	2.8964 0.2658
3	0.9997	-0.0261	99.81	77.77	2.4117 0.2744	2.8638 0.2635
4	0.9998	-0.0219	99.85	81.45	2.0078 0.2720	2.4981 0.2611
5	0.9999	-0.0159	99.87	85.66	1.5421 0.2706	2.0420 0.2599
6	0.9998	-0.0221	99.92	87.73	1.3124 0.2696	1.8307 0.2595
7	0.9765	0.2156	99.98	88.31	1.2517 0.2507	1.8353 0.2455
8	0.9996	-0.0265	99.99	90.60	0.9955 0.2499	1.7655 0.2464
9	0.9996	-0.0276	99.99	92.14	0.8213 0.2491	1.6175 0.2453
10	0.9997	-0.0249	99.99	93.38	**0.6785** 0.2486	**1.5366** 0.2440
11	0.9965	-0.0839	99.99	94.57	0.4374 **0.2483**	1.6421 **0.2440**

Table 18. Application of SPLS to alkylating data

From Figure 4.32, the structure of the first latent variable and the weight vector is similar to that for RVPLS, but the RMSEC and RMSEP are significantly higher. Again, the variance in the t-u plot for the first latent variable is not uniformly distributed along the t-axis, and the average smoothing parameter constructed using the plug-in bandwidth estimates too high a value for the lower part of the t-u plot (lowest **t** values). Thus, the mismatch between the structure observed in the t-u plot and the estimated inner mapping is even greater than for RVPLS. As a result this error that is included in the subsequent latent variables will be higher and further reduce the capability of the method.

The largest value of the Y-loading for the by-product is obtained for the seventh latent variable, i.e. 0.2156. Consequently the potential to model the behaviour of the by-product is low. For this data set, SPLS tends to describe more of the variance in the predictor variables compared with the variance in the response variables as seen from the variance described, Table 18. This method works in a similar manner to that of linear PLS in that it tries to explain the variance in both the **X** and **Y** matrices, and as a result the underlying structure is not necessarily identified as observed from the t-u plots in Figure 4.32. However, the focus of the method results in low noise for the weight vectors compared to the other methods. The structure of the t-u plots reveals different levels of nonlinearity depending on which region of the spectrum is primarily defined by the weight vector. Only the first five latent variables are plotted, since including

more plots does not contribute further to the discussion. In general, the higher the latent variable the less structure in the t-u plot.

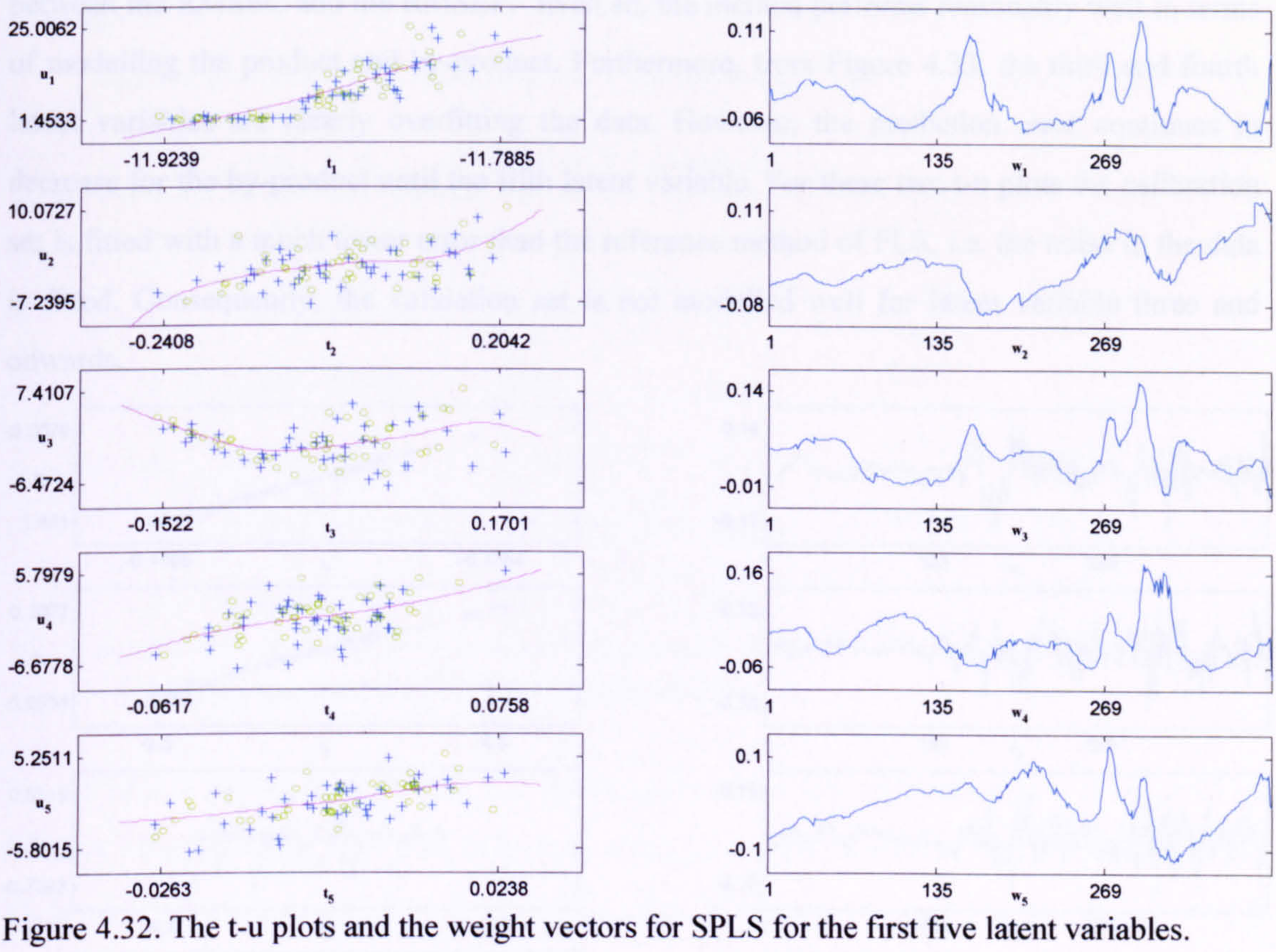

Figure 4.32. The t-u plots and the weight vectors for SPLS for the first five latent variables.

4.3.4 Error Based PLS

The EBPLS method captures the underlying structure, with the behaviour associated with the concentration of the product captured by the first latent variable and the concentration of the by-product captured by the second latent variable, Table 19 (Q values).

LV's	Q q_1	Q q_2	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	0.9998	0.0177	97.26	97.25	**0.0063** 0.3000	**0.4077** 0.2869
2	0.0208	–0.9998	99.36	99.99	0.0010 0.0006	0.4089 0.0341
3	0.9000	-0.4358	99.39	99.99	0.0005 0.0005	0.4089 0.0340
4	0.5059	0.8626	99.42	100.00	0.0004 0.0003	0.4089 0.0338
5	0.8841	-0.4674	99.44	100.00	0.0002 **0.0002**	0.4089 **0.0338**
6	0.9164	-0.4003	99.48	100.00	0.0002 0.0002	0.4089 0.0338

Table 19. Application of EBPLS to alkylating data

The second inner mapping exhibited nonlinear behaviour compared with the first nonlinear function, which was close to linear. The method appears to overfit as there is large differences between the RMSEC and the RMSEP. Even so, the method performs reasonably well in terms of modelling the product and by-product. Furthermore, from Figure 4.33, the third and fourth latent variables are clearly overfitting the data. However, the prediction error continues to decrease for the by-product until the fifth latent variable. For these two t-u plots the calibration set is fitted with a much lower error than the reference method of PLS, i.e. the noise in the data is fitted. Consequently, the validation set is not modelled well for latent variable three and onwards.

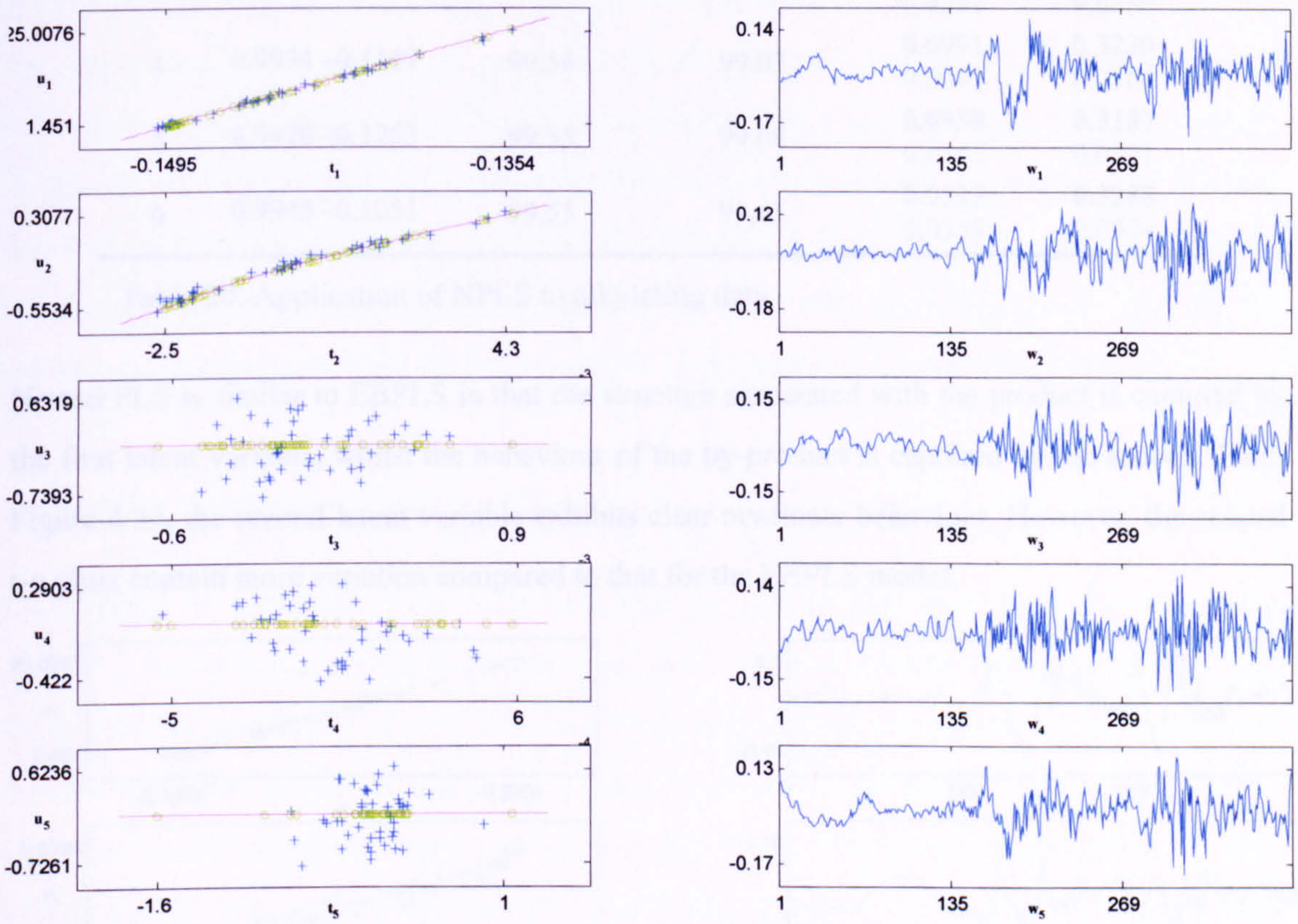

Figure 4.33. The t-u plots and the weight vectors for EBPLS for the first five latent variables.

Modelling two response variables that are correlated generally decreases the tendency for overfitting, as the latent variable **u** is a linear combination of the responses. This linear combination can be considered a weighted average (scaled), and thus will have lower noise than the responses. Decreasing the noise improves the possibility of overfitting. However, if the two responses are modelled separately, the RMSEP values decrease to 0.38 for the product and 0.026 for the by-product applying one latent variable in the model, possibly due to the nonlinear relation between the two responses, Figure 4.25. The weight vectors are relatively noisy, but certain structural information is observed – particularly for the first two latent variables. In particular, the variables between 150 to 200 possess a similar structure to what is observed for the RVPLS method, Figure 4.30.

4.3.5 Nested PLS

The main results after applying NPLS are summarized in Table 20, the best results are identified in bold.

LV's	Q q_1	Q q_2	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	0.9998	0.0176	96.84	96.03	0.3124 0.3008	0.4453 0.2862
2	0.2023	0.9793	99.53	97.17	0.3061 0.0422	0.4404 0.0397
3	0.9981	–0.0609	99.54	98.62	0.1459 **0.0391**	0.3384 **0.0359**
4	0.9934	–0.1147	99.54	99.03	0.0991 0.0372	0.3230 0.0365
5	0.9920	–0.1263	99.55	99.08	**0.0938** 0.0367	**0.3187** 0.0371
6	0.9945	–0.1051	99.55	99.41	0.0533 0.0358	0.3288 0.0374

Table 20. Application of NPLS to alkylating data

Nested PLS is similar to EBPLS in that the structure associated with the product is captured by the first latent variable, whilst the behaviour of the by-product is captured by the second. From Figure 4.34, the second latent variable exhibits clear nonlinear behaviour. However, the second t-u plots contain more variation compared to that for the EBPLS model.

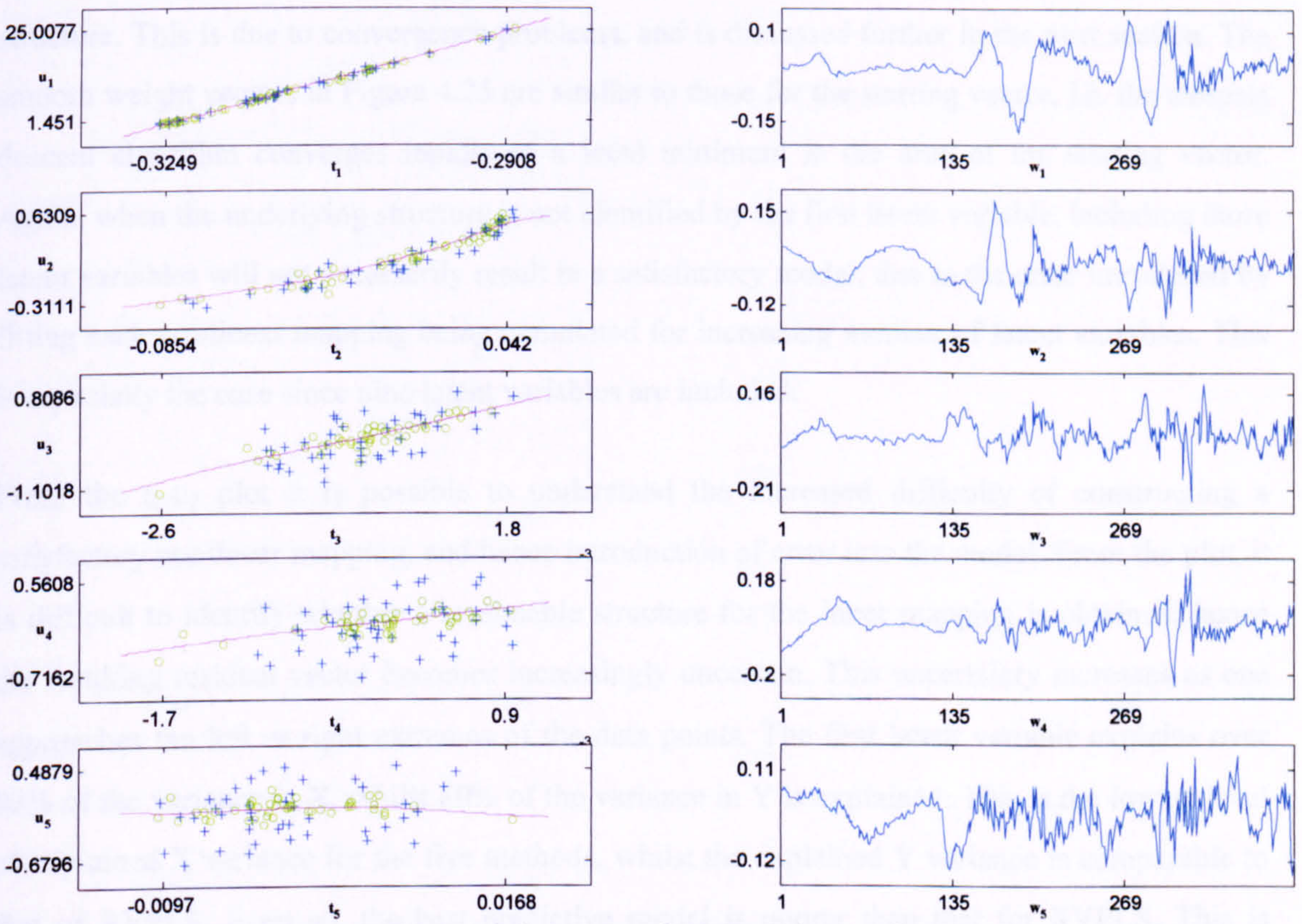

Figure 4.34. The t-u plots and the weight vectors for NPLS for the first five latent variables.

Furthermore, the influence of the product on the second latent variable is greater than for EBPLS. The effect of overfitting is less obvious with the deviation between the prediction error and the calibration error being smaller. In particular, the third latent variable captures the residual structure common to both the calibration and the validation set. From Figure 4.34, the weight vectors are less noisy and contain more structure compared with those obtained using EBPLS. For the fifth latent variable there is little structure observed in the t_5-u_5 plot, with the weight vector being noisier than the previous latent variables.

The curvature in the t_2-u_2 plot, Figure 4.34, is the opposite of that seen in the t_2-u_2 plot, Figure 4.33. This is due the sign of the Y-loading, i.e. -0.9998 in Table 19 and 0.9793 in Table 20. As the two responses are negatively correlated (-0.90), the choice of sign for EBPLS is more natural since the Y-loading values for latent variable two have opposite signs. It may be caused by significant residual variance, originating from the product variable, remaining after the first latent variable model being built. Consequently, it will dominate the second latent variable. If the two responses are modelled separately, the RMSEP value is unchanged at 0.31 for the product and decreases to 0.028 for the by-product applying two latent variables in the model

4.3.6 Steepest Descent PLS

From Table 21 and Figure 4.35, it is clear that the method fails to identify the underlying structure. This is due to convergence problems, and is discussed further in the next section. The smooth weight vectors in Figure 4.25 are similar to those for the starting vector, i.e. the steepest descent algorithm converges rapidly to a local minimum in the area of the starting vector. Again, when the underlying structure is not identified by the first latent variable, including more latent variables will not necessarily result in a satisfactory model, due to the error introduced by fitting each nonlinear mapping being cumulated for increasing number of latent variables. This is especially the case since nine latent variables are included.

From the t_2-u_2 plot it is possible to understand the increased difficulty of constructing a satisfactory nonlinear mapping, and hence introduction of error into the model. From the plot, it is difficult to identify whether a reasonable structure for the inner mapping is obtained, hence the resulting residual vector becomes increasingly uncertain. This uncertainty increases as one approaches the left or right extremes of the data points. The first latent variable explains over 90% of the variance in **X**, whilst 80% of the variance in **Y** is explained. This is the lowest level of explained **X** variance for the five methods, whilst the explained **Y** variance is comparable to that of RVPLS. Even so, the best predictive model is poorer than that for RVPLS. This is possibly due to the issue of convergence resulting in a higher level of nonlinearity for the inner

mapping between the pair of scores – due to greater uncertainty in the high and low extremes of the data (Figure 4.35, second t-u plot and onwards).

LV's	Q q_1	Q q_2	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	0.9998	0.0190	99.32	79.18	2.2537 0.2929	2.3115 0.2809
2	0.9998	0.0193	99.44	81.26	2.0257 0.2918	2.2086 0.2770
3	0.9989	-0.0463	99.45	89.02	1.1647 0.2836	1.3805 0.2701
4	0.5521	-0.8337	99.60	89.31	1.1543 0.1746	1.3762 0.1840
5	0.9415	0.3371	99.67	89.70	1.1168 0.1378	1.4098 0.1690
6	1.0000	-0.0077	99.68	91.51	0.9282 0.1377	1.2350 0.1690
7	0.9999	0.0104	99.68	92.06	0.8562 0.1377	1.1928 0.1687
8	0.9997	-0.0248	99.69	92.93	0.7600 0.1371	1.1850 0.1682
9	0.9986	-0.0521	99.70	93.70	**0.6746** **0.1359**	**1.0545** **0.1676**
10	0.9996	0.0273	99.70	94.05	0.6356 0.1357	1.1226 0.1683

Table 21. Application of SDPLS to alkylating data

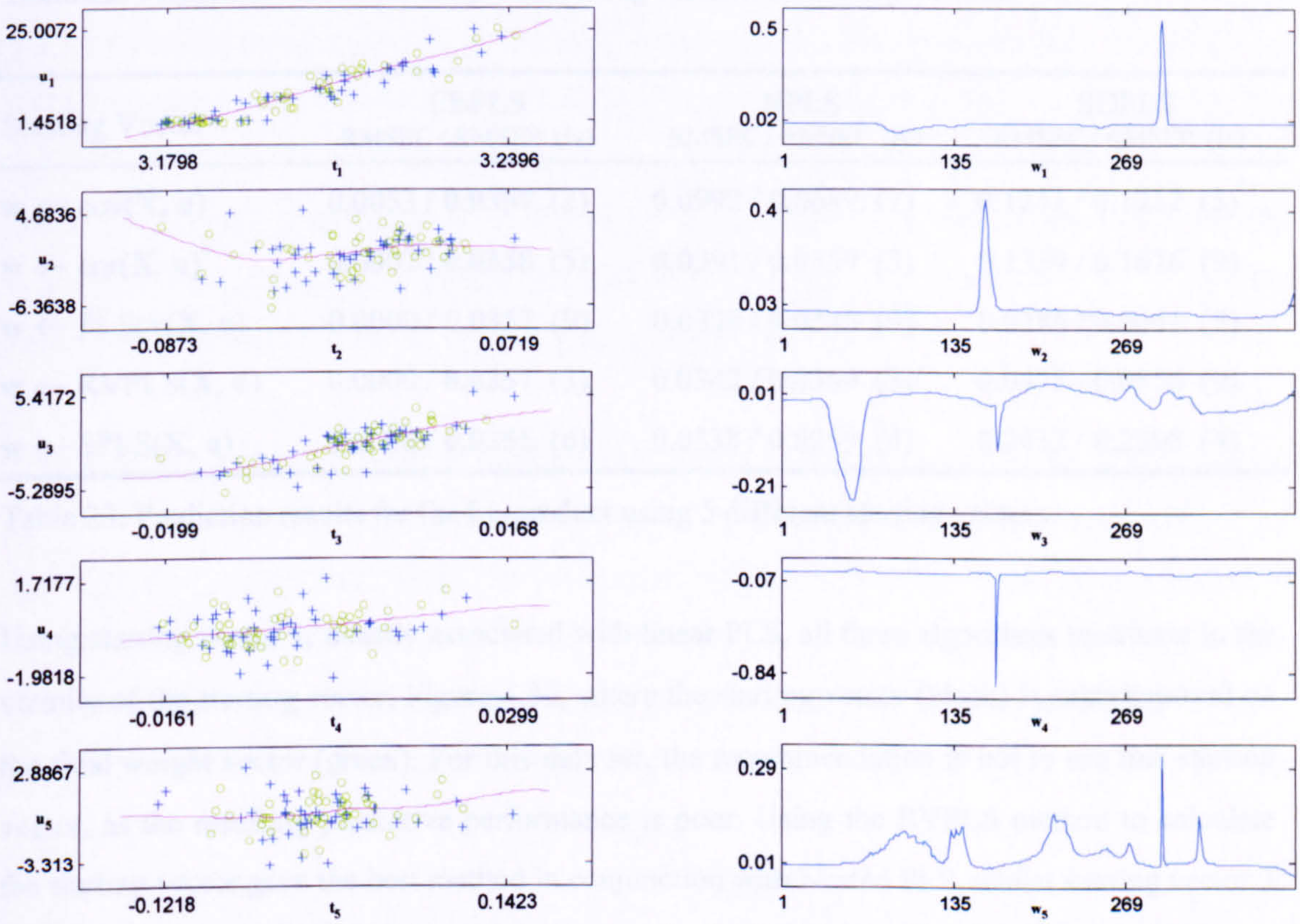

Figure 4.35. The t-u plots and the weight vectors for SDPLS for the first five latent variables.

The weight vectors are spiky due to the starting vector used, cor(**X**, **u**)^9, combined with premature convergence close to the starting vector, since the covariance information is not used to construct the weight updating vector. These issues will be discussed in the next section.

4.3.7 Influence of the Starting Vector

The influence of the starting vector was investigated for Framework 1, i.e. EBPLS, NPLS and SDPLS. In particular, the deviation between the weight vectors for the first latent variable and the prediction ability of the resulting models were investigated for five starting vectors. The results from the different starting vectors including the prediction results are presented in Table 22 and Table 23, for the product and the by-product, respectively.

Starting Vector	EBPLS RMSEC / RMSEP (lv)	NPLS RMSEC / RMSEP (lv)	SDPLS RMSEC / RMSEP (lv)
w ← cov(**X**, **u**)	0.3000 / 0.4039 (1)	0.1558 / **0.3758** (7)	0.4663 / 0.7036 (7)
w ← cor(**X**, **u**)^9	0.0063 / 0.4077 (1)	0.0938 / **0.3187** (5)	0.6746 / 1.0545 (9)
w ← PLScv(**X**, **u**)	0.0000 / 0.4028 (1)	0.0177 / **0.3185** (9)	0.0582 / 0.3237 (5)
w ← RVPLS(**X**, **u**)	0.0000 / 0.4033 (1)	0.2102 / **0.2969** (3)	0.4548 / 1.0121 (2)
w ← SPLS(**X**, **u**)	0.0000 / 0.4040 (1)	0.2101 / **0.3071** (4)	0.2441 / 0.9169 (5)

Table 22. Prediction results for the product using 5 different starting vectors.

Starting Vector	EBPLS RMSEC / RMSEP (lv)	NPLS RMSEC / RMSEP (lv)	SDPLS RMSEC / RMSEP (lv)
w ← cov(**X**, **u**)	0.0053 / **0.0357** (2)	0.0992 / 0.0689 (7)	0.1241 / 0.1212 (5)
w ← cor(**X**, **u**)^9	0.0002 / **0.0338** (5)	0.0391 / 0.0359 (3)	0.1359 / 0.1676 (9)
w ← PLScv(**X**, **u**)	0.0000 / 0.0357 (9)	0.0326 / **0.0345** (9)	0.0286 / 0.0641 (5)
w ← RVPLS(**X**, **u**)	0.0000 / **0.0357** (3)	0.0342 / 0.0369 (3)	0.0475 / 0.0856 (9)
w ← SPLS(**X**, **u**)	0.0000 / **0.0355** (6)	0.0338 / 0.0359 (4)	0.2432 / 0.2298 (4)

Table 23. Prediction results for the by-product using 5 different starting vectors.

Using starting vector 1, usually associated with linear PLS, all three algorithms terminate in the vicinity of the starting vector, Figure 4.36, where the starting vector (black) is superimposed on the final weight vector (green). For this data set, the recommendation is not to use this starting vector, as the resulting predictive performance is poor. Using the RVPLS method to calculate the starting vector gave the best method in conjunction with Nested PLS, whilst starting vector 3 (PLScv) gave the best result for SDPLS.

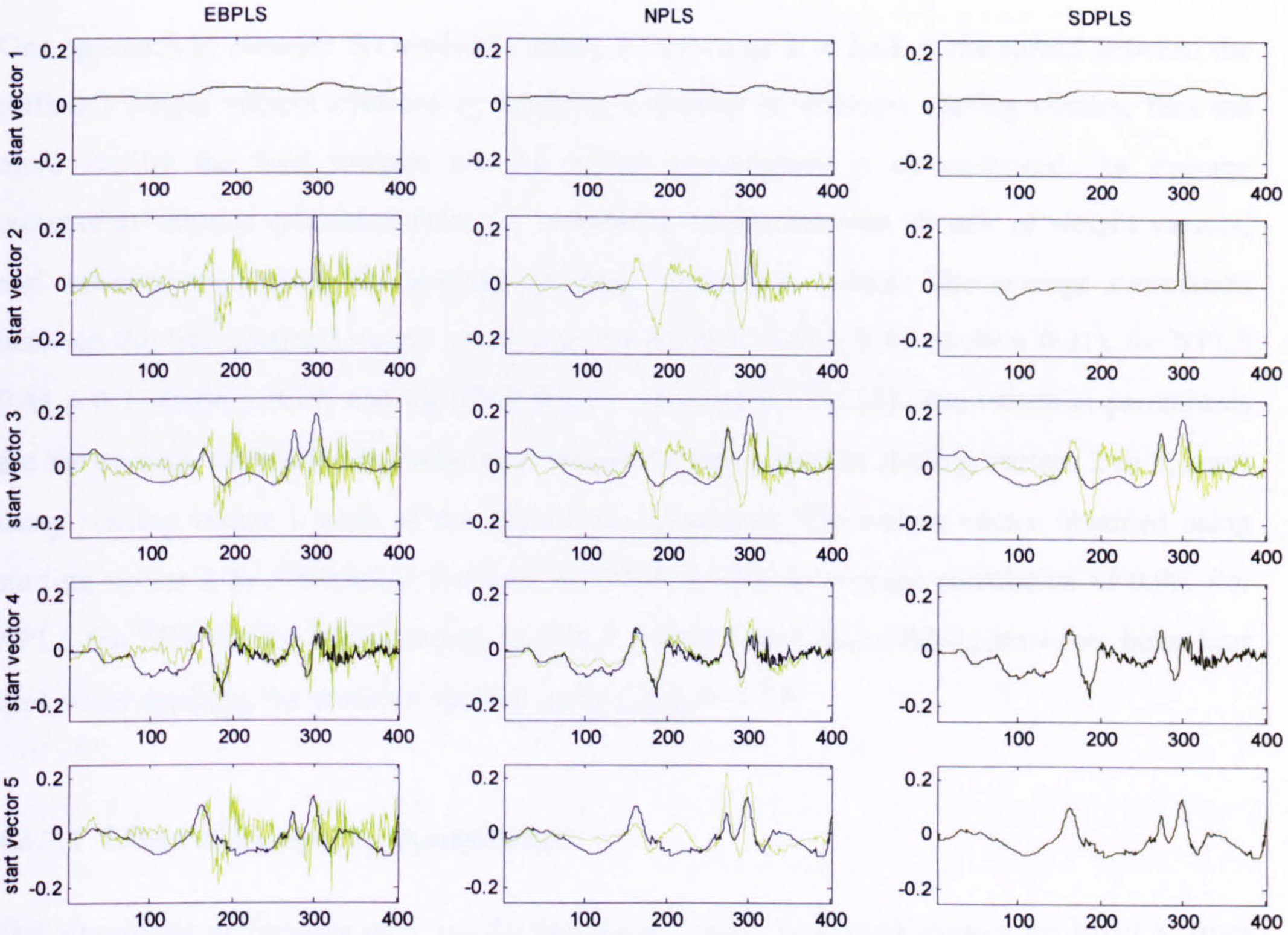

Figure 4.36. The starting vector (black) and the final weights (green) for the first latent variable.

Nested PLS gave the best result for the product response, whilst the by-product is modelled best using the EBPLS approach. This is due to EBPLS modelling the two responses individually with the product being modelled by the first latent variable while the by-product was modelled from the second latent variable. For NPLS, the two responses are modelled in a similar way, but the influence from the residual variance of the product is larger for the second latent variable. The SDPLS result is generally poor, except for the product model that is obtained when the regression vector from PLS with cross validations is used as the starting vector. The RMSEP is then lower than that of linear PLS, i.e. 0.32 compared with 0.38.

Again, SDPLS appeared to be the most affected by the choice of starting vector. This was also reflected in the diversity in the prediction ability of the five SDPLS models. However, by using the regression vector from linear PLS with cross validation (starting vector 3), an acceptable model was obtained. This was the only case where the weight vector significantly was different from that of the starting vector, Figure 4.36. For this model, the first weight vector exhibits similar behaviour to that of the first weight vector obtained using Nested PLS, using starting vectors two to five.

One approach to measure the method's ability to converge is to look at the spread between the different weight vectors obtained by applying a number of different starting vectors, thus the more similar the final weights are the higher convergence is demonstrated. An average correlation value is calculated from the correlation values between all pair of weight vectors, and subsequently taking the average of these correlation values. The average correlation between the five obtained weight vectors are for EBPLS: 0.63 ± 0.47 (0.99 ± 0.01), for NPLS 0.44 ± 0.32 (0.66 ± 0.19) and for SDPLS 0.26 ± 0.22 (0.32 ± 0.23). The values in parenthesis are the average correlation between final weight vectors using the starting vectors 2 to 5, since using starting vector 1 none of the algorithms converged. The weight vector obtained using starting vector 2 to 5 is almost identical for EBPLS with an average correlation of 0.99. For NPLS the fifth starting vector results in a first weight vector that exhibits smoother behaviour than when applying the previous starting vectors, Figure 4.36.

4.3.7.1 Effect of Dampened Optimisation

The algorithms of Framework 1 use the regression vector from least squares for EBPLS, from PLS with the number of latent variables found from cross validation for NPLS, or from PLS with one latent variable for SDPLS, to define the increment vector or search direction, $\partial \mathbf{w}$. Since in PLS the step length and the step direction are found by regressing **u** on **t** (termed **r** on **s** for the inner PLS) using least squares, all methods find a step length that minimizes the squared residual error. Thus, applying dampening does not necessarily improve the prediction ability of the method, but will influence the convergence as cross validation is included in the search algorithm since it is used to find the step length. However, dampening may improve termination by applying a sufficiently large step length that forces the solution out of a local minimum, or by improving the termination of the algorithm. That is, when the chosen step length is found to be zero, the algorithm will terminate since no change in the score vector between iterations is observed.

Starting Vector	EBPLS RMSEC / RMSEP (lv)	NPLS RMSEC / RMSEP (lv)	SDPLS RMSEC / RMSEP (lv)
$\mathbf{w} \leftarrow \text{cov}(\mathbf{X}, \mathbf{u})$	0.0040 / **0.4022** (1)	0.0898 / 0.4543 (8)	0.0714 / 0.5750 (10)
$\mathbf{w} \leftarrow \text{cor}(\mathbf{X}, \mathbf{u})$[9]	0.0054 / **0.4052** (1)	0.1005 / 0.4626 (8)	1.1284 / 1.0631 (2)
$\mathbf{w} \leftarrow \text{PLScv}(\mathbf{X}, \mathbf{u})$	0.0169 / 0.3750 (3)	0.0453 / 0.3195 (4)	0.1136 / **0.3071** (3)
$\mathbf{w} \leftarrow \text{RVPLS}(\mathbf{X}, \mathbf{u})$	0.0198 / 0.3668 (3)	0.0655 / **0.3142** (6)	0.0449 / 0.3334 (8)
$\mathbf{w} \leftarrow \text{SPLS}(\mathbf{X}, \mathbf{u})$	0.0053 / 0.4045 (1)	0.2045 / **0.2891** (3)	0.4297 / 1.2326 (2)

Table 24. Prediction results for the product using 5 different starting vectors (dampening).

Starting Vector	EBPLS RMSEC / RMSEP (lv)	NPLS RMSEC / RMSEP (lv)	SDPLS RMSEC / RMSEP (lv)
$\mathbf{w} \leftarrow \text{cov}(\mathbf{X}, \mathbf{u})$	0.0000 / **0.0368** (2)	0.0426 / 0.0625 (10)	0.1624 / 0.1877 (8)
$\mathbf{w} \leftarrow \text{cor}(\mathbf{X}, \mathbf{u})$[9]	0.0003 / **0.0336** (3)	0.0196 / 0.0533 (10)	0.2042 / 0.2379 (4)
$\mathbf{w} \leftarrow \text{PLScv}(\mathbf{X}, \mathbf{u})$	0.0092 / **0.0333** (9)	0.0397 / 0.0354 (2)	0.0455 / 0.0569 (8)
$\mathbf{w} \leftarrow \text{RVPLS}(\mathbf{X}, \mathbf{u})$	0.0010 / **0.0371** (5)	0.0473 / 0.0453 (6)	0.0330 / 0.0644 (10)
$\mathbf{w} \leftarrow \text{SPLS}(\mathbf{X}, \mathbf{u})$	0.0000 / **0.0334** (6)	0.0350 / 0.0406 (3)	0.0537 / 0.0889 (10)

Table 25. Prediction results for the by-product using 5 different starting vectors (dampening).

The effect of dampening is investigated for the five different starting vectors discussed in the previous section by comparing Table 24 with Table 22. The greatest effect of dampening is observed for Steepest Descent PLS (SDPLS), with the algorithm terminating further away from the starting vector than when dampening is not applied, Figure 4.37. However, applying dampening produces both better and poorer models for SDPLS. For Nested PLS, by applying starting vector one and two, considerably poorer models are obtained, whilst slightly better models are obtained using starting vector five, Table 24. In particular, dampening causes Nested PLS not to converge properly when applying starting vector 2, in contrast to when not applying dampening, Figure 4.37. For EBPLS, the effect of dampening is small. Thus, it is not possible to conclude whether damped optimisation generally improves the algorithms.

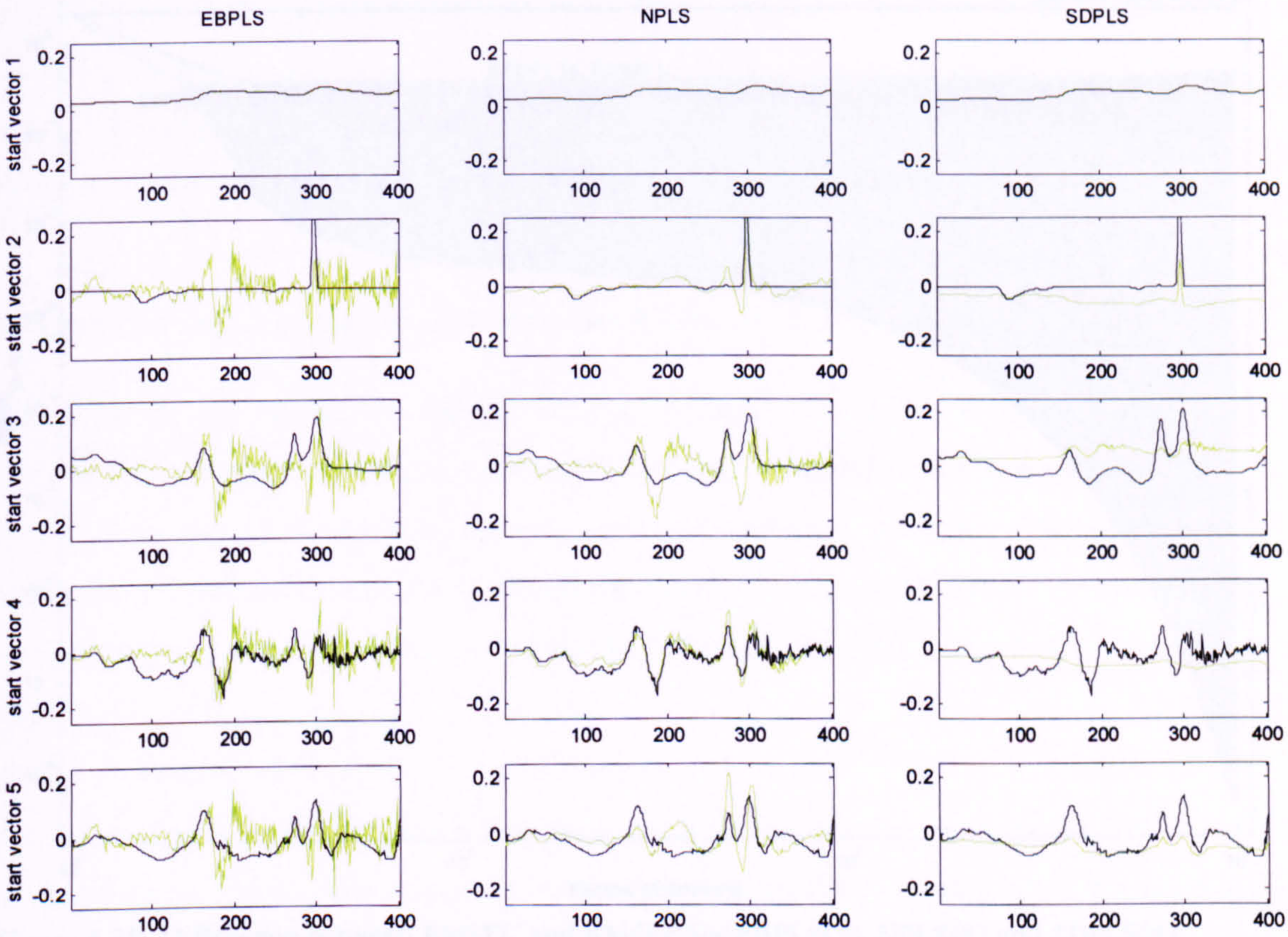

Figure 4.37. The starting vector (black) and the final weights (green) for the first latent variable.

The average correlation between the obtained weight vectors for the different starting vectors are for EBPLS: 0.56 ± 0.42 (0.88 ± 0.10), for NPLS 0.42 ± 0.31 (0.63 ± 0.21) and finally for SDPLS 0.13 ± 0.78 (0.02 ± 0.76). Again, the values in parenthesis are the average correlation between the four last staring vectors. As the average correlation between the weight vectors by applying dampening are generally less compared with not applying dampening, it could be used as an argument not to apply dampening for this data set. SDPLS is the method that is most affected by local minima, due to the low average correlation among the weight vector obtained for the first latent variable applying the different starting vectors.

4.3.8 Impact of the Termination Criteria

The difference between the prediction error and the calibration error is investigated in terms of the number of iterations performed. The difference between the RMSEC and RMSEP values are plotted as an area for the three optimisation based approaches, EBPLS, NPLS and SDPLS, using the standard parameter settings, and hence starting vector 3, Figure 4.38. In particular, the lowest border-line of each area represents the RMSEC value, whilst the highest border-line of each area represent the RMSEP value.

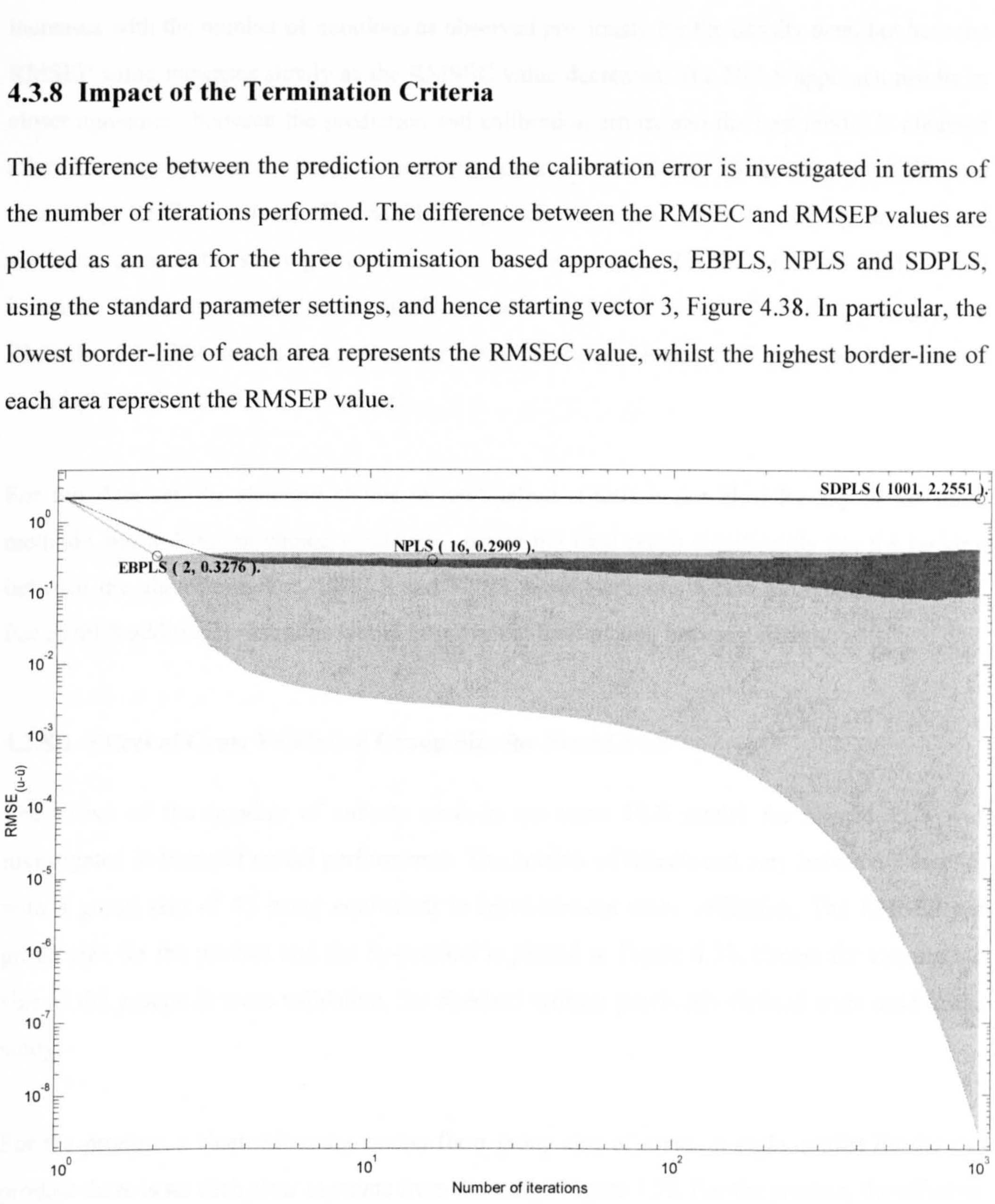

Figure 4.38. Difference between RMSEC and RMSEP for EBPLS(▪), NPLS(▪) and SDPLS(▪).

The area represents the difference between RMSEC and RMSEP per number of iterations applied, thus a large area indicates overfitting. The point at which the lowest RMSEP value for the three methods is attained are added to the plot, including the number of iterations where the minimum RMSEP value occurs. Since there are two response variables the errors used to calculate the RMSEC and RMSEP values are calculated based on the difference between the actual and predicted Y-score, i.e. $\mathbf{e} = (\mathbf{u} - \hat{\mathbf{u}})$, and will be closely related to RMSEC and RMSEP values of the product variable due to the greater variance of the product.

Investigating the EBPLS approach, the difference between the RMSEC and RMSEP values increases with the number of iterations as observed previously for the density data, but here the RMSEP value increases slowly as the RMSEC value decreases. The NPLS approach results in closer agreement between the prediction and calibration errors, and the best model is obtained after 16 iterations since further iterations result in overfitting (increasing RMSEP and decreasing RMSEC values). The SDPLS approach converges extremely slowly into a local minimum close to the starting vector, thus the area between the RMSEC and the RMSEP value is small. However, overfitting eventually occurs, i.e. a test using 10000 iterations gives a RMSEP of 2.2780, this is larger than for 1001 iterations. Nested PLS gave the best overall model (RMSEP = 0.29 at 16 iterations).

For this data set, the standard choice of termination criteria is not ideal for any of the three methods, but a different choice would not change the final result significantly nor the ranking between the algorithms. For, EBPLS and NPLS fewer iterations would have been beneficial. For SDPLS additional iterations would improve the final model, but only slightly.

4.3.8.1 Effect of Cross Validation Group Size for Nested PLS

The effect of the number of subsets used in the inner PLS model for Nested PLS was investigated in terms of model performance. The number of subsets can vary between 2 and 45, with a group size of 45 being equivalent to leave-one-out cross validation. The RMSEP per group size for the product and the by-product is plotted in Figure 4.39. Except for varying the size of the groups in cross validation, the standard settings previously defined were used in the study.

For the product, a clear minimum occurs for a group size of seven or eight, whilst for the by-product there is no such clear outcome from the study, Figure 4.39. For the product, the effect of choosing the optimal number of subgroups in the cross validation is significant, as a

significantly improved model is achieved using eight subsets as opposed to the two groups used in the standard parameter settings. The reason for this not occurring for the by-product is not fully understand, nor is the reason for the peak observed using six cross validation groups. However, the result may be influenced by the difference in variation between the two response variables.

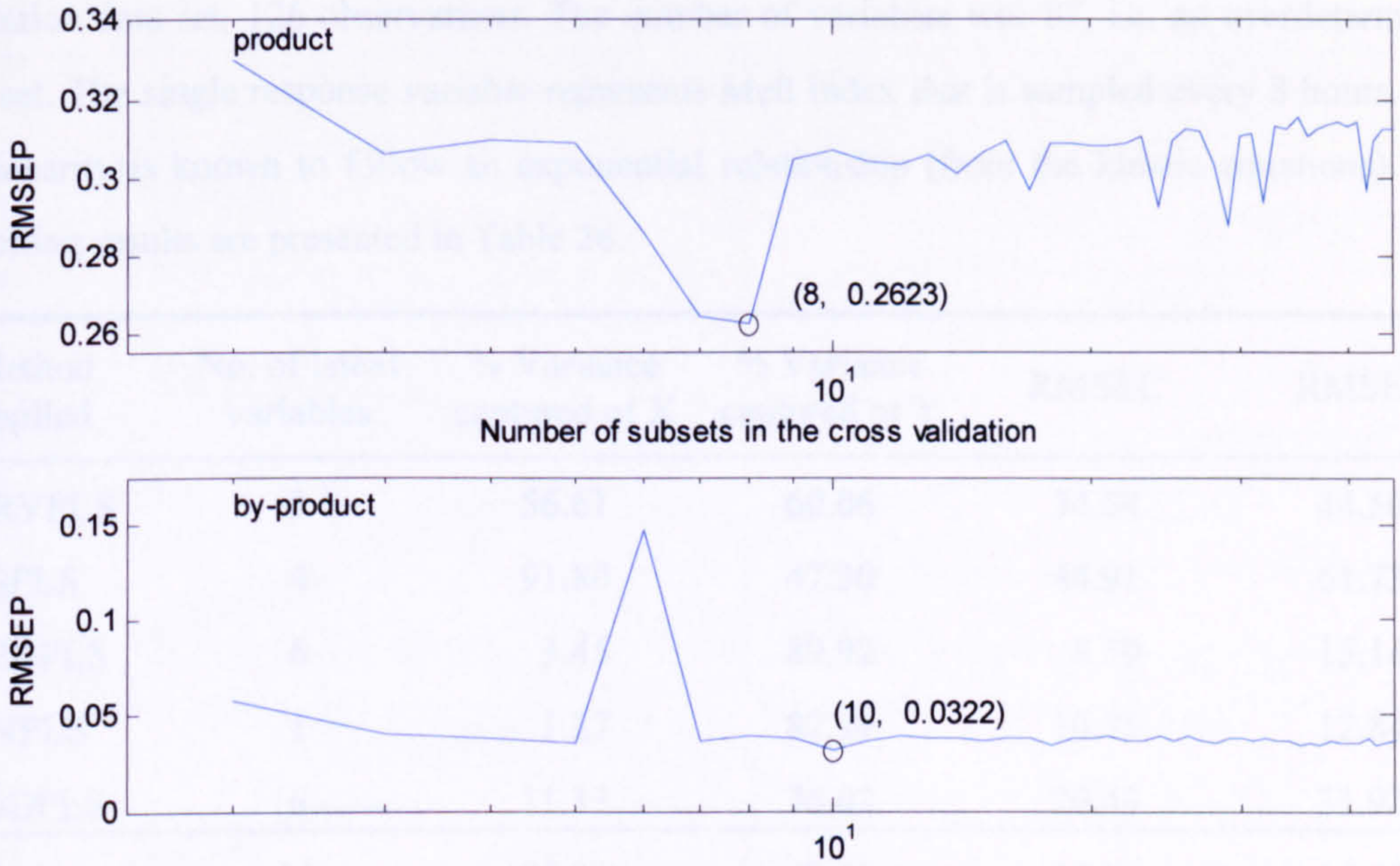

Figure 4.39. RMSEP for NPLS for different group sizes in cross validation.

Generally, when using a higher number of subgroups in cross validation, the calibration sets will be larger while the validation set will be smaller. Thus, the quality of the calibration will be better but the estimation of the validation error may increase, e.g. applying the highest number of subgroups (leave-one-out cross validation) often gives optimistic Root Mean Squared Error of Cross Validation (RMSECV), Martens and Næs (1989). The result for this data set indicates that it can be beneficial to try different group sizes for cross validation when applying Nested PLS, and that the optimal number lies between 4 and 11 as reported by Wold (1978).

4.4 Melt Index of Polymer.

The third data set comprehensively investigated was based on data recorded from a high-density polyethylene (HDPE) plant between 1999 and 2000. The predictor variables include pressures, temperatures and concentrations. Thus the predictor variables are not as strongly correlated as for the previous two data sets. The calibration data set contains 300 observations and the validation data set, 126 observations. The number of variables was 87, i.e. an overdetermined data set. The single response variable represents Melt Index that is sampled every 8 hours. The nonlinearity is known to follow an exponential relationship (from the kinetic equations). The modelling results are presented in Table 26.

Method applied	No. of latent variables	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
A: RVPLS	3	56.61	60.06	34.04	44.50
B: SPLS	4	91.80	47.30	44.91	61.73
C: EBPLS	6	3.45	89.92	8.59	15.14
D: NPLS	1	1.27	87.39	10.75	12.84
E: SDPLS	6	11.37	76.03	20.43	31.92
F: PLS	35	99.99	82.56	14.86	15.60
G: BPLS	35	1.05	84.78	12.97	14.11

Table 26. Comparison between the methods for Melt Index data

Process data often exhibits complex relationships between the predictor variables and the response variables of interest and this data set was no exception. As a consequence, it is more challenging to build a high-quality model as the start vector is assumed to be relatively far away from the resulting weight vector. In linear PLS, the complexity may be expressed in terms of a large number of latent variables in the final model. Finally, the resulting weights are more challenging to validate, since they normally lack any structure compared with spectral data.

The RVPLS and SPLS methods (Framework 2) are unable to identify a good inner mapping for this data set. This result in the error from the fitting the nonlinear function in the first t-u plot, becoming large. In particular for SPLS (Figure 4.40, B), the variation in the predictor matrix (92%) is much higher than for the response variance (47%), thus the underlying structure was not properly modelled. As a consequence of the starting vector being far from the desired minimum and the fact that data is affected by process noise, SDPLS fails to approach the global minimum resulting in a prediction error that is far larger than for linear PLS. Only NPLS and EBPLS perform better than the reference method of linear PLS when considering the RMSEP,

and the underlying nonlinear structure is more clearly defined, Figure 4.40. The large number of latent variables included in liner PLS confirms the complexity of the model. The BPLS model, obtained by adding a nonlinear mapping between the predicted response from the linear PLS model and the measured Melt Index, increases the prediction performance and is only outperformed by Nested PLS. Finally, applying linear PLS, after linearization of the response variable by a logarithmic transformation, resulted in a RMSEP of 13.5 using 18 latent variables.

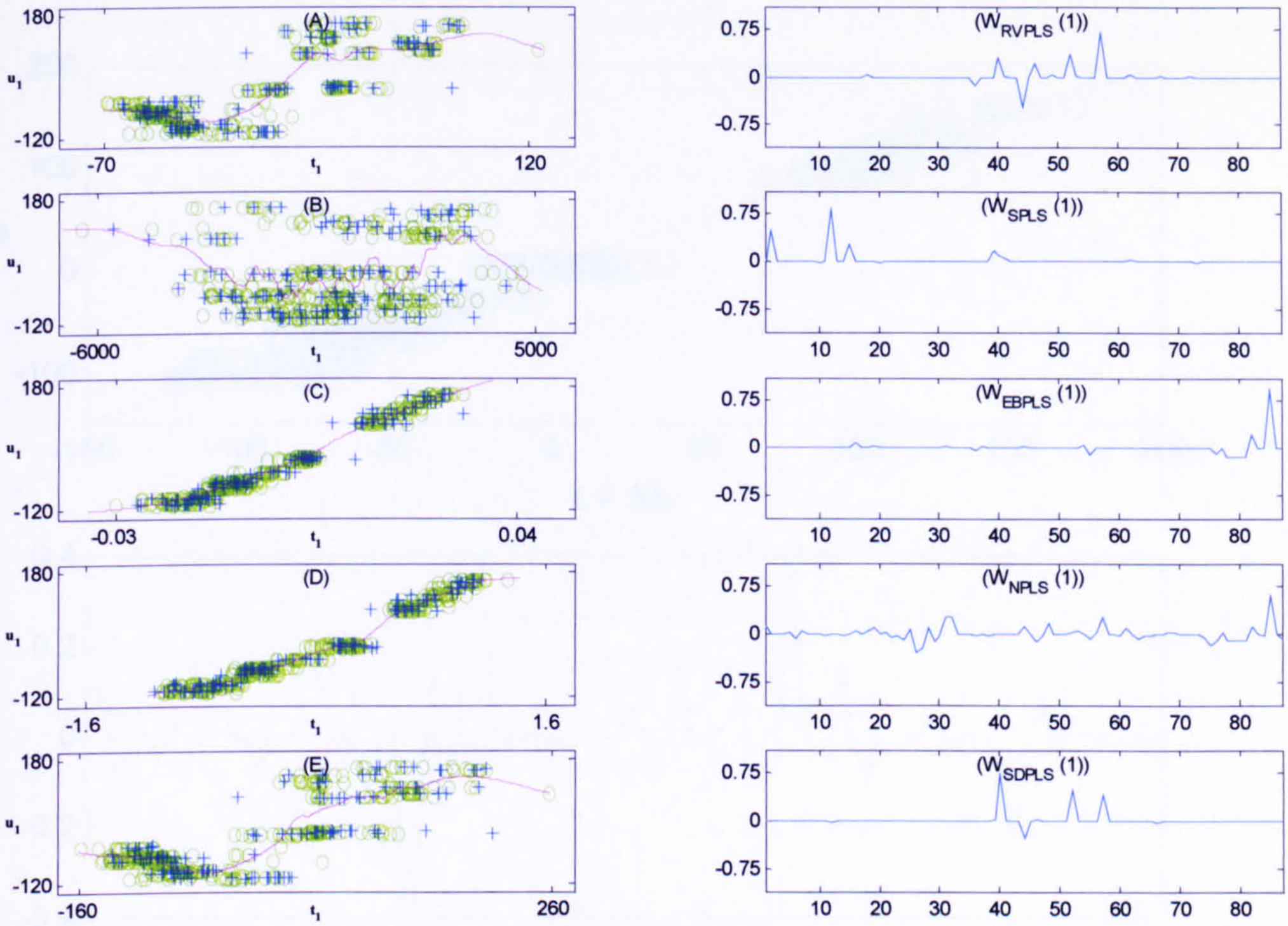

Figure 4.40. Plot of t_1-u_1 scores and corresponding weight vector for the Melt Index data

The weight vectors for the first latent variable are quite different for the five models, with each focusing on different predictor variables. However, the weight vectors for NPLS and EBPLS exhibit similar behaviour for variables in excess of 80, with RVPLS and SDPLS demonstrating structural similarity. EBPLS appears to materialize in a smoother nonlinear mapping than NPLS, but three observations result in a deviation from the model thus making the prediction error larger, Figure 4.40 (C). The major variance in the predictor variables is not associated with the response variable, observed by the low explained variance of **X** for NPLS and EBPLS, Table 26. To confirming this hypothesis, the variance explained by the regression vector from linear PLS, applied in the BPLS approach, is also low. This fact explains the difficulty of constructing a good model for the SPLS algorithm, as it tries to model both the **X** and **Y** variance. Nested PLS seems to give higher significance to additional variables in construction of the weight vector, compared with RVPLS, SPLS, EBPLS and SDPLS, Figure 4.40. A more detailed analysis of the different models follows in the next sections.

4.4.1 PLS Reference Method

A linear Partial Least Squares model using 35 latent variables was applied to the process data. The high number of latent variables indicates the complexity of the model. The resulting regression vector and the observed versus the predicted Melt Index are presented in Figure 4.41. In the upper t-u plot, the linear and the nonlinear mappings are for PLS (black) and BPLS (red), respectively. The corresponding regression vector is included in the lower plot.

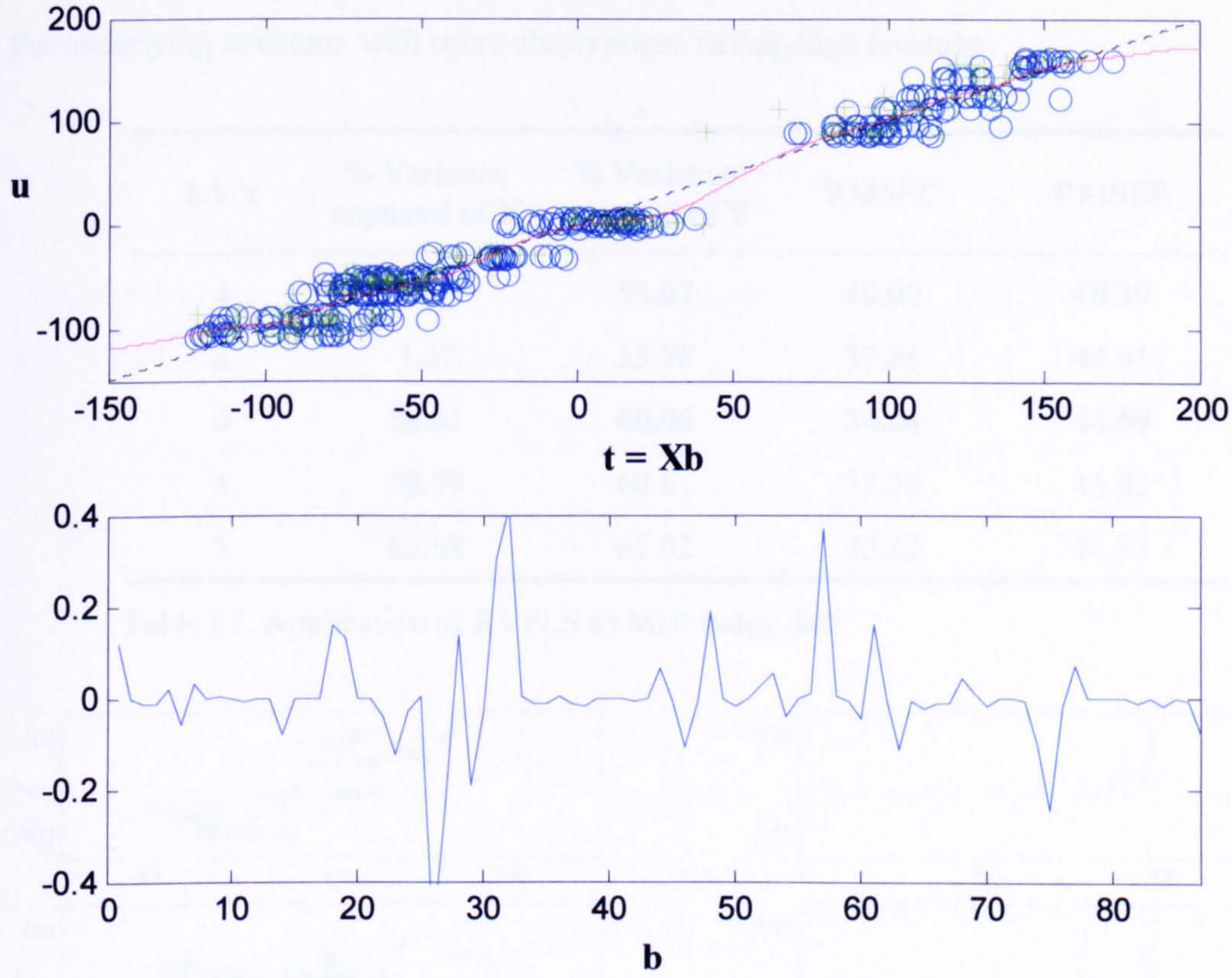

Figure 4.41. Plot of predicted and observed Melt Index and the regression vector.

The shape of the regression vector is similar to that constructed by means of NPLS, for the region from variable number 20 to 70, but is highly different for variable numbers over 80 where it resembles the weights obtained by EBPLS (Figure 4.41 compared to Figure 4.40). Thus, NPLS demonstrates resemblance to PLS, through the use of the inner PLS, but demonstrates also resemblance to EBPLS due to the utilisation of the weight updating procedure (Baffi *et al.,* 1999a,b). From the t-u plot in Figure 4.41, the response is clustered into groups and besides the uncertainty is larger for the prediction than for the measured value. Thus, it can either be due to PLS not being able to describe the complexity of the data set, or the variation not being included in the predictor matrix.

4.4.2 Reciprocal Variance PLS

The RVPLS approach is not identifying the complex data structure required to produce a good model. The methodology was developed for where there exists one underlying relationship between the data and the response as is typical for some spectral data sets. In particular, the explained variance of the response is low. The first latent variable is capable of extracting some of the underlying structure, but for the second latent variable, the nonlinear mapping cannot be modelled accurately, Figure 4.42. The fourth and fifth latent variables tend not to reflect any of the underlying structure, with some observations having high leverage.

LV's	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	3.34	53.07	40.00	48.39
2	3.37	55.58	37.86	44.61
3	**56.61**	**60.06**	**34.04**	**44.50**
4	60.59	60.61	33.56	45.02
5	60.68	61.02	33.22	44.55

Table 27. Application of RVPLS to Melt Index data

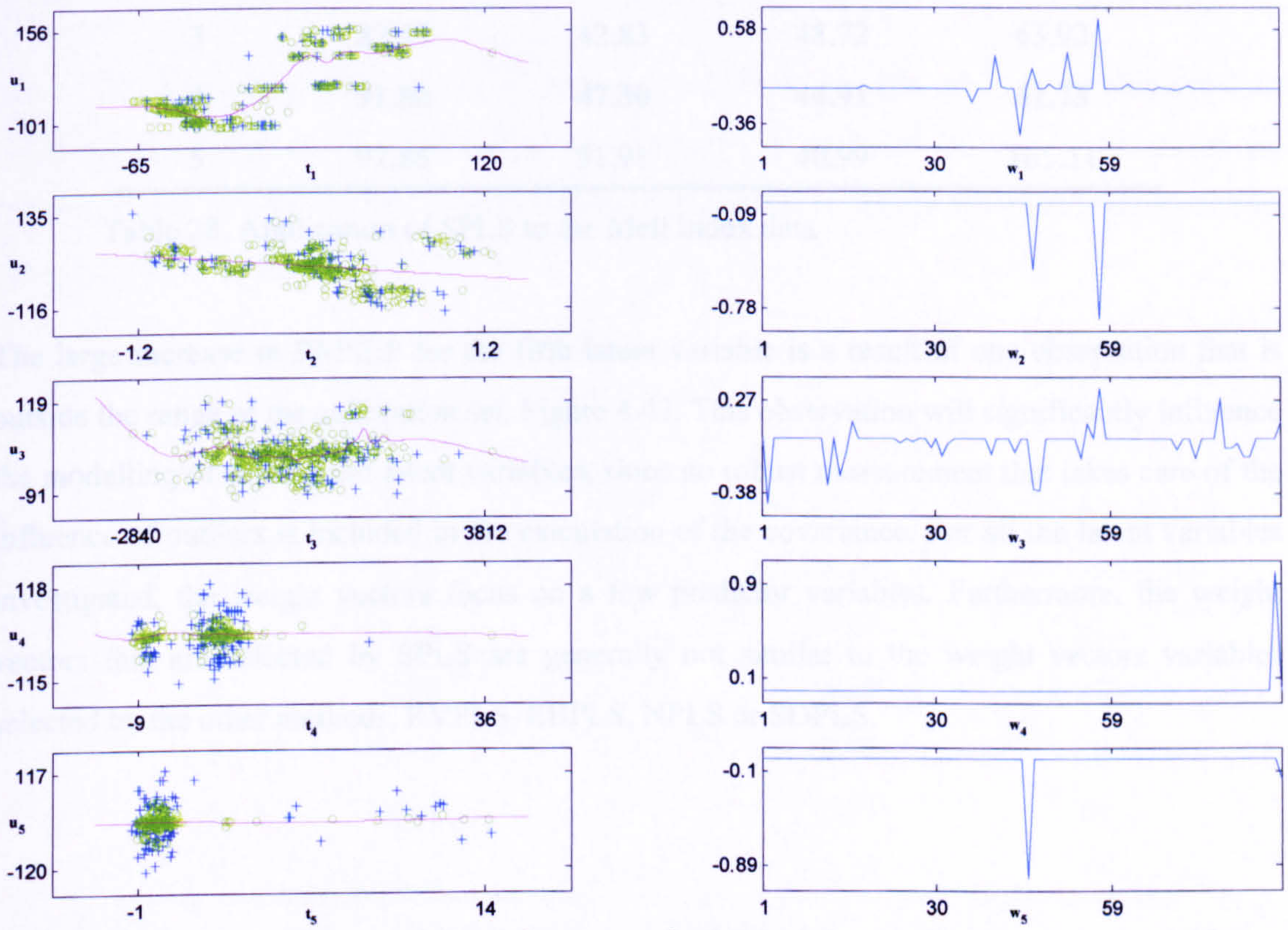

Figure 4.42. The t-u plots and the weight vectors for RVPLS for the first five latent variables.

The RMSEP values decrease until latent variable 3 is included. Apart from latent variable 3, the weight vectors generally focus on a few specific variables. However, the five first weight vectors include some of the same variables as those obtained for the first weight vector when applying Nested PLS.

4.4.3 Spline PLS

The issues with SPLS are apparent from the t-u plots, Figure 4.43. Since the underlying structure cannot be properly identified, it is not possible to fit a nonlinear function between the scores. Furthermore, the structure of the underlying mapping does not demonstrate a smooth behaviour. Consequently, an error is introduced by the nonlinear mapping that cannot be recovered through the inclusion of more latent variables. The final model therefore gives a poor representation of the response variable.

LV's	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	56.21	14.46	72.90	83.48
2	82.20	24.33	64.49	80.60
3	87.57	42.83	48.72	63.92
4	**91.80**	**47.30**	**44.91**	**61.73**
5	92.86	51.91	40.99	165.54

Table 28. Application of SPLS to the Melt Index data

The large increase in RMSEP for the fifth latent variable is a result of one observation that is outside the range of the calibration set, Figure 4.43. This observation will significantly influence the modelling of subsequent latent variables, since no robust measurement that takes care of the influence of outliers is included in the calculation of the covariance. For all the latent variables investigated, the weight vectors focus on a few predictor variables. Furthermore, the weight vectors that are selected by SPLS are generally not similar to the weight vectors variables selected by the other methods, RVPLS, EBPLS, NPLS or SDPLS.

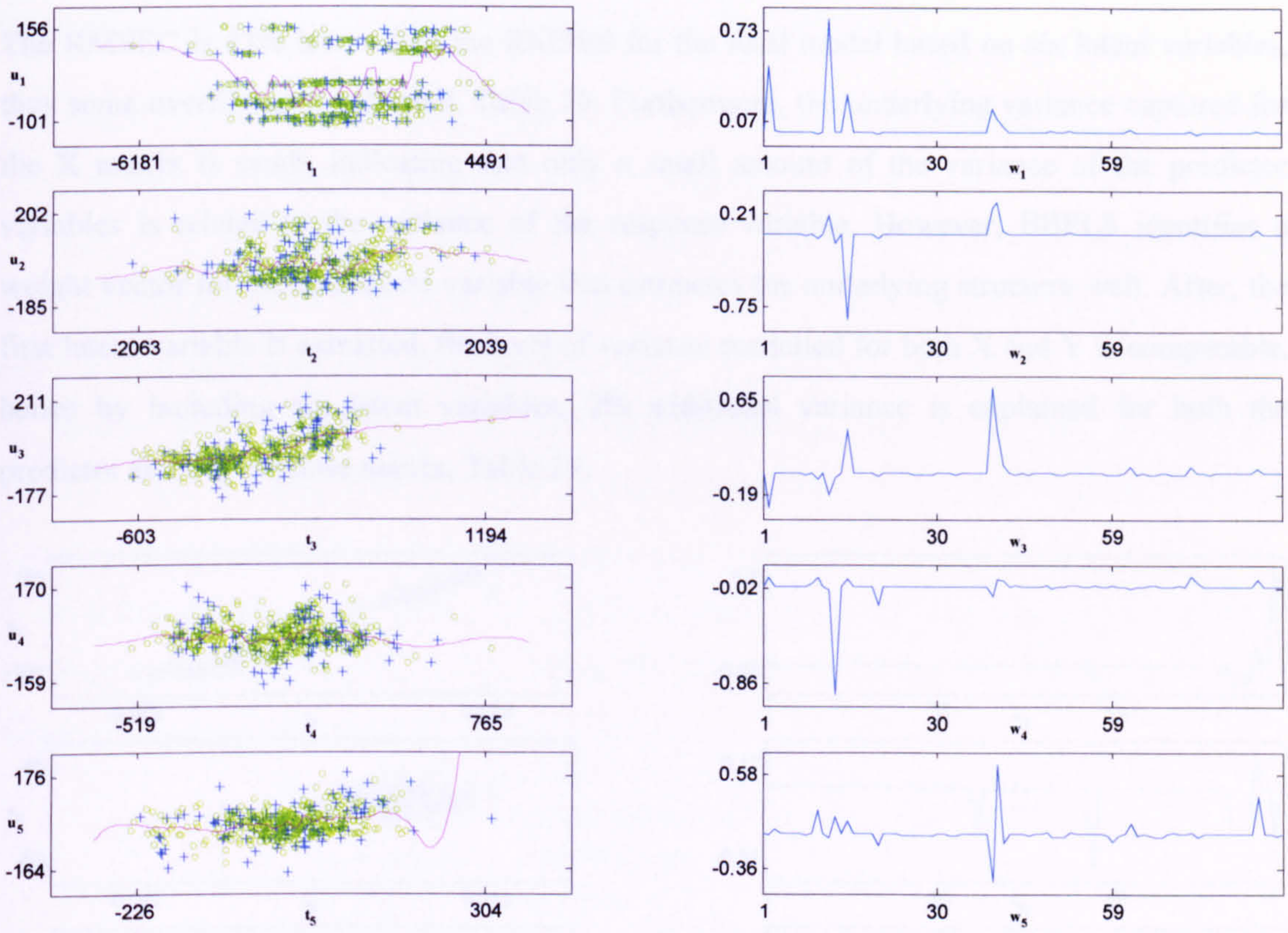

Figure 4.43. The t-u plots and the weight vectors for SPLS for the first five latent variables.

4.4.4 Error Based PLS

The EBPLS approach is expected to perform well since the data set is overdetermined and exhibits a low level of collinearity. The underlying structure is clearly identified and approximated well by the first latent variable, Figure 4.44. Including additional latent variables only improves the model slightly.

LV's	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	1.12	87.39	10.75	15.80
2	1.33	88.55	9.76	15.73
3	1.45	88.92	9.44	15.87
4	1.80	89.20	9.20	15.31
5	3.43	89.52	8.93	15.27
6	**3.45**	**89.92**	**8.59**	**15.14**
7	3.50	90.44	8.14	15.55

Table 29. Application of EBPLS to Melt Index data

The RMSEC is 43% lower than the RMSEP for the final model based on six latent variables, thus some overfitting is indicated, Table 29. Furthermore, the underlying variance captured for the **X** matrix is small, indicating that only a small amount of the variance of the predictor variables is related to the variance of the response variable. However, EBPLS identifies a weight vector for the first latent variable that estimates the underlying structure well. After, the first latent variable is extracted, the level of variance modelled for both **X** and **Y** is comparable, hence by including six latent variables, 2% additional variance is explained for both the predictor and the response matrix, Table 29.

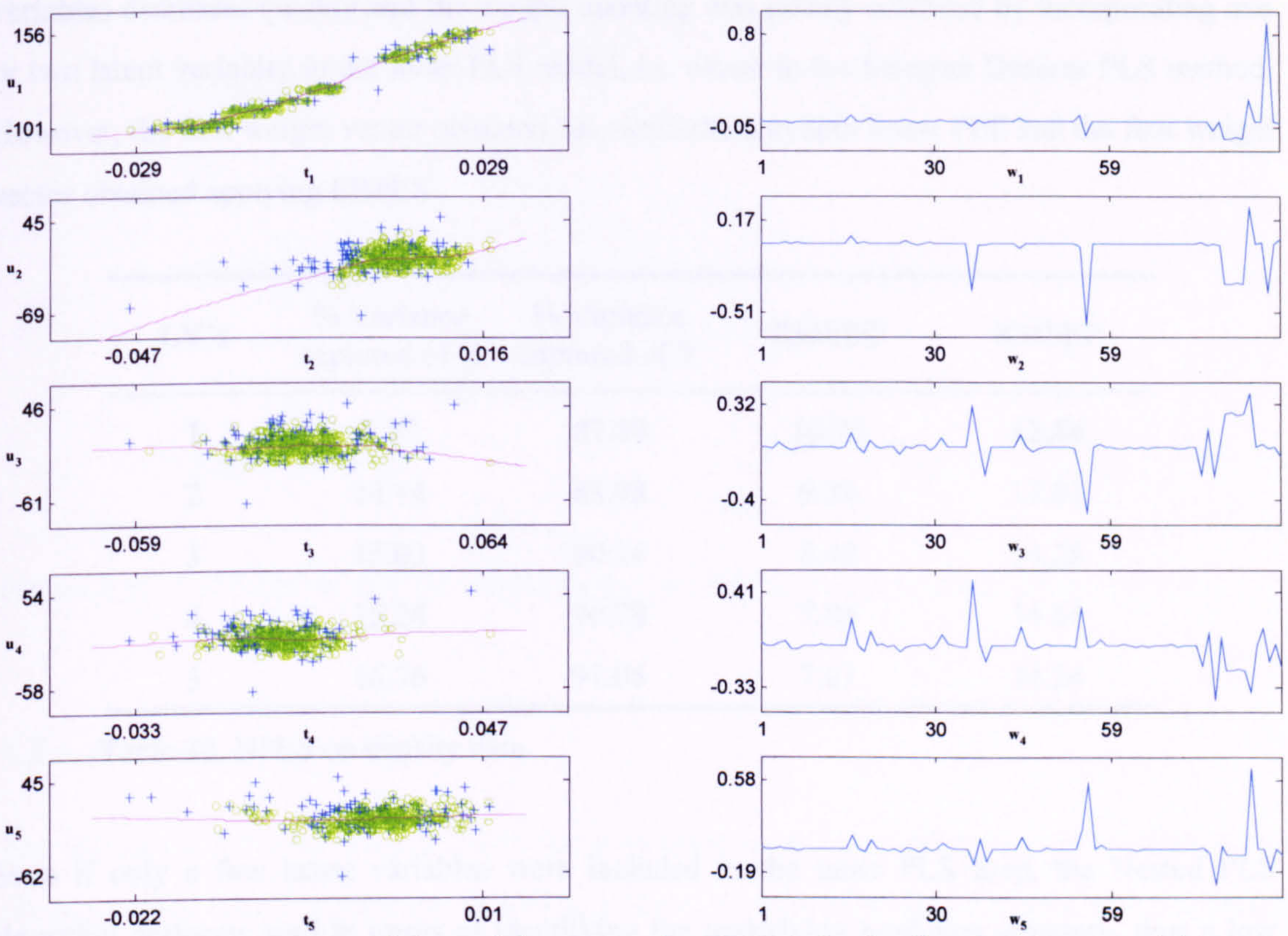

Figure 4.44. The t-u plots and the weight vectors for EBPLS for the first five latent variables.

It appears that the EBPLS algorithm terminates in a local minimum when constructing the first latent variable, since six latent variables are needed. This is reflected in the first weight vector that focuses on fewer variables compared with NPLS, thus there is still some variance to be modelled after the inclusion of the first latent variable. In particular, applying EBPLS results in higher RMSEP than applying NPLS, due to the high number of latent variables needed to be included in the model. Furthermore, for the second to the fifth latent variables, the focus is on the same predictor variables which are included for the first latent variable of NPLS. For further discussions, see the section on the influence of the starting vector, Section 4.4.7.

4.4.5 Nested PLS

The Nested PLS approach should provide a compromise between the updating vector of the full Gauss-Newton approach used in EBPLS and the steepest descent approach used in SDPLS. As this data set is overdetermined and has a low level of collinearity it was conjectured that the results from the Nested PLS algorithm would be closest to the Gauss-Newton approach of EBPLS, i.e. a large number of latent variables would be included in the inner PLS. Investigating the number of latent variables in the inner PLS loop showed that, for the first 10 iterations, between 4 and 7 latent variables were included. After 10 iterations, the number of latent variables decreases quickly and the weight updating was mainly achieved by incorporating one or two latent variables in the inner PLS model, i.e. closer to the Steepest Descent PLS method. However, the first weight vector obtained has similarities to both linear PLS and the first weight vector obtained applying EBPLS.

LV's	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	**1.27**	**87.39**	**10.75**	**12.84**
2	14.14	88.98	9.39	13.81
3	15.03	90.14	8.40	14.28
4	15.24	90.78	7.86	14.61
5	16.76	91.06	7.61	14.24

Table 30. NPLS on density data

Even if only a few latent variables were included in the inner PLS loop, the Nested PLS algorithm performs well in terms of identifying the underlying nonlinear structure, thus a low RMSEP value is attained, Table 30. Including additional latent variables did not improve the performance, with little structure being observed in the second t-u plot, Figure 4.45. The NPLS approach exhibits a smaller difference between the RMSEC and the RMSEP values than for the EBPLS method (RMSEC is 16% lower than RMSEP), indicating limited tendency to overfit the data. The first score vector is influenced by more variables than the other methods as observed from the weight vector, and the variance explained by the first latent variable is slightly higher than for EBPLS, Figure 4.45. Of particular note is the similarity between the two first weight vectors for NPLS, indicating that there is still some structure left in the data after application of the first latent variable. More specifically, different termination criteria or more subgroups in the cross validation of the inner PLS loop, could improve the model as some structure remains.

This will be investigated in subsequent sections, particular when examining the termination criteria, Section 4.4.8.

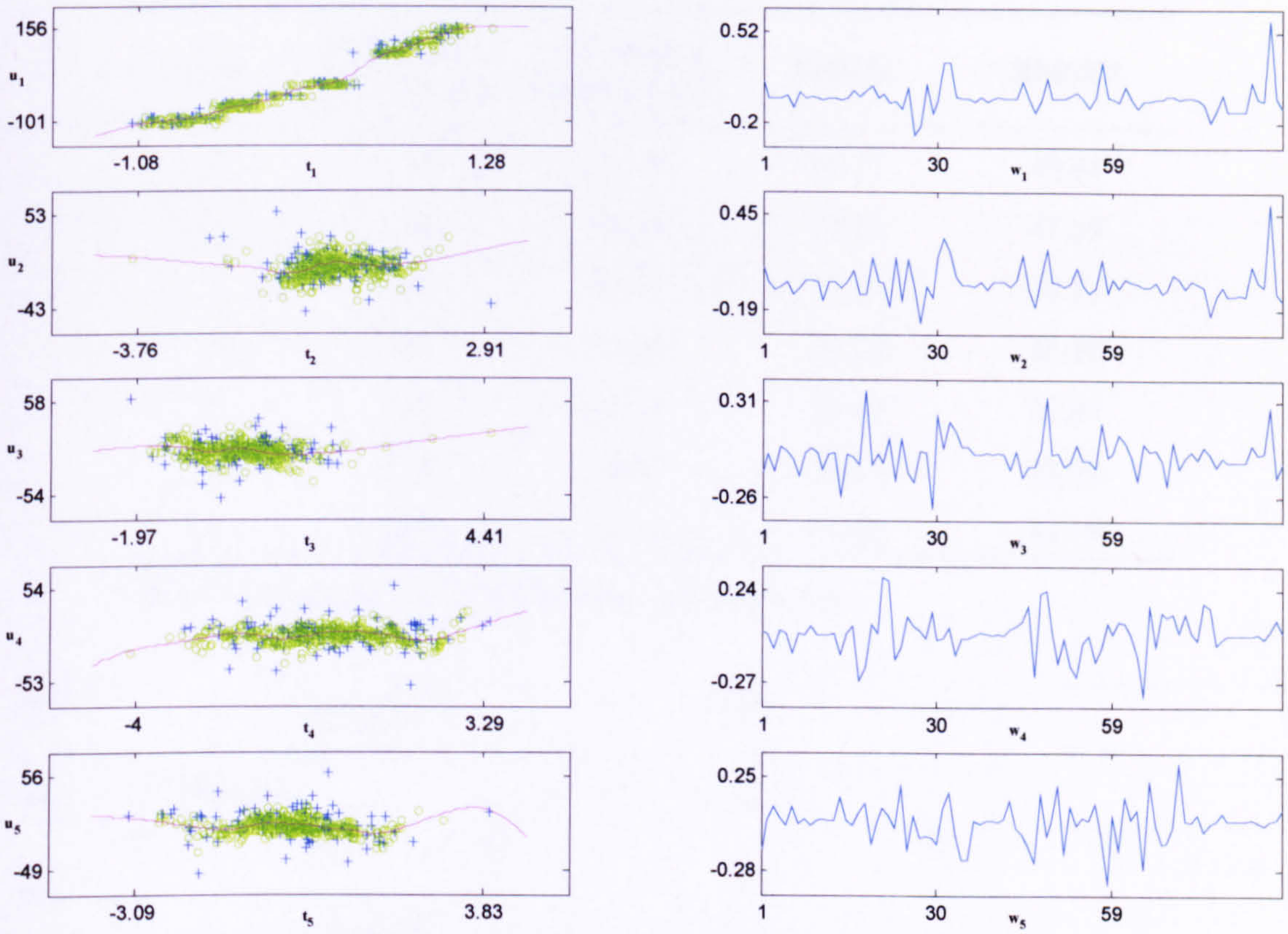

Figure 4.45. The t-u plots and the weight vectors for NPLS for the first five latent variables.

4.4.6 Steepest Descent PLS

The difference between using a single latent variable in the inner PLS loop as in SDPLS and allowing more latent variables as in NPLS, makes a considerable improvement in terms of the prediction performance for this data set. The final model is almost three times poorer in terms of the RMSEP compared with NPLS, Table 31. In particular, the first latent variable does not identify the underlying structure as well as Nested PLS, as observed from the first t-u plot, Figure 4.46. Again the issue is related to the problem of convergence. This is a consequence of low level of the predictor variance associated with the response value, combined with high noise levels from the process. In particular, the Melt Index value is determined by the longest molecules in the molecular weight distribution. Thus, the process variables do not directly describe the amount of the longest molecules, since they only constitute a small fraction of the total molecular weight distribution that is produced in the process. Furthermore, it may also be influenced by batch-to-batch deviations of the catalyst and that the data set consists of five different products. This information is not included in the model. SDPLS is strongly affected by local minima related to the above phenomena since no covariance information is used to obtain

the gradient information in calculating the step direction ($\partial \mathbf{w}$). The final model is achieved by including six latent variables.

LV's	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
1	1.49	53.34	39.77	49.61
2	4.30	59.80	34.26	47.59
3	4.82	67.70	27.53	37.77
4	5.94	71.58	24.22	35.42
5	7.05	74.63	21.62	32.97
6	**11.37**	**76.03**	**20.43**	**31.92**
7	12.03	77.16	19.46	32.18

Table 31. Application of SDPLS to the Melt Index data

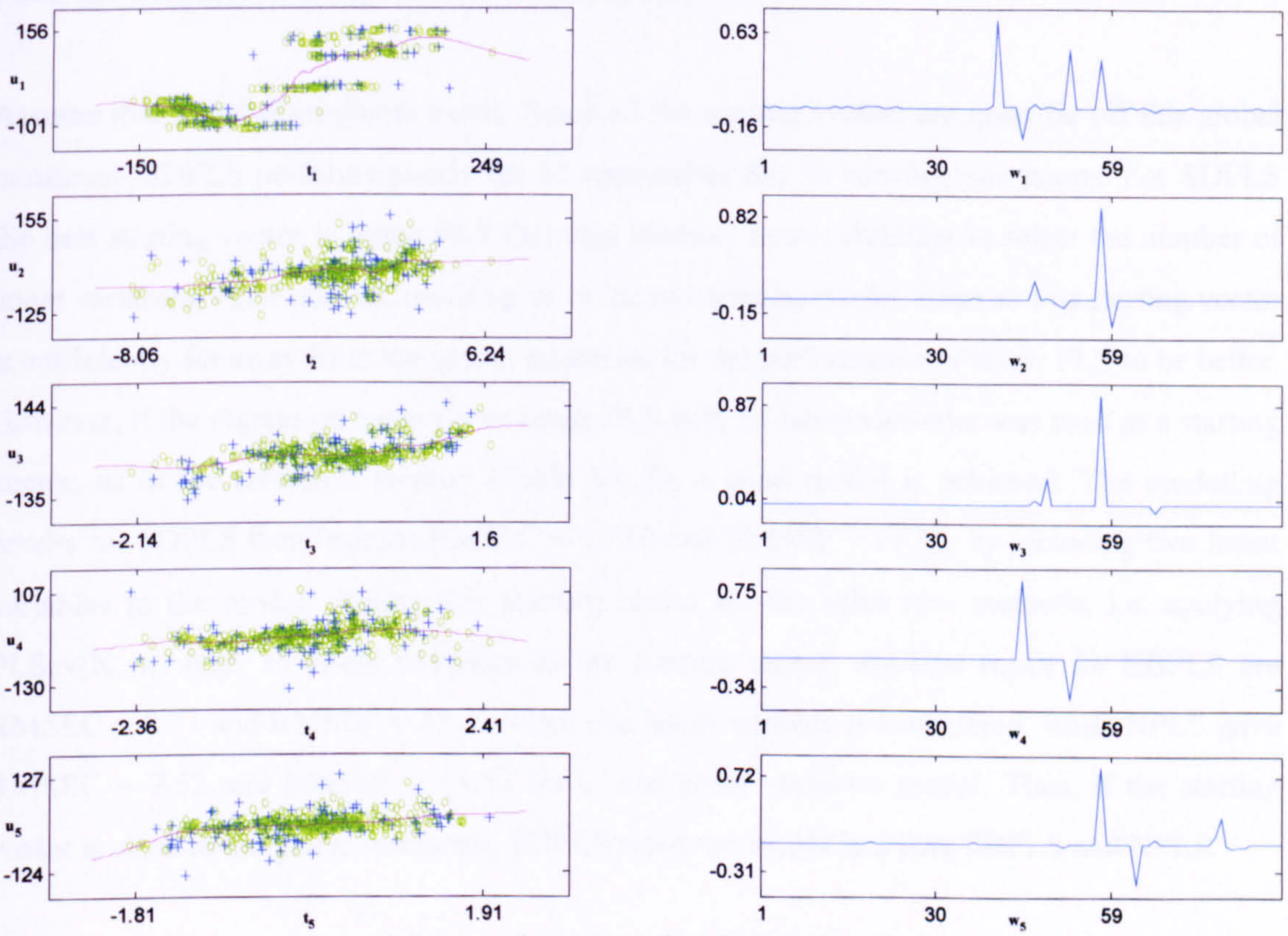

Figure 4.46. The t-u plots and the weight vectors for SDPLS for the first five latent variables.

The weight vectors gives strong emphasis to a few predictor variables, compared to EBPLS and NPLS. The first t-u plot indicates difficulties with the segregation of the different products (five clusters), that is handled better using Nested PLS and EBPLS.

4.4.7 Influence of the Starting Vector

The influence of the starting vector was again investigated for models belonging to Framework 1. Five different starting vectors were examined. Again, the spread of the final weight vector for the first latent variable and the prediction abilities of the final models formed the basis of the study. The results for the different starting vector are presented in Table 32.

Starting Vector	EBPLS RMSEC / RMSEP (lv)	NPLS RMSEC / RMSEP (lv)	SDPLS RMSEC / RMSEP (lv)
$\mathbf{w} \leftarrow \text{cov}(\mathbf{X}, \mathbf{u})$	12.73 / 22.52 (4)	12.03 / **14.26** (2)	28.74 / 45.08 (10)
$\mathbf{w} \leftarrow \text{cor}(\mathbf{X}, \mathbf{u})$[9]	8.59 / 15.14 (6)	10.75 / **12.84** (1)	20.43 / 31.92 (6)
$\mathbf{w} \leftarrow \text{PLScv}(\mathbf{X}, \mathbf{u})$	11.49 / **14.79** (5)	11.79 / 14.91 (2)	21.34 / 25.89 (8)
$\mathbf{w} \leftarrow \text{RVPLS}(\mathbf{X}, \mathbf{u})$	11.63 / 14.82 (8)	12.50 / **14.05** (1)	16.24 / 30.28 (12)
$\mathbf{w} \leftarrow \text{SPLS}(\mathbf{X}, \mathbf{u})$	11.77 / 14.84 (5)	12.65 / **14.58** (1)	20.08 / 43.41 (9)

Table 32. Best models using 5 different starting vectors.

Assume that a global minimum exists. Since all the starting vectors are quite far off this global minimum, SDPLS performs poorly for all approaches due to convergence issues. For SDPLS the best starting vector is linear PLS that uses internal cross validation to select the number of latent variables, PLScv(**X**, **u**), resulting in 15 latent variables model. Even so this starting vector is sufficiently far away from the global minimum for the performance of linear PLS to be better. However, if the regression vector from linear PLS with 35 latent variables was used as a starting vector, as in the reference method (Table 26, F), a good model is achieved. The modelling results for SDPLS then become RMSEC = 11.60 and RMSEP = 13.35, by including two latent variables in the model. Using this starting vector for the other two methods, i.e. applying PLScv(**X**, **u**) with 35 latent variables as the starting vector, the best result for EBPLS are RMSEC = 8.77 and RMSEP = 15.23 when one latent variable is considered, while NPLS gave RMSEC = 7.52 and RMSEP = 13.53 for a four latent variables model. Thus, if the starting vector is close to the global minimum, SDPLS seems to overfit less than EBPLS and NPLS.

Generally, NPLS gives the narrowest range between the RMSEP values for the different starting vectors. EBPLS performs well, except when using the covariance criterion as the starting vector. The average correlation between the final weight vector for the first latent variable using 5 different starting vectors are for EBPLS: 0.97 ± 0.04, for NPLS 0.89 ± 0.14 and finally for SDPLS: 0.38 ± 0.26, confirming the lack of consistency for the SDPLS approach relating to the convergence issue.

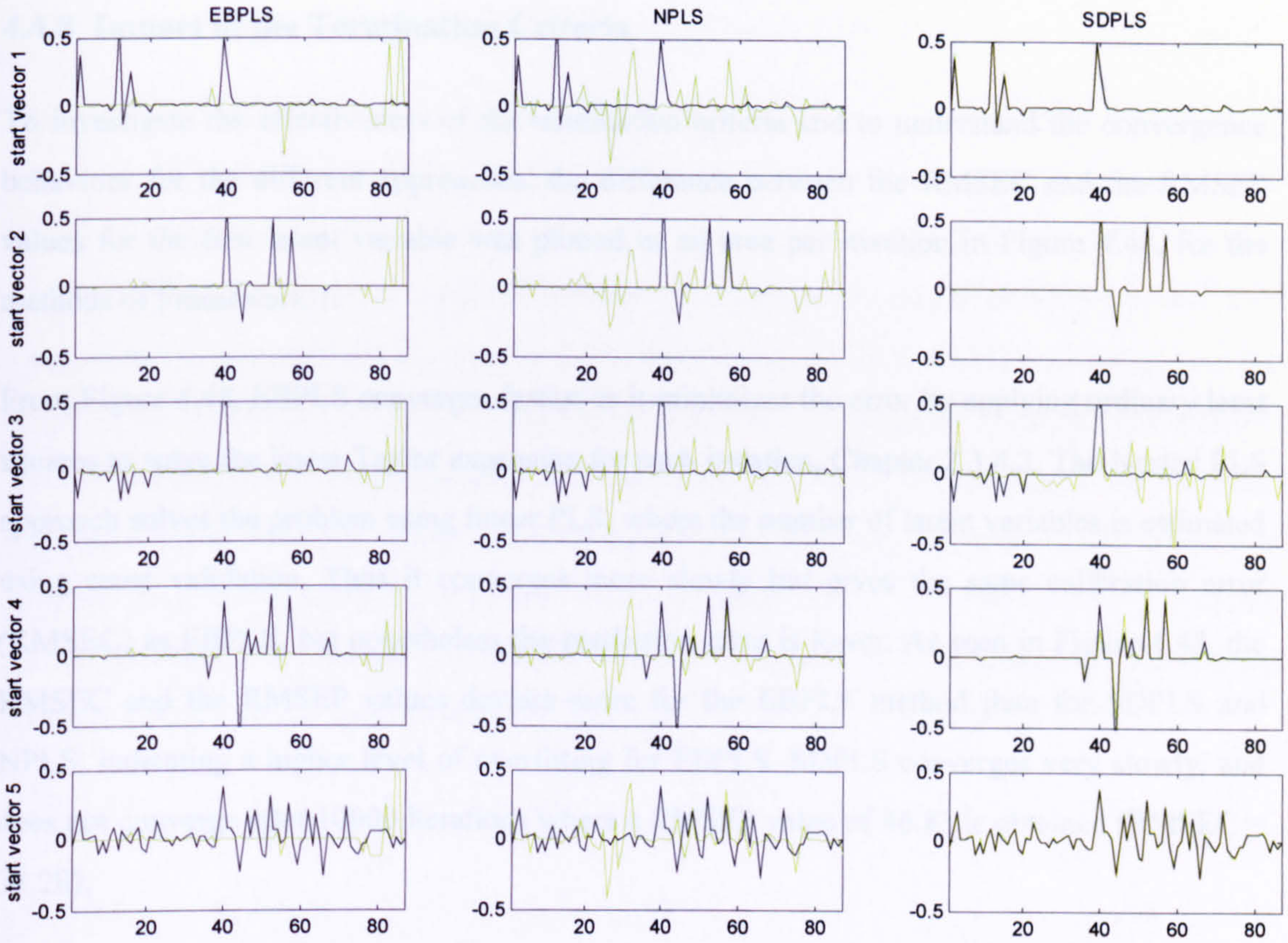

Figure 4.47. The starting vector (black) and the final weights (green) for the first latent variable.

Of particular note is the low number of latent variables used in Nested PLS, compared with EBPLS and SDPLS. The reason why NPLS uses fewer latent variables than EBPLS is not fully understood, but may have to do with the singularity problem in calculating the pseudoinverse of Equation (2.53). Of greater significance, is that the weight vector for the best NPLS model, i.e. applying starting vector two, gives emphasis to the area over variable number 80, in contrast to when using the other starting vectors, Figure 4.47. A possible explanation is that a higher number of latent variables is selected for the inner PLS when applying starting vector two. In particular, the highest number of latent variables achieved for the inner PLS, when applying starting vector two, was seven latent variables for the third iteration. The highest numbers of latent variables achieved for the inner PLS when applying the other starting vectors were located between 3 and 5.

From Figure 4.47, a first weight vector for SDPLS is obtained which it is only just possible to distinguish from the starting vectors, except when applying the regression coefficient from linear PLS with cross validation as the starting vector (start vector 3). The effect of dampened optimisation gave worse predictions for all methods and all start vectors and is not recommended for any of the methods; RVPLS, SPLS, EBPLS, NPLS and SDPLS. The reason is possibly the high level of noise in this data set.

4.4.8 Impact of the Termination Criteria

To investigate the effectiveness of the termination criteria and to understand the convergence behaviour for the different approaches, the difference between the RMSEC and the RMSEP values for the first latent variable was plotted as an area per iteration in Figure 4.48, for the methods of Framework 1.

From Figure 4.48, EBPLS converges fastest as it minimizes the error by applying ordinary least squares to solve the linear Taylor expansion for each iteration, Chapter 2.3.4.2. The Nested PLS approach solves the problem using linear PLS, where the number of latent variables is estimated using cross validation. Thus it converges more slowly but gives the same calibration error (RMSEC) as EBPLS, but nonetheless the prediction error is lower. As seen in Figure 4.48, the RMSEC and the RMSEP values deviate more for the EBPLS method than for SDPLS and NPLS, indicating a higher level of overfitting for EBPLS. SDPLS converges very slowly, and does not converge after 10000 iterations where a RMSEP value of 46.83 is obtained (RMSEC = 33.28).

Dampening did not improve any of the models. The choice of standard termination criteria does not influence the ranking of the methods. However, increasing the maximum number of iterations would improve the results of SDPLS, i.e. it is possible to reduce the RMSEP from 49.6 to 46.1 for the first latent variable by increasing the number of iterations.

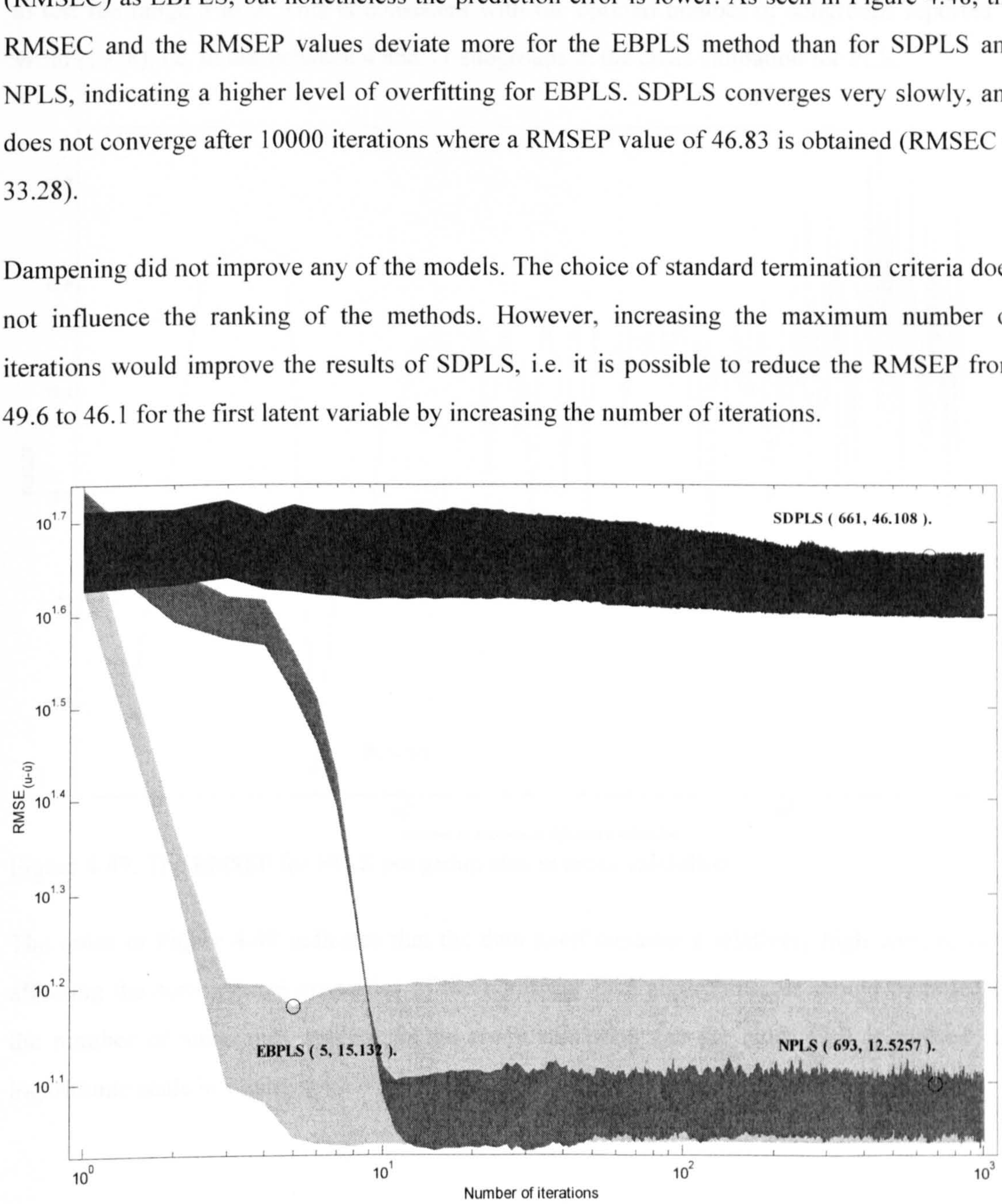

Figure 4.48. The difference of RMSEC and RMSEP for EBPLS(▪), NPLS(▪) and SDPLS(▪).

4.4.8.1 Effect of Cross Validation Group Size for Nested PLS

The prediction error is plotted as a function of group size for the application of cross validation in the inner PLS loop of Nested PLS. Once more an acceptable choice is around 6 to 8 subgroups, Figure 4.49. The best model occurred for 6 subsets, resulting in a RMSEP of 12.67. Moreover, independent of the different number of subgroups the NPLS model performed well since even the worst model comprising 179 subgroups (RMSEP = 13.87) was better than for EBPLS, the only other method that gave satisfactory results in terms of modelling this data set. That is, the number of subgroups has no consequence in terms of the ranking of the nonlinear PLS approaches. From the results obtained for the three data sets examined this chapter, it is difficult to propose a universal optimum number of subgroups. The recommendation would be to test the range 5 to 9. This is consistent with the optimal number of subgroups reported by Wold (1978), i.e. to use between 4 and 11 subgroups in the cross validation for PLS.

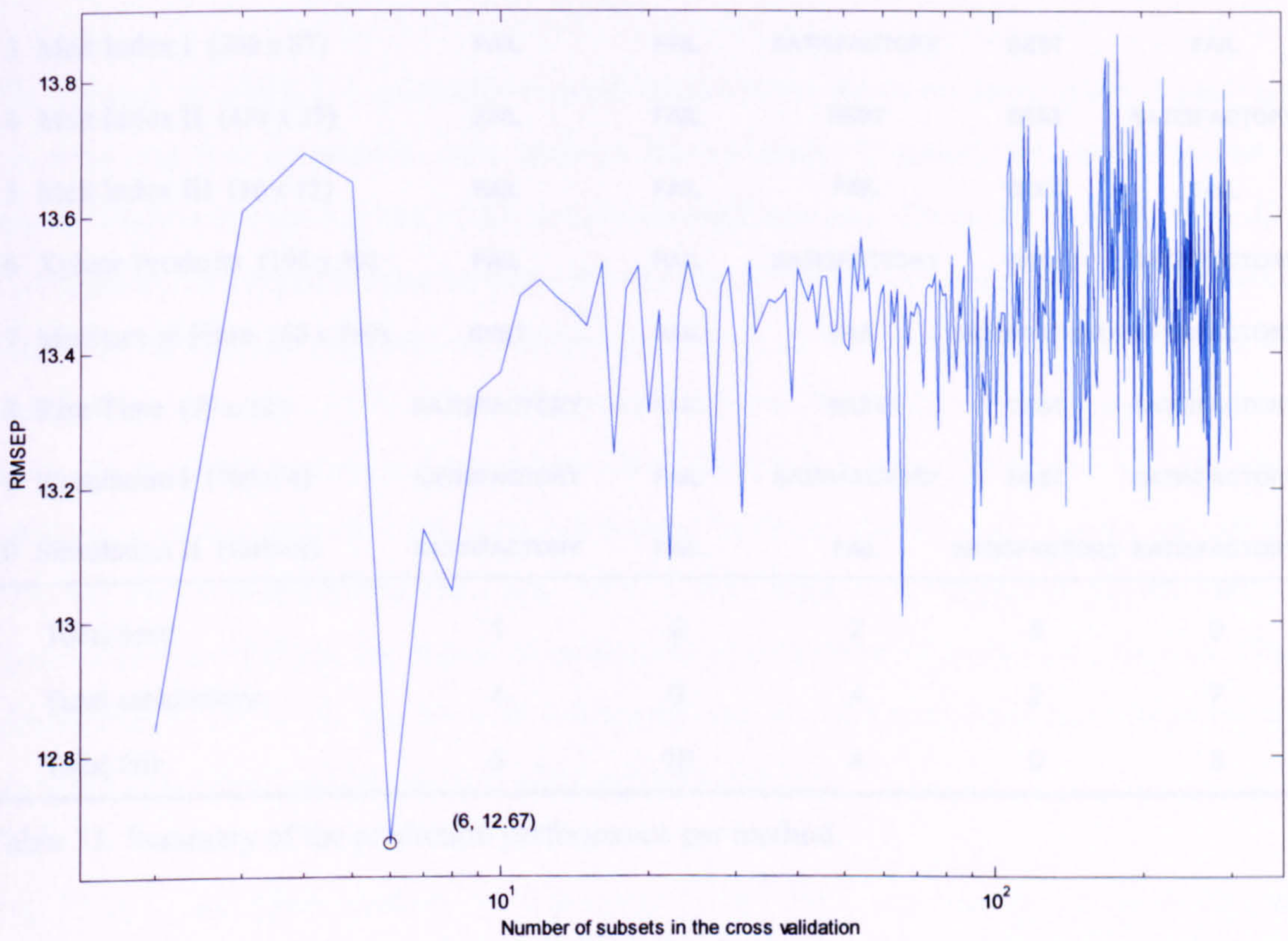

Figure 4.49. The RMSEP for NPLS per group size in cross validation.

The noise in Figure 4.49 indicates that the data itself contains a relatively high level of noise, affecting the convergence properties of the nonlinear PLS algorithms. It should be noted that the number of subgroups applied in the cross validation for the inner PLS is plotted on a logarithmic scale in Figure 4.49.

4.5 Discussion

The results obtained from modelling the ten data sets by applying the standard criteria are summarised in Table 33. The details of the analysis for the remaining seven data sets are presented in Appendix A1. In Table 33, the results are evaluated by assigning prediction performance into three categories; BEST, SATISFACTORY and FAIL. The RMSEP value that gives the best or equal best RMSEP value is denoted BEST. If the RMSEP is better than the reference method of linear PLS, and within 20% of the best model, the method is denoted SATISFACTORY, otherwise it is denoted FAIL.

Data sets	RVPLS	SPLS	EBPLS	NPLS	SDPLS
1 Polymer Density (87 x 301)	SATISFACTORY	FAIL	FAIL	BEST	SATISFACTORY
2 Alkylation Product (45 x 401)	FAIL	FAIL	SATISFACTORY	BEST	FAIL
3 Melt Index I (300 x 87)	FAIL	FAIL	SATISFACTORY	BEST	FAIL
4 Melt Index II (454 x 28)	FAIL	FAIL	BEST	BEST	SATISFACTORY
5 Melt Index III (50 x 15)	FAIL	FAIL	FAIL	BEST	FAIL
6 Xylene Products (196 x 30)	FAIL	FAIL	SATISFACTORY	BEST	SATISFACTORY
7 Moisture in Fibre (60 x 700)	BEST	FAIL	FAIL	SATISFACTORY	SATISFACTORY
8 Rise Time (99 x 12)	SATISFACTORY	FAIL	BEST	BEST	SATISFACTORY
9 Simulation I (700 x 4)	SATISFACTORY	FAIL	SATISFACTORY	BEST	SATISFACTORY
10 Simulation II (various)	SATISFACTORY	FAIL	FAIL	SATISFACTORY	SATISFACTORY
Total best:	1	0	2	8	0
Total satisfactory:	4	0	4	2	7
Total fail:	5	10	4	0	3

Table 33. Summary of the prediction performance per method.

As can be seen from Table 33, Nested PLS never fails, and is the best method for 8 of the 10 data sets. As the study represents a comprehensive range of different data sets with diverse characteristics, it is believed that these results demonstrate that Nested PLS can be applied as a universal nonlinear empirical regression technique. However, as the investigation of the effect of the termination criteria shows, Nested PLS has a tendency to overfit for underdetermined data sets. Of particular note, for data sets that exhibit weak nonlinearity, the first weight vector obtained using Nested PLS resembles the regression vector of linear PLS.

For Framework 1, Nested PLS is a balance between EBPLS and SDPLS. EBPLS is the most powerful method when it comes to convergence, but has a tendency to overfit the data due to either underdetermined data sets or multicollinearity. SDPLS, on the other hand, shows the weakest convergence properties as no covariance information is used to construct the updating vector, but has the least tendency to overfit. In Nested PLS, these two issues, convergence and overfitting, are addressed by the inner PLS loop whereby cross validation is applied to select the number of latent variables. If only one latent variable is selected the solution will be as for SDPLS. If all latent variables are applied the solution will be that of EBPLS. For any other number of latent variables selected, the solution will be between SDPLS and EBPLS.

After Nested PLS, the second best overall method was SDPLS. It failed for three of the data sets, due to convergence problems. Convergence may be improved through a better choice of the starting vector, but an appropriate choice is not easily identified.

The method of EBPLS is generally a good choice when there are more observations than variables and low correlation exits between the variables. However, the method failed to appropriately converge for one of the overdetermined data sets (Table 33: 5. Melt Index III). Furthermore, an increased ratio between the number of observations and number of variables decreases the effect of high level of collinearity, due to an improved condition number for the pseudoinverse (Almøy, 1996). For undetermined data sets, the EBPLS method generally failed, one exception was the second data set, the Alkylation Product data. The reason why it worked satisfactory on this data set is not clear, but it could be due to the low noise associated with the data set. For overdetermined data sets, EBPLS will work well depending on the ratio of observations to variables, the degree of collinearity and the level of noise in the data set. High dimensionality ($m >> n$), low level of collinearity and low signal to noise level improves the probability of achieving good models when applying EBPLS.

The RVPLS and SPLS methods of Framework 2 are often unsuccessful. In particular, the reference method of PLS performed better than SPLS for all ten data sets. For highly multicollinear data sets, RVPLS can perform satisfactorily, and even achieve the best result, e.g. data set 7 Table 33. RVPLS is not a universal method, but can generally be used to estimate a good starting vector for Nested PLS, especially for highly multicollinear data, such as spectral data.

Of the five methods Nested PLS has the greatest potential to become a universal nonlinear modelling technique. Detailed examination of the three data sets in Chapter 4, demonstrated that some caution must be exercised when applying Nested PLS as a universal nonlinear modelling technique:

- From the study it was observed that the results are dependent on the starting vector. However generally good convergence was obtained for Nested PLS. If necessary a restarting procedure could be applied, i.e. when the solution is believed to be poor another starting vector can be applied to check if a better result is obtained.

- The termination criteria may influence prediction performance. However, the standard termination criteria generally performed well, observed from the examination of the impact of the termination criteria. However, to use the lowest cross validation or test set error, for the selection of the number of iterations, may improve the convergence. Generally, a small number of iterations is recommended for Nested PLS. Dampening is not universally recommended, as convergence may be poorer. Instead internal cross validation can be used to control convergence. However, this increases the computation costs. An alternative to cross validation is to use a test set to terminate the algorithm and to determine the optimal number of cross validations in the inner PLS.

- The evaluation of the number of subgroups in cross validation of the inner PLS indicated that 5 to 9 subgroups is desirable. This is consistent with the number (4 to 11) suggested by Wold (1978) for linear PLS.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 Summary

The aim of the Thesis was to contribute to the field of nonlinear PLS which is not a well-established area. In particular, a nonlinear PLS algorithm that provides good universal performance was sought. Furthermore, the algorithm should be related to linear PLS, in structure and performance. Consequently, the issue of multicollinearity, underdetermined or overdetermined data sets, variables not measured, various signal to noise levels and all types of nonlinearities should all be appropriately handled by the resulting algorithm. Although several nonlinear PLS algorithms have been proposed, no particular algorithm has been shown to be generally applicable for a wide range of data sets. The literature survey confirmed this statement with very few applications having been published where nonlinear PLS algorithms have been applied.

The work presented comprises two new methods for the determination of the weight vector for nonlinear PLS. Both were developed from existing frameworks. Identifying an appropriate weight vector is a key issue in nonlinear PLS. Also of importance is the need to define an appropriate nonlinear mapping that reflects the underlying nonlinear structure inherent within the data. The nonlinear mapping is strongly influenced by the choice of weight vector. However, the method for determining the weight vector is independent of the choice of nonlinear mapping, hence any nonlinear function can be used. Thus, the focus was on the calculation of the weight vector. The nonlinear mapping selected for the comparison of the performance of these methods was a local linear kernel regression, which combines satisfactory performance with ease of calculation. Nevertheless, local linear kernel regression may give an inappropriate mapping at the extremes of the data cluster, increasingly so if the underlying function is not properly identified.

The first method proposed, Nested PLS (Li *et al.*, 2001), belongs to the same framework as the error based weight updating procedure (EBPLS) of Baffi *et al.* (1999a,b), and Steepest Descent PLS (SDPLS) of Wold *et al.* (1989). These methods are all variations of nonlinear least squares, where the weight vector is optimised in a weight updating scheme. This is achieved by minimising the sum-of-squared errors, i.e. minimising the expression:

$$\mathbf{e}^T\mathbf{e} = (\mathbf{u} - \hat{\mathbf{u}})^T(\mathbf{u} - \hat{\mathbf{u}}), \tag{5.1}$$

where $\mathbf{u} = \mathbf{Yq}$ and $\hat{\mathbf{u}} = f(\mathbf{Xw})$. These nonlinear PLS algorithms minimize the sum-of-squared errors sequentially, with the Y-loading vector, **q**, being updated in each iteration through least squares regression on $\hat{\mathbf{u}}$. The major benefit of this approach is that it directly minimizes the error associated with the inner mapping ($\mathbf{u}$ - $\hat{\mathbf{u}}$). Consequently, the error originating from fitting the nonlinear function, $\hat{\mathbf{u}} = f(\mathbf{t})$, for the inner mapping is decreased. The main drawback with these methods are those relating to nonlinear least squares, i.e. the problems of convergence versus local minima, singularity in the matrix inversion and potential overfitting.

Equation (5.1) is minimised differently for each of the methodologies in terms of solving the first order Taylor series estimation, thus the issues associated with nonlinear least squares are handled differently. In particular, SDPLS uses PLS applied to one latent variable, EBPLS uses PLS applied to all the latent variables, whilst NPLS uses PLS where the number of latent variables is selected from internal cross validation. Consequently, the difference in performance is from the different methodology used to solve Equation (5.1). PLS has shown its value for a range of applications, including multicollinear and underdetermined data sets, consequently the inner PLS loop of Nested PLS expands the capability to the nonlinear case.

The second method introduced, Reciprocal Variance PLS (RVPLS), belongs to the second framework that includes Spline PLS (SPLS) (Wold, 1992). In this paper, Wold extended the linear covariance criterion to the nonlinear situation (Section 2.3.4.3). This criterion does not focus on minimizing the error between $\mathbf{u}$ and $\hat{\mathbf{u}}$ (Equation, 5.1), but on simultaneously explaining the variance in **X** and **Y**. Consequently, the risk of introducing error when fitting the nonlinear function is greater than for the approach described in the optimisation framework common to SDPLS, EBPLS and NPLS.

In the proposed RVPLS approach (Hassel *et al.*, 2002), each weight vector is independently calculated in a similar manner to the covariance criterion. Thus, multicollinearity and underdetermined data set is of no concern, due to the individual calculation of the weight vector. However, as the reciprocal variance criterion focuses on explaining the response variance, the fitting error associated with the nonlinear mapping is reduced compared with SPLS. In particular, the approach will be successful for data sets that exhibit a relationship between the predictor and the response variables without interference from other constituents, as for some spectral data sets without overlapping bands or data sets exhibiting a high degree of

multicollinearity. Consequently, RVPLS is more applicable to FT-IR data sets than to complex NIR data sets that have overlapping bands.

To assess the effectiveness of the nonlinear PLS algorithms, ten data sets were selected to address the specific issues concerned with the two frameworks. In general, the Error Based PLS algorithm failed for underdetermined or multicollinear data sets, whilst the Steepest Descent PLS algorithm had convergence problems. The approach of Nested PLS enhances these two methods by finding a solution that lies between these two extremes. The problem of multicollinearity is prevented, and the problems of overfitting and convergence in local minima are significantly reduced.

As Nested PLS never completely failed for any of the ten data sets and gave either the best RMSEP value or a RMSEP value close to the best model, the method could be considered a universal nonlinear PLS algorithm. Still, caution must be shown to ensure global convergence by selecting an appropriate starting vector and to prevent overfitting by controlling the termination

For the second framework, RVPLS performs better than SPLS, but performance is highly dependent on the type of data modelled. The RVPLS method is only applicable when a simple relationship exists between the predictor and the response variables. For example, the RVPLS gives the best model for data set number seven, Table 33, where the response variables represent the moisture content in acrylonitrile-vinyl acetate polymer (Blanco, *et al.* 2000). This is a consequence of little interference between the water band (approx. wavelength 1940 nm) and other constituents in the NIR spectrum. Explicitly, the band at 1940 nm is an O-H stretching and bending combination band (Osborne and Fearn, 1988). Finally, wider application of the reciprocal variance approach could be to initiate the Nested PLS algorithm, by estimating an appropriate starting vector.

To conclude, Nested PLS outperformed the other algorithms examined in terms of prediction error (RMSEP), with the resulting weight vector resembling the regression vector obtained using linear PLS, when the underlying nonlinearity was weak. Thus, the aim of a universal nonlinear PLS algorithm appears to have been achieved.

5.2 Future Work

The Thesis focuses on two frameworks aligned with the original PLS algorithm of Wold *et al.* (1983), thus examination of alternative schemes has not been included. In particular, the Nested PLS algorithm proposed has been shown to be a wide-ranging nonlinear regression approach that enhances the performance of the methods on which it was based. However, a number of issues associated with the NPLS and the RVPLS approach give reason for further investigation.

- Applying the RVPLS method to construct a good starting vector showed promise for NPLS and would be a specific approach for multicollinear data. Other methods for finding an appropriate starting vector could be investigated, such as nonlinear PCR, e.g. using EBPLS or NPLS on a reduced rank matrix $\mathbf{T} = \mathbf{XW}$. The resulting regression coefficient, $\mathbf{b}$, could then be used to construct a starting vector, i.e. $\mathbf{w}_0 = \mathbf{Wb}$.

- The stopping criterion in NPLS could be improved, e.g. internal cross validation to supervise the convergence or the use of a test set. Furthermore, the effect of applying different types of cross validation for the inner PLS could be investigated.

- For NPLS, the multicollinearity problem relating to the first order Taylor series approximation was addressed by applying linear PLS in the inner loop, and where the number of latent variables was chosen using cross validation. This solution seems intuitive due to the association between the NPLS approach and ordinary linear PLS. However, a competitive linear regression method that deals with the problem of multicollinearity is Ridge Regression (Levenberg-Marquardt), or simply using linear PCR to solve the Taylor series expansion. Finally, the unified approach of Continuum Regression (Stone and Brooks, 1990) should be examined.

- The method of Reciprocal Variance Partial Least Squares did not have the universal appeal of NPLS, but could be applicable for spectral data set where the predictor variable is not influenced by overlapping bands. However, modifying the algorithm to simultaneously model a number of latent variables, by including it in a neural network algorithm is possible. The method is similar to a neural network as several "nodes" or latent variables are modelled simultaneously, but the RVPLS algorithm does not optimize the error directly hence the level of overfitting would be reduced.

- If the residuals are correlated when applying RVPLS, the performance of the algorithm is reduced. One idea is to include orthogonal signal correction (Wold *et al.*, 1998 and Fearn, 2000) in the algorithm to remove unwanted variance that could influence the method of determining the weight vector.

- This Thesis has been restricted to the current forms of the Nonlinear PLS algorithms, where the residuals of the response variables are modelled sequentially for each latent variable. Alternative approaches exist, e.g. to compress the original **X** matrix to its most relevant orthogonal factors, **T**, based on the response variables and use these compressed variables, **T**, as regressors for the responses, **Y**. This approach is aligned with linear PLS, since linear regression on the final orthogonal score matrix **T** (m x A) give identical results to sequential regression per score vector, i.e. $\mathbf{t}_j$, $j \in \{1 \cdots A\}$ as in Algorithm 2.2. A central ambition ought to be that any alternative nonlinear PLS algorithm proposed should not deviate considerably from the idea defined in the original PLS algorithm.

APPENDICES

A1. Modelling Results from Data Sets 4 to 10

In this section, the results from the supplementary data set are presented. The data sets are modelled using the standard criteria described in Section 4.1. Examination of these data sets is less comprehensive than for the three first data sets, discussed in Chapter 4. The reference methods are linear PLS and BPLS.

A1.1. Data Set 4, Melt Index II (NMR data)

This section examines Nuclear Magnetic Resonance (NMR) data collected on a polypropylene plant situated in Belgium. The data set consists of 772 observations and 7 variables, where six variables are fitted parameters from the NMR curve, and the seventh is a categorical variable representing catalyst type. The cross-terms of the fitted parameters were added so that the total number of variables was 28. From discussions with the owner of this data set, 14 observations or 1.8 % of the observations were removed as outliers. The calibration data set consists of 454 observations and the validation set comprises 304 observations, thus the data set is overdetermined. The response variable is Melt Index. The modelling results are presented in Table 34, while the t-u plot for the first latent variable and the corresponding weight vector is plotted in Figure A1.

Method applied	No. of latent variables	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
A: RVPLS	5	51.63	79.84	9.56	10.05
B: SPLS	5	52.74	81.66	8.70	9.84
C: EBPLS	1	11.34	90.92	4.30	5.13
D: NPLS	2	15.34	90.73	4.40	5.10
E: SDPLS	3	21.38	88.70	5.36	6.08
F: PLS	10	90.41	98.65	5.50	7.06
G: BPLS	9	11.58	89.64	4.92	6.08

Table 34. Comparison of the methods for the NMR data set.

The number of observations is much larger than the number of variables, i.e. EBPLS is expected to perform well. EBPLS is the best model together with NPLS. The result from SDPLS, that uses the same optimization framework, is somewhat poorer. Examining the weights W_{NPLS} obtained from NPLS shown in Figure A1, it appears that the structure is a compromise between

the weight vectors, W_{EBPLS} and W_{SDPLS} obtained from applying EBPLS and SDPLS, respectively. Since the structure of the weight vectors for EBPLS, NPLS and SDPLS is similar the resulting t-u plots are comparable, where the structure for the t-u plot of the first latent variable obtained for NPLS is a compromise between the two t-u plots from applying EBPLS and SDPLS.

The second framework that includes SPLS and RVPLS, does not produce satisfactory results and the performance from applying the two methods is similar. The reason why these two methods do not perform well is observed from the first t-u plot in Figure A1. For both RVPLS and SPLS, the nonlinear relationship is not properly identified, hence fitting a nonlinear curve to the data introduces an error that is not recovered by including more latent variables.

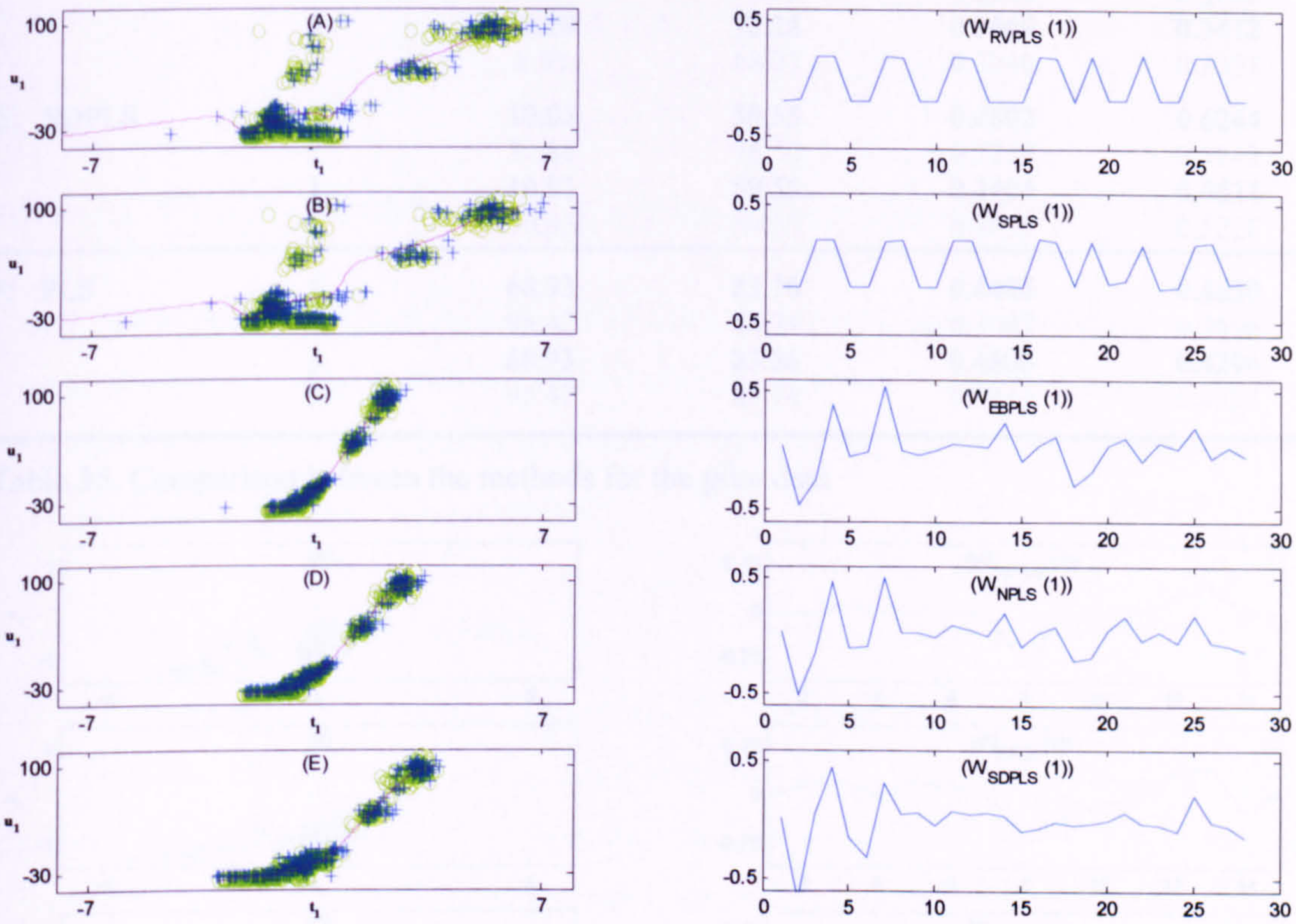

Figure A1. t_1-u_1 plot and plot of corresponding weight vector for the NMR data

A1.2. Data Set 5, Melt Index III (Pilot data).

The data set is generated from a polyolefin pilot plant. The predictor variables are different process variables relating to pressures, temperatures and concentrations. The data set contains 4 response variables, **Y** (67 x 4), and 15 predictor variables, **X** (67 x 15). Since it had only 67 observations, the calibration set was chosen to include 50 observations and the validation set 17 observations. Both the **X** and **Y** variables were autoscaled prior to the analysis. The response variables represent different Melt Index values. The fact that the responses are correlated makes it reasonable to form a multivariate model and not to model the responses separately.

Method applied	No. of latent variables	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
A: RVPLS	7	64.17	72.21	0.3177	0.5027
	7	64.17	72.21	0.2776	0.3984
	5	47.62	69.56	0.4702	0.4428
	7	64.17	72.21	0.2546	0.4751
B: SPLS	5	48.97	69.28	0.3701	0.5186
	1	11.93	40.56	0.6255	0.4326
	1	11.93	40.56	0.5205	0.4422
	2	24.75	51.34	0.3935	0.4919
C: EBPLS	1	8.86	70.61	0.3461	0.5066
	1	8.86	70.61	0.2438	0.4012
	2	20.98	74.55	0.3475	0.3743
	2	20.98	74.55	0.2122	0.3900
D: NPLS	2	20.29	72.28	0.3338	0.4346
	2	20.29	72.28	0.2232	0.3062
	2	20.29	72.28	0.3669	0.3412
	1	8.89	68.27	0.3040	0.3551
E: SDPLS	1	10.03	59.55	0.4802	0.6244
	3	32.88	76.50	0.2259	0.5242
	1	10.03	59.55	0.4404	0.4621
	1	10.03	59.55	0.3630	0.5228
F: PLS	5	68.93	83.36	0.4486	0.4630
	10	95.42	88.74	0.3047	0.3596
	5	68.93	83.36	0.4806	0.4296
	10	95.42	88.74	0.3111	0.4327

Table 35. Comparison between the methods for the pilot data

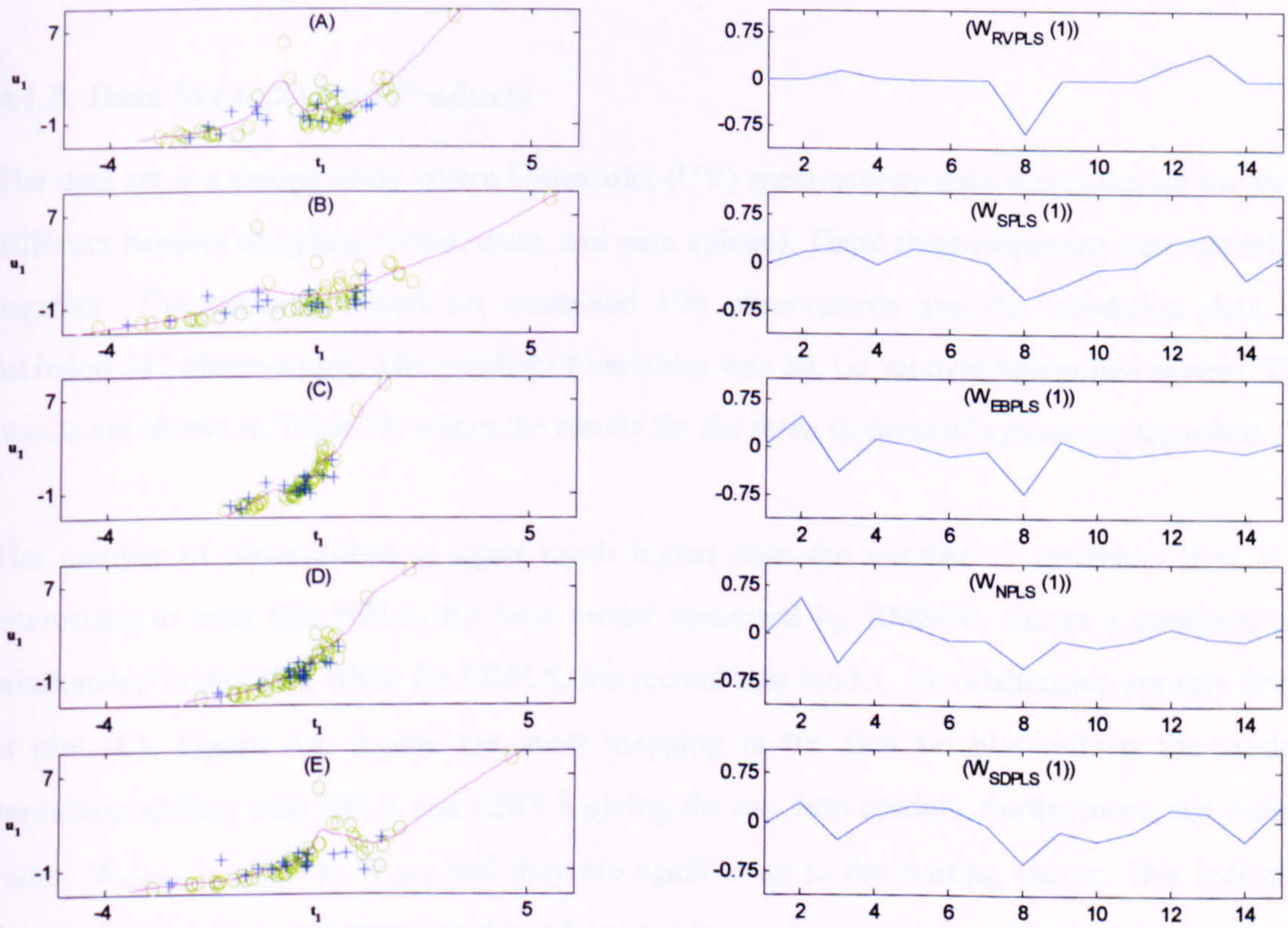

Figure A2. t_1-u_1 plot and plot of corresponding weight vector for the pilot data

In terms of increasing RMSEP value, the order of the methods is, NPLS, PLS, EBPLS, RVPLS, SPLS, SDPLS. Of particular note, SDPLS gives the worst model, due to termination in a local minimum. In Figure A2 (D) it can be observed why NPLS gives the best model, i.e. it identifies the underlying nonlinearity of the data better than the other methods. The nonlinear function shows a smooth curve, whilst the other methods have non-regular shapes. As a result, only NPLS performs better than the reference method of PLS. The methods are not capable of modelling the error introduced by the nonlinear function as a result of including more latent variables to the models. Thus, it is vital that the underlying structure is approximated in the best possible way, Figure A2. The weight vectors of the first latent variable show some degree of similarity, with weights obtained from EBPLS, NPLS and SDPLS being most similar.

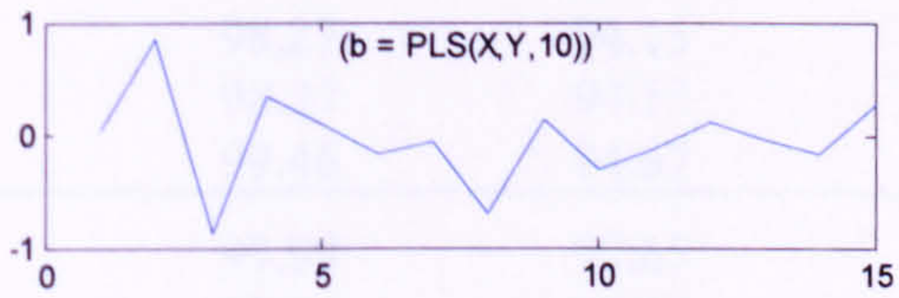

Figure A3. The regression vector from PLS including 10 latent variables.

The regression vector obtained from liner PLS when applying ten latent variables, Figure A3, is compared with the weight vectors obtained for the first latent variables, Figure A2. The regression vector from PLS is similar to the weight vectors obtained for the first latent variables when applying EBPLS and NPLS, Figure A2.

A1.3. Data Set 6, Xylene Products

The data set is a design study where Ultraviolet (UV) spectroscopy data was collected for three different isomers of xylene (ortho, meta, and para xylene). These three responses were modelled together. The calibration data set contained 196 observations and the validation data set included 147 observations. The number of variables was 30, i.e. an overdetermined system. The results are shown in Table 36, where the results for the three isomers of xylene are separated.

The number of observations is again much higher than the number of variables. Here it is interesting to note that NPLS, the best model measured by RMSEP, shows a nonlinear t-u relationship in plot (D), while for EBPLS, the second best model, the relationship appears linear in plot (C), Figure A4. Again, the inner mapping in the first t-u plot reflects the models prediction ability, with NPLS and EBPLS giving the two best models. Furthermore, the weight vector W_{SDPLS} is close to W_{SPLS} and they are again close to the starting vector. This indicates that the SDPLS has again terminated in a local minimum, in the vicinity of the starting vector.

Method applied	No. of latent variables	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
A: RVPLS	10	99.68	92.79	0.0010	0.0010
	8	99.59	92.54	0.0009	0.0009
	6	99.29	91.80	0.0010	0.0009
B: SPLS	10	99.76	91.31	0.0012	0.0009
	5	99.13	91.29	0.0010	0.0013
	4	98.88	90.23	0.0006	0.0009
C: EBPLS	8	87.63	94.80	0.0008	0.0008
	5	87.42	94.74	0.0007	0.0009
	8	87.63	94.80	0.0002	0.0004
D: NPLS	4	92.45	94.56	0.0008	**0.0007**
	4	92.45	94.56	0.0007	**0.0008**
	3	84.77	94.09	0.0002	**0.0003**
E: SDPLS	5	98.27	94.15	0.0008	0.0009
	5	98.27	94.15	0.0008	0.0010
	8	99.46	94.67	0.0002	0.0004
F: PLS	19	99.99	99.65	0.0008	0.0011
	21	99.99	99.67	0.0007	0.0009
	17	99.99	99.65	0.0002	**0.0003**

Table 36. Comparison between the methods for the xylene data

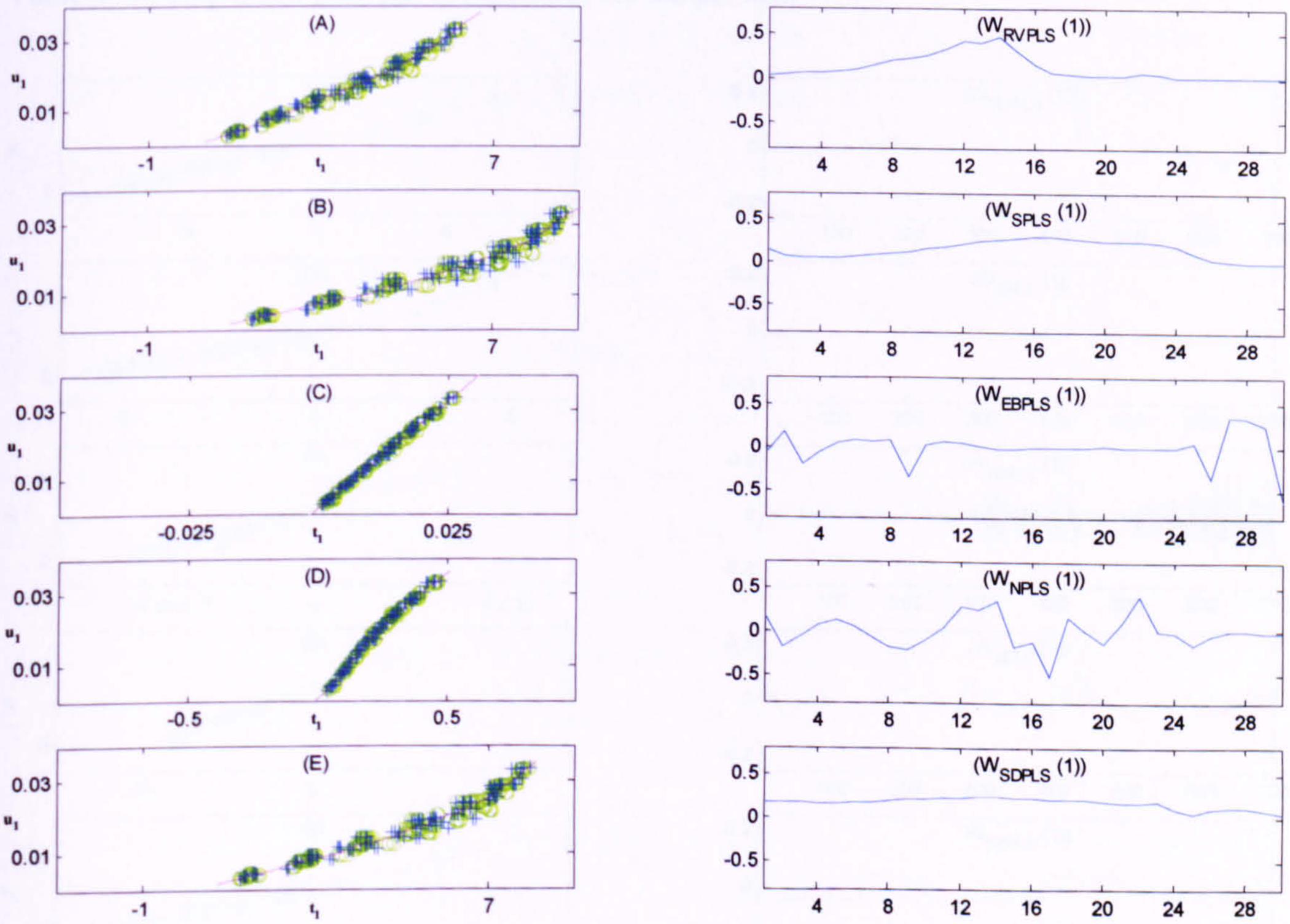

Figure A4. t_1-u_1 plot and plot of corresponding weight vector for the xylene data

The difference in the weight vector of W_{NPLS} and W_{EBPLS} is notable, with focus on different variables. Consequently, the inner mappings of the t-u plots (C and D) Figure A4, are different.

A1.4. Data Set 7, Moisture in Fibre

A diffuse reflectance near infrared data set generated by Blanco *et al.* (2000) was studied. The calibration set contained 60 observations and the validation data set, 17 samples. The number of variables was 700. Thus, the data set was underdetermined, i.e. the number of observation was less than the number of variables. The response variables represent the moisture content in acrylonitrile-vinyl acetate polymer.

Method applied	No. of latent variables	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
A: RVPLS	8	99.12	95.19	0.0690	0.0779
B: SPLS	1	17.02	90.478	0.1368	0.0966
C: EBPLS	3	98.08	99.99	0.000007	0.2949
D: NPLS	3	50.74	94.68	0.0773	0.0807
E: SDPLS	4	98.73	94.11	0.0845	0.0791
F: PLS	5	99.99	99.48	0.1034	0.0829
G: BPLS	3	89.29	94.31	0.1262	0.0816

Table 37. Comparison between the methods for the pH data

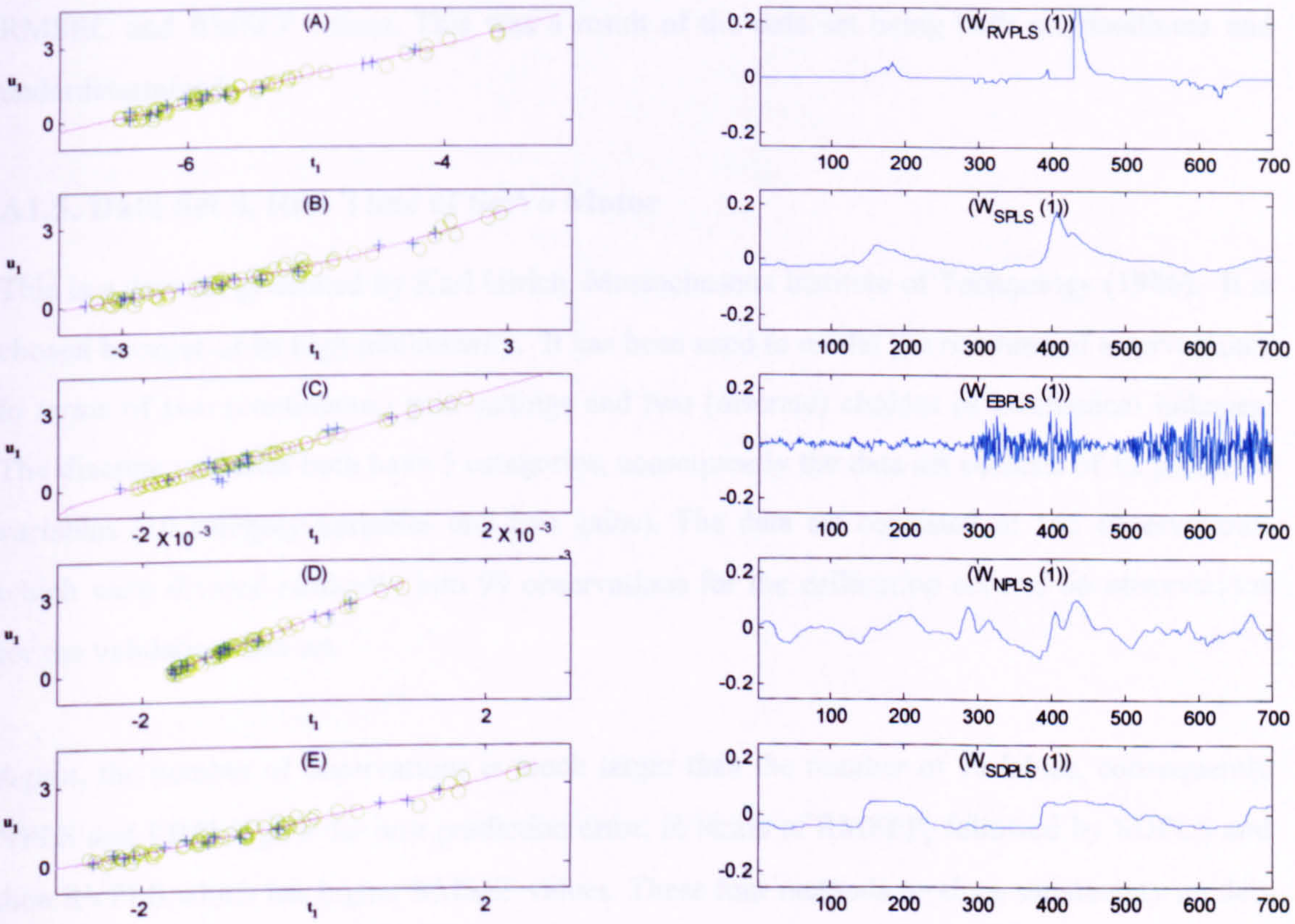

Figure A5. t_1-u_1 plot and plot of corresponding weight vector for the acrylic data

In general, the nonlinear PLS methods investigated resulted in a large difference between the structure of the obtained weight vectors for this particular data set, Figure A5. The level of nonlinearity was either small or insignificant for the different methods for the first pair of latent variables as seen from the t-u plots in Figure A5 (A – E). The best REMSEP value is obtained by applying RVPLS, Table 37. The reason why RVPLS method gives a better model for this data set is due to the strong water band at wavelength 1940 (variable 440) in the NIR spectrum. More specifically, since the water molecules are occurring freely in the polymer, the main interactions between the water molecules will be hydrogen bonding which makes a fairly simple spectral signature with an uncomplicated relationship with the concentration, due to minor influence from overlapping bands of other constituents. This makes the data set closer to the starting point in terms of the theoretical assumptions used to develop the RVPLS method, i.e. independent measurement error. The simple weight vector W_{RVPLS} in Figure A5 confirms this, as the focus is on the known band generated by water.

The prediction results using SDPLS and NPLS follow closely, and are better than the reference method of linear PLS. SPLS displays the greatest nonlinear behaviour in the t-u plot, but the prediction result is considerable less than that of linear PLS. The EBPLS fails. This is confirmed by the noisy weight vector W_{EBPLS} in Figure A5, and by the large difference between the RMSEC and RMSEP values. This was a result of the data set being both multicollinear and underdetermined.

A1.5. Data Set 8, Rise Time of Servo Motor

This is a data set generated by Karl Ulrich, Massachusetts Institute of Technology (1986). It is chosen because of its high nonlinearity. It has been used to model the rise time of a servomotor in terms of two (continuous) gain settings and two (discrete) choices of mechanical linkages. The discrete variables both have 5 categories, consequently the data set consists of 12 predictor variables (10 category variables and two gains). The data set consisted of 165 observations, which were divided randomly into 99 observations for the calibration set and 66 observations for the validation data set.

Again, the number of observations is much larger than the number of variables, consequently NPLS and EBPLS give the best prediction error, in terms of RMSEP, followed by SDPLS and then RVPLS which has higher RMSEP values. These four methods produce satisfactory models that are similar in structure. The SPLS model has a RMSEP value twice as high as the RMSEP value obtained by the RVPLS model. This is reflected in the t-u plot in Figure A6(B). The structures of the weight vectors for the first latent variable are similar, with the weight vector

W_{SPLS} being the most different, Figure A6. Consequently, the t-u plots display similar nonlinear structure, with the t-u plot in Figure A6(B) being the most different. This is due to the SPLS method focus on explaining both the **X** and **Y** variance, as observed from the high **X** variance included in the SPLS model. Due to the high nonlinearity, linear PLS gives the worst model.

Method applied	No. of latent variables	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
A: RVPLS	1	6.87	70.35	0.4295	0.4455
B: SPLS	5	30.42	71.62	0.4110	0.9034
C: EBPLS	2	10.99	87.55	0.1803	0.3575
D: NPLS	2	12.12	86.28	0.1987	0.3516
E: SDPLS	2	12.70	89.27	0.1554	0.4193
F: PLS	9	90.67	59.29	0.9242	0.9988
G: BPLS	3	4.44	33.15	0.9468	0.9468

Table 38. Comparison between the methods for the servo data

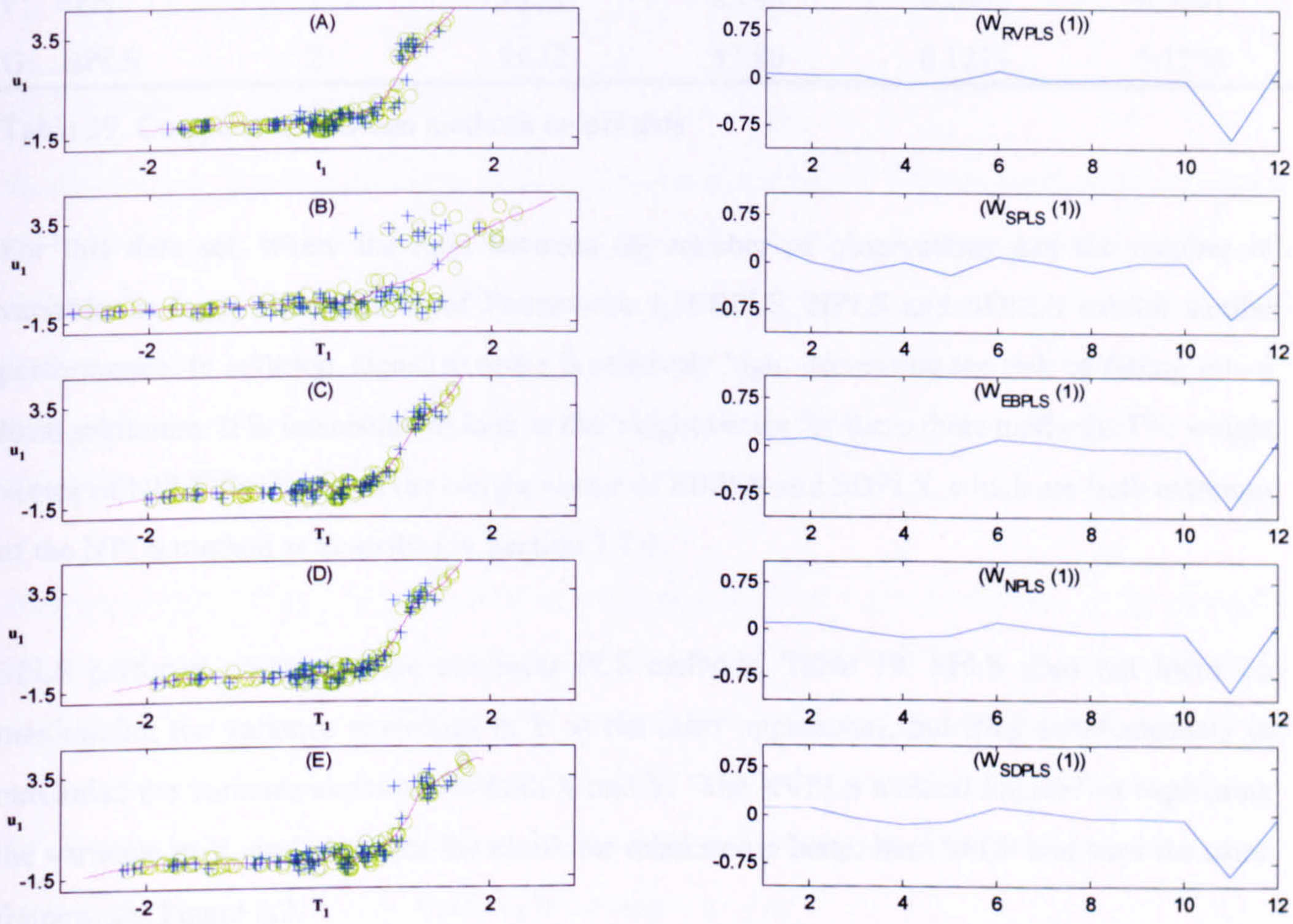

Figure A6. t_1-u_1 plot and plot of corresponding weight vector for the servo data

A1.6. Data Set 9, Simulation I (pH).

This data set was used in the paper by Baffi *et al.* (1999b. The data is from a dynamic simulation of a pH process described by Henson and Seborg (1994). A strong acid (HNO3) is neutralised by a strong base (NaOH) in the presence of a buffer (NaHCO3). The data set is described in more detail by Baffi *et al.* (1999b). It consists of 4 predictor variables, representing different flows, having 999 observations. The calibration data set has 700 observations and the validation data set, 299 observations. The single response variable is pH.

Method applied	No. of latent variables	% Variance captured of X	% Variance captured of Y	RMSEC	RMSEP
A: RVPLS	4	100.00	95.78	0.0421	0.0429
B: SPLS	4	100.00	93.16	0.0683	0.0808
C: EBPLS	1	35.23	97.33	0.0267	0.0306
D: NPLS	1	35.24	97.53	0.0247	0.0277
E: SDPLS	1	35.25	97.32	0.0268	0.0312
F: PLS	1	59.21	83.48	0.2552	0.2581
G: BPLS	2	31.12	87.86	0.1213	0.1256

Table 39. Comparison between methods on pH data

For this data set, where the ratio between the number of observations and the number of variables is large, the methods of Framework 1, EBPLS, NPLS and SDPLS exhibit similar performance. In addition, signal to noise is relatively high, decreasing the risk of falling into a local minimum. It is interesting to look at the weight vector for these three methods. The weight vector of NPLS lies between the weight vector of EBPLS and SDPLS, which are both extremes of the NPLS method as described in Section 3.2.4.

SPLS performs poorest of the nonlinear PLS methods, Table 39. SPLS does not focus on maximizing the variance explained in **Y** as the other approaches, but tries simultaneously to maximize the variance explained in both **X** and **Y**. The RVPLS method focuses on explaining the variance in **Y**, and identifies the nonlinear relationship better than SPLS that uses the same framework, Figure A7.

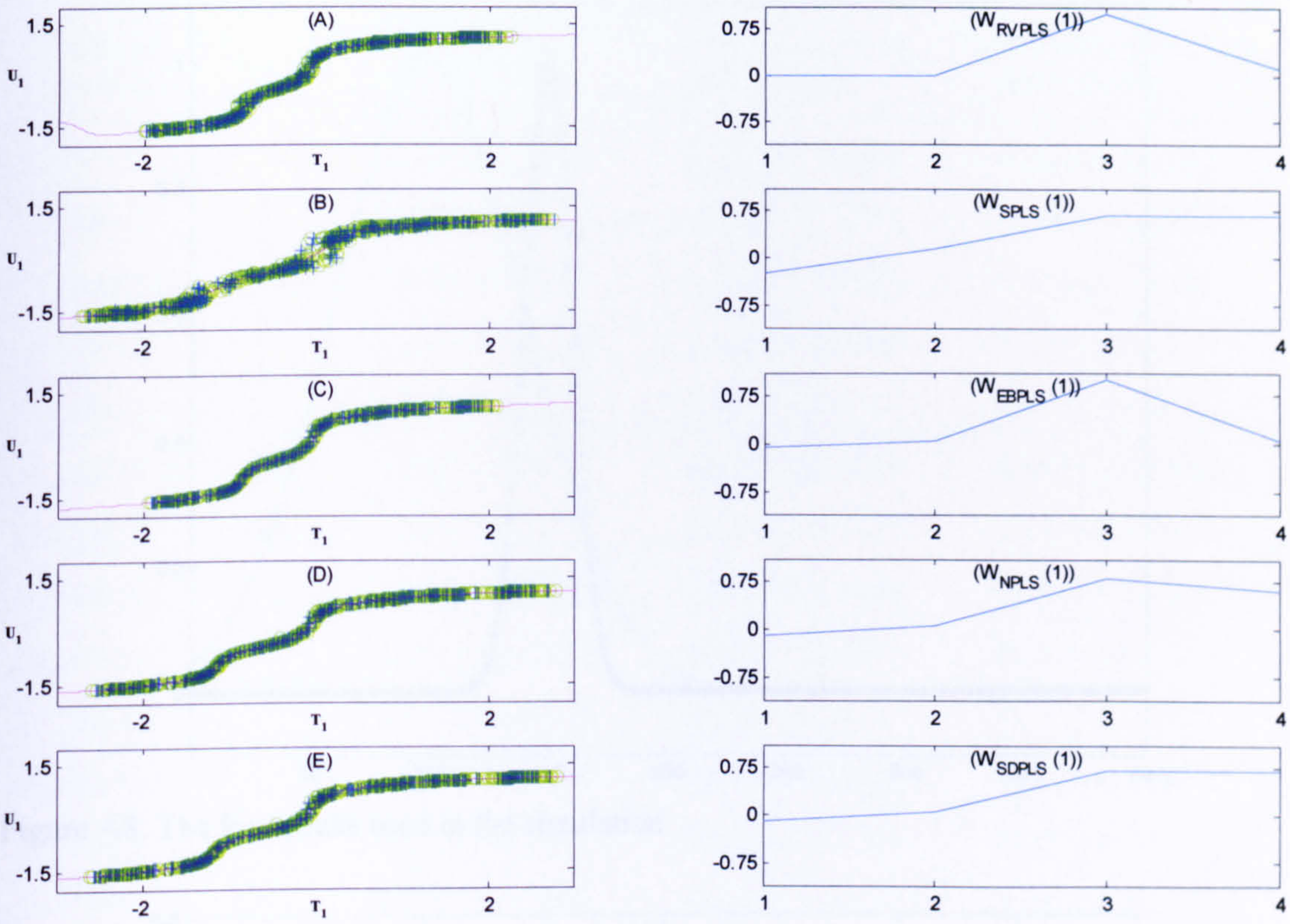

Figure A7. t_1-u_1 plot and plot of corresponding the weight vector for pH data

A1.7. Data Set 10, Simulation II

In this section, the five nonlinear PLS approaches described in Table 4 were compared using simulated spectral data. This simulation favours the reciprocal variance criterion, since only one peak in the spectrum is related to the response values, Chapter 3.3. Thus the method is expected to perform well on this simulated data. The spectral data set is constructed from four peaks, where the second peak, positioned at variable number 100, is related to the single response through a nonlinear function, Figure A8.

Every spectrum of the simulation data set was constructed as a sum of the four peaks shown in Figure A8. To make the simulated data set more realistic, different types of noise and effects were added. The noise included both multiplicative and additive forms, and the effects examined included bandwidth position, bandwidth shift and nonlinear baseline effects. In Figure A9, a selection of 16 spectra having medium noise is plotted to illustrate the level of noise added. The response was only given additive noise.

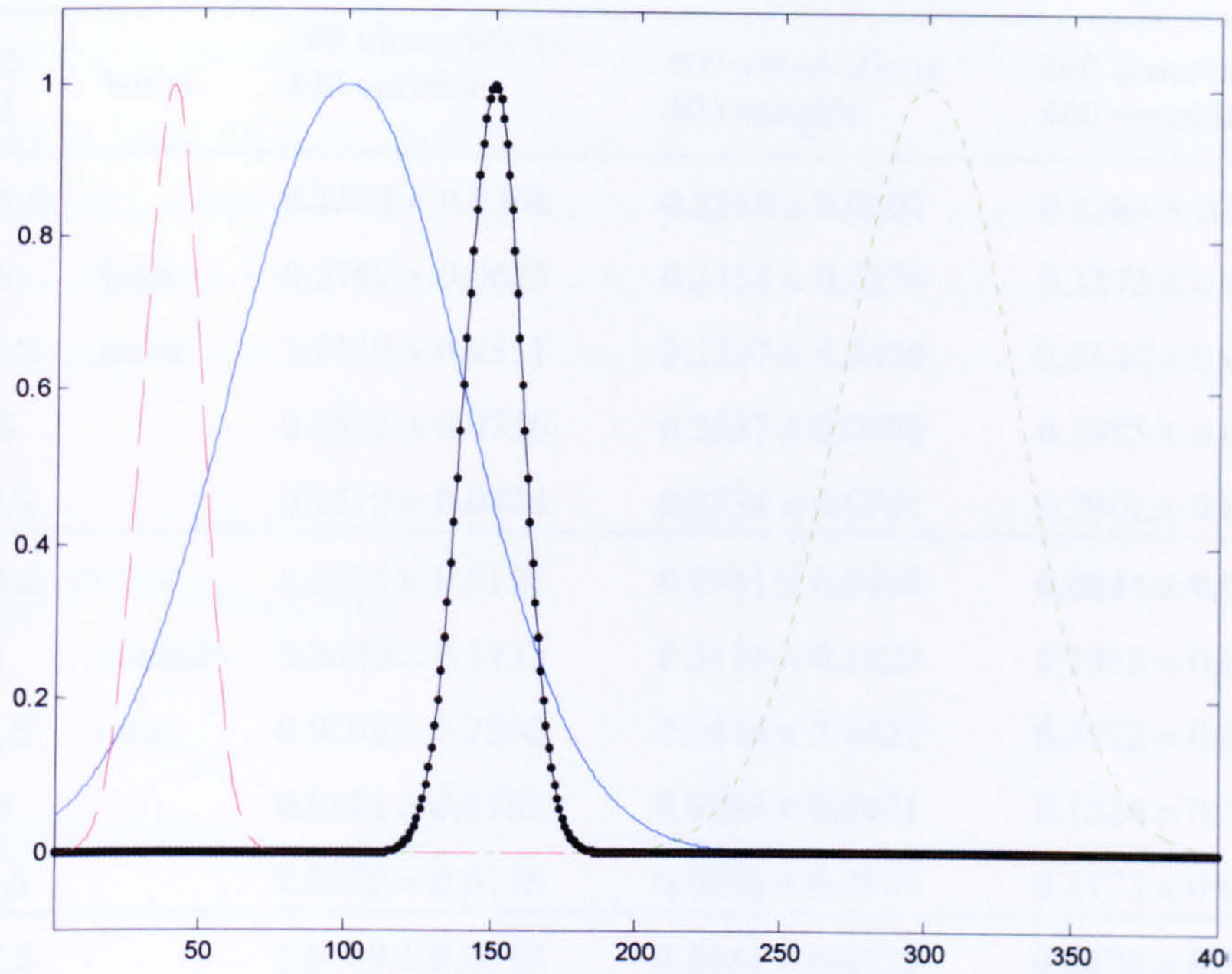

Figure A8. The four peaks used in the simulation

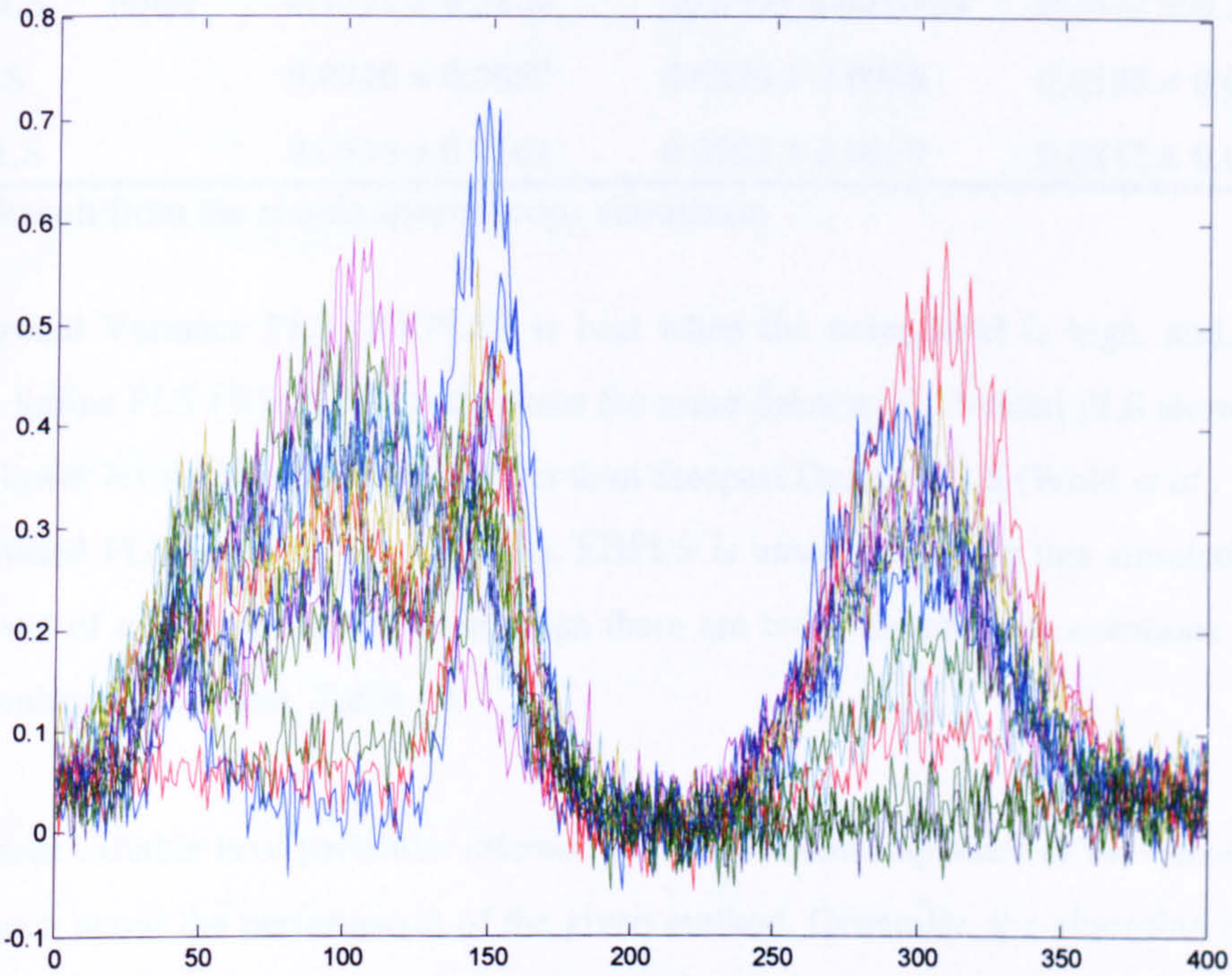

Figure A9. Typical simulation data after medium noise was applied.

The effect of the ratio of the number of observations to the number of variables and the impact of the noise level was investigated for the five approaches, RVPLS, SPLS, EBPLS, NPLS and SDPLS, in terms of the estimation of the weight vector. The results from the nine simulations are reported in Table 40 in terms of the Root Mean Squared Error of Prediction (RMSEP) calculated from an average of ten simulated models. The best RMSEP value for each simulation is identified in bold.

Method applied	Noise	100 observations 400 variable	400 observations 400 variable	800 observations 400 variable
A: RVPLS		**0.2303 ± 0.0368**	**0.2210 ± 0.0102**	**0.2264 ± 0.0122**
B: SPLS	high	0.3769 ± 0.0623	0.3155 ± 0.0278	0.3272 ± 0.0725
C: EBPLS	noise	1.0119 ± 0.9351	3.2237 ± 4.4650	0.5544 ± 0.0790
D: NPLS		0.2553 ± 0.0736	0.2837 ± 0.0902	0.3085 ± 0.0687
E: SDPLS		0.2519 ± 0.0424	0.2734 ± 0.0761	0.2902 ± 0.0704
A: RVPLS		0.0991 ± 0.0128	0.0941 ± 0.0065	**0.0944 ± 0.0040**
B: SPLS	medium	0.3040 ± 0.1432	0.3176 ± 0.1622	0.1915 ± 0.0097
C: EBPLS	noise	0.9102 ± 0.7506	2.2416 ± 2.4421	0.4772 ± 0.0377
D: NPLS		**0.0821 ± 0.0182**	**0.0766 ± 0.0071**	0.1524 ± 0.1192
E: SDPLS		0.0972 ± 0.0178	0.0858 ± 0.0103	0.1173 ± 0.0791
A: RVPLS		0.0449 ± 0.0059	0.0484 ± 0.0033	**0.0474 ± 0.0028**
B: SPLS	low	0.2939 ± 0.1643	0.2507 ± 0.1796	0.2009 ± 0.1560
C: EBPLS	noise	0.7221 ± 0.2864	20.7395 ± 60.5734	0.2372 ± 0.2212
D: NPLS		**0.0320 ± 0.0057**	**0.0336 ± 0.0016**	0.0596 ± 0.0826
E: SDPLS		0.0636 ± 0.0164	0.0522 ± 0.0029	0.0812 ± 0.0752

Table 40. Result from the simple spectroscopy simulation

The Reciprocal Variance PLS (RVPLS), is best when the noise level is high, and is always better than Spline PLS (Wold, 1992) that uses the same framework. Nested PLS shows the best results for lower levels of noise, and is better than Steepest Descent PLS (Wold *et al.*, 1989) and the Error Based PLS (Baffi *et al.*, 1999a,b). EBPLS is unsuccessful for this simulation due to the high level of multicollinearity, even when there are twice as many observations compared with the number of variables, Table 40.

The first latent variable is of particular interest. This is the most important of the latent variables and says much about the performance of the given method. Generally, the algorithm explaining most of the response variance for the first latent variable will give the best model. Therefore it is the first t-u plot that is shown for the five criteria, with the corresponding weight vector plotted on the right hand side. The discussion is mostly based on Figure A10, where the simulation is conducted by applying the same number of observations as variables and applying medium noise. Recalling that the calibration data is plotted as circles (o), while the validation data is plotted as crosses (+).

From Figure A10, it is clear that RVPLS (A), NPLS (D) and SDPLS (E) identify the nonlinear relationship better than the two other methods. In particular, EBPLS fails due to the

multicollinearity in the data set. The three methods, RVPLS (A), NPLS (D) and SDPLS (E) identify weight vectors that are similar. These are also the best models in terms of RMSEP. SDPLS is slightly worse than NPLS but shows less noisy characteristics. It is believed to be caused by the algorithm being more easily terminated at a local minimum closer to the starting vector. Increasing the maximum number of iterations does not improve the prediction ability of SDPLS.

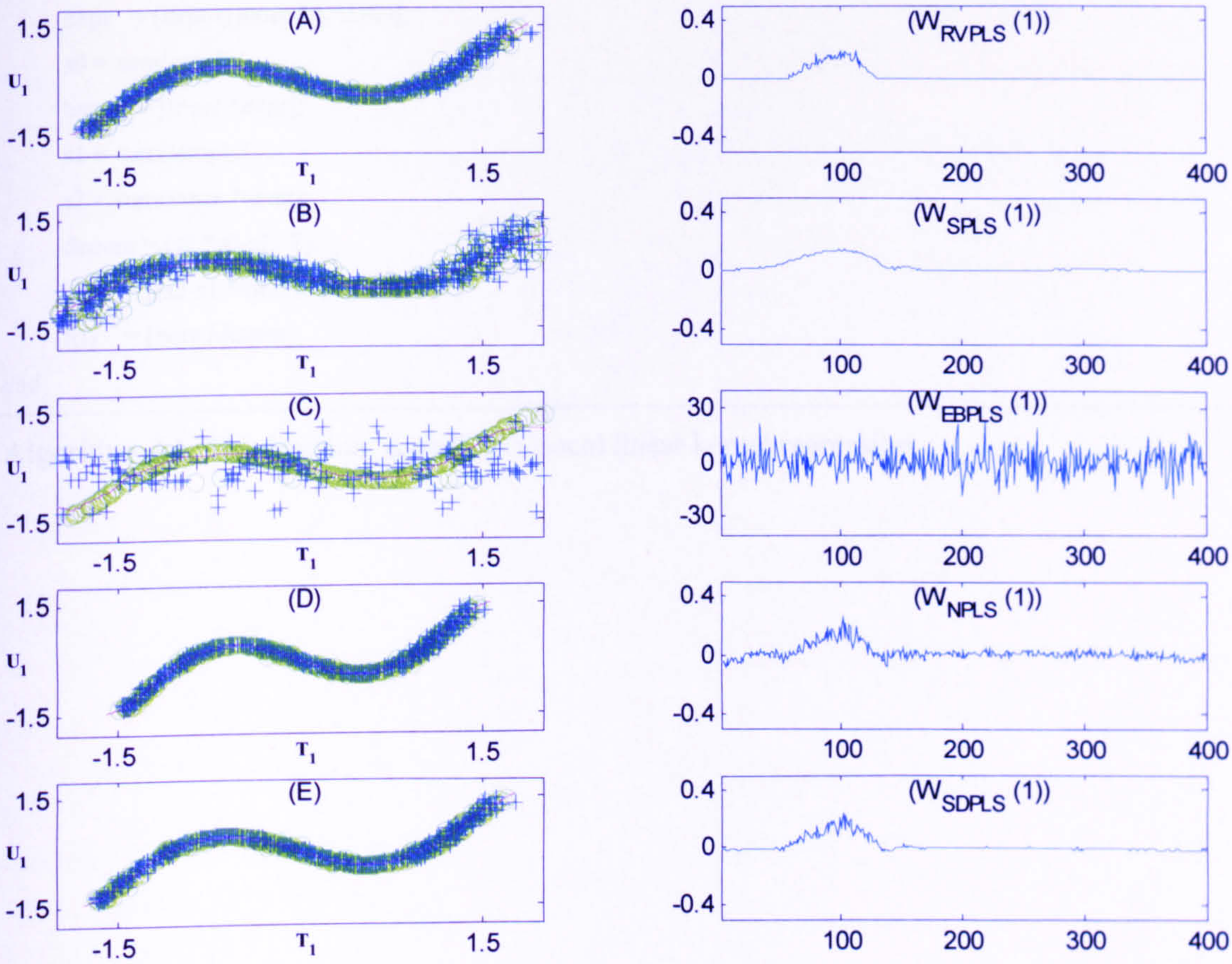

Figure A10. T_1-U_1 plot and plot of corresponding the weight vector for simulation data.

For low signal to noise, RVPLS appears to be the best choice for this particular simulation. In this case, SDPLS does better than NPLS. RVPLS shows increased performance with increasing number of observations. EBPLS does worst when the number of observations matches the number of variables, probably due to the construction of the pseudoinverse (using the standard MATLAB function *pinv*). For low or medium noise, NPLS appears to be the best choice.

A2. MATLAB functions

A2.1. Local Linear Kernel Regression I

```
function  f = KernReg1(z,x,y,h)
siz = length(z);
h2  = mean(h).^2;
for j = 1:siz,
        inner  = [x-z(j)];
        expr   = [exp(-((inner./h).^2)/2)];
        s0 = sum(expr);
        temp   = [inner.*expr];
        s1 = sum(temp);
        s2 = sum(inner.*temp);
        denom = (s2.*s0-s1.^2);
        nom  = [(s2-s1.*inner).*expr];
        f(j)    = [nom./denom];
end
```

Algorithm A1. The function, KernReg1: Local linear kernel regression.

A2.2. Local Linear Kernel Regression Bandwidth (*loess*)

```
function h  = hvariable(x, y);
x = x – mean(x); y = y - mean(y); n = length(x);
sgmh = (median(abs(x-median(x)))/0.6745);
h  = ((4/(3*n))^0.2)*sgmh;
sgmreg = sigmaK(x,y);
[temp1, mder1] = Kreg(x, [x, y], h);
mder2 = gradient(mder1, x);
a = find(isnan(mder2)==1);
mder2(a) = 0; mder2(a) = mean(mder2); temp2 = mean(mder2.^2);
hopt = [sgmreg^2/(2*sqrt(pi)*n*temp2)]^0.2;
% h_opt the optimal (or plug-in) bandwidth of Bowmann and Azzalini (1997)
kth = min([floor(n/10) 5]);
for hloop = 1:n,
   if hloop-kth > 0
      ts1=abs(x(hloop,1)-x(hloop-kth,1));
   else
     ts1=1e+10;
   end
   if  hloop + kth <= n
     ts2 = abs(x(hloop,1)-x(hloop+kth,1));
   else
     ts2 = 1e+10;
 end
    hdistance(hloop,1) = min(ts1, ts2);
end
hdistance = hdistance/norm(hdistance);
temp3 = exp(mean(log(hdistance)));
h = hopt*hdistance/temp3;
```

Algorithm A2. The function, hvariable: The *loess* bandwidth.

```
function sig = sigmaK(x, y)
x  = x(:)'; y  = y(:)'; n  = length(x);
[xx xxs] =  sort(x); yy     = y(xxs);
xx1 =  abs(xx(2:length(xx))-xx(1:(length(xx)-1)));
xx2  =  abs(xx(3:length(xx))-xx(1:(length(xx)-2)));
a   = xx1(2:length(xx1))./xx2;
b   = xx1(1:(length(xx1)-1))./xx2;
a(xx2 == 0) = 0.5;
b(xx2 == 0) = 0.5;
cc  = sqrt(a.^2 + b.^2 + 1);
eps = yy(1:(n-2)).* a./cc + yy(3:n).* b./cc - yy(2:(n-1))./cc;
sig = sqrt(sum(eps.^2)/(n-2));
```

Algorithm A3. The function, sigmaK: Used in the hvariable function, Algorithm A2.

A2.3. Local Linear Kernel Regression II

```
function  [f, dfdt] = KernReg(z,x,y,h)

siz = length(z);

h2  = mean(h).^2;

for j = 1:siz,

      inner  = [x-z(j)];

      expr   = [exp(-((inner./h).^2)/2)];

      s0 = sum(expr);

      temp   = [inner.*expr];

      s1 = sum(temp);

      s2 = sum(inner.*temp);

      s3     = sum(inner.^2.*temp);

      denom = (s2.*s0-s1.^2);

      nom  = [(s2-s1.*inner).*expr];

      f(j)   = [nom./denom];

      r2     = r1'*y;

      temp   = s2/h2 - s0;

      temp2  = s3/h2 - s1;

      e1     = [((temp2 – temp.*inner).*expr)/denom];

      e2     = [(inner/h2).*r1];

      e3     = [(((temp2 - s1)*s0 + s2*s1/h2 - 2*s1*temp)/(denom^2)).*(nom)];

      dfdt(j) = ([e1+e2-e3]'*[y]) - r2*(sum(e1)+sum(e2)-sum(e3));

end
```

Algorithm A4. The function, KernReg2: Local linear kernel regression and 1st derivative.

A2.4. Cubic Smoothing Splines

```
function [pp,y] = CSplines(x,y,p)
x = x(:); y = y(:); m = length(x);
[x,ind] = sort(x); y = y(ind);
dx = diff(x); dydx = diff(y)./dx;
R = spdiags([dx(2:m-1), 2*(dx(2:m-1)+dx(1:m-2)), dx(1:m-2)],[-1 0 1], m-2,m-2);
odx = ones(m-1,1)./dx;
Qt = spdiags([odx(1:m-2), -(odx(2:m-1)+odx(1:m-2)), odx(2:m-1)], [0 1 2], m-2,m);
u = (6*(1-p)*Qt*Qt'+p*R)\diff(dydx);
y = y - (6*(1-p))*diff([0 ; diff([0;u;0])./dx ; 0]);
c3 = [0;p*u;0];
c2 = diff(y)./dx-dx.*(2*c3(1:m-1,:)+c3(2:m,:));
coefs = reshape([(diff(c3)./dx)',3*c3(1:m-1,:)',c2',y(1:m-1,:)'],(m-1),4);
[l,k] = size(coefs);
breaks = reshape(x',1,l+1);
pp.form = 'pp';
pp.breaks = x';
pp.coefs = coefs;
pp.pieces = l;
pp.order = k;
pp.dim = 1;
```

Algorithm A5. The function, CSplines: Cubic smoothing spline.

A2.5. The General Nonlinear PLS Algorithm

```
function [W, Q, T, U, P, Û] =NLPLS(X,Y,lv)   % The general nonlinear PLS algorithm
  for j = 1 : lv,                            % Repeat until lv latent variables are found
     u = Y(:,1);                             % The Y-score u is selected to be the first variable
     w = cor(X,u).^9;                        % First estimate the weight vector (starting vector)
     w = w/norm(w);                          % Normalize w to unit length
     t = X*w;                                % The corresponding X-score vector t
     term = 0; best = 1e10; count = 0;stop = 0;  % Initializing termination criteria
    while  term==0,                          % Repeat until convergence, i.e. term ≠ 0
      count  = count + 1;                    % Count the no. of iterations per latent variable (for termination)
(i)  [û, dûdt] = f(t, [t u]);                % Fit a nonlinear function between t and u, estimate the derivative
(ii) q = Y'*û;                               % Estimate the loading vector q applying û
     q = q/norm(q);                          % Normalize q to unit length
     u = Y*q;                                % Estimate the next u vector using the linear combination q
     error = u'*r;                           % Calculate the squared error
     if  best > error;                       % Check if the error is the smallest yet
         best = error;                       % Retain the smallest error value
         wb = b; qb = q; ûb = û;             % Retain the best model
         stop = 0;                           % Initialize the third termination criterion counter
     else                                    % If not an improved model is obtained
         stop  = stop + 1;                   % Increase the stop by one
     end                                     % Two sequential worse models terminates the inner loop
(iii) w = fw(X, w, Y, q, û, dûdt, method);   % Estimate the weight vector by the different methods.
     told = t;                               % Keep the old score vector t to control the convergence
     t = X*w;                                % Calculate the next X score vector
     con = norm(t-told)/norm(t);             % Convergence if the successive relative score is less than 1e-8
     if (count > 25 | stop == 2 | con < 1e-8),  % , or if the number of iteration is over 25
        term = 1;                            % , or if 2 successive iterations do not improve the model
     end                                     % End if (convergence criteria)
    end                                      % End while loop (convergence)
    t = X*wb;                                % Calculate the score vector t for X
    u = Y*qb;                                % Calculate the score vector u for Y
    p = X'*t; /t'*t;                         % Calculate the loading vector p for X
    X = X – t*p';                            % Rank one reduction of the X matrix
    Y = Y – ûb *qb';                         % Rank one reduction of the Y matrix
    T(:, j) = t;                             % Store the X scores t
    P(:, j) = p;                             % Store the X loadings p
    W(:, j) = wb;                            % Store the X weights w
    U(:, j) = u;                             % Store the Y scores u
    Q(:, j) = qb;                            % Store the Y loadings q
    Û(:, j) = ûb;                            % Store the nonlinear mapping û
end                                          % End outer loop (latent variables)
```

Algorithm A6. The function, NLPLS: The general nonlinear PLS algorithm.

```
function w = fw(X, w, t, u, û, dûdt, method);

[xûow xcol]=size(X);

if  method==1,    % RVPLS

    level = 1/([median(abs(u-median(u)))].^2);

    for k = 1:xcol,

        s = sign(t'*X(:,k));

        temp = s.*X(:,k);
```

```
    temp=[temp].*[std(t(:))/std(temp)] -mean(temp)+mean(t);
    û  = f(temp, t, u);
    e = u-û;
    w(k) = s/([median(abs(e-median(e)))].^2);
end
[a b]  = find(abs(w)<level);
w(a)   = 0;
[a b]  = find(abs(w)>=0);
    w(a)   = sign(w(a)).*(abs(w(a))-level);
    w =  w./norm(w);
elseif  method ==2,    % SPLS
    for k = 1:xcol,
        s = sign(t'*X(:,k));
        temp = s.*X(:,k);
        temp=[temp].*[std(t(:))/std(temp)]-mean(temp)+mean(t);
        û  = f(temp, t, u);
       w(k) = s.*cor(u,û).*std(X(:,k));
    end
     w =  w./norm(w);
elseif  method == 3,    % EBPLS
    error = u - û;
    Z = dûdt (:,ones(1,xcol)).*X;
    dw = pinv(Z'*Z)*Z'*error;
    w = w + dw;
    w =  w./norm(w);
elseif  method = = 4,    % NPLS
    error = u - û;
    Z = dûdt (:,ones(1,xcol)).*X;
    dw = PLScv(Z, error, rank(Z));
    w = w + dw;
    w =  w./norm(w);
elseif  method = = 5,    % SDPLS
    error = u - û;
    Z = dûdt (:,ones(1,xcol)).*X;
    dw = PLS(Z, error, 1);
    w = w + dw;
```

```
    w =  w./norm(w);
else
    error('The method does not exist');
end
```

Algorithm A7. The function, fw: Calculation of the weight vector per method.

```
function [û, dûdt]  = f(temp, t, u);
    [x, tn]  = sort(t);
    y = u(tn);
    h = hVariable(x, y);
    [û dûdt] = Kreg(temp,[x y], h);
end
```

Algorithm A8. The function, f: The nonlinear function and its derivative

REFERENCES

Albers, W., (1978), Testing the mean of a normal population inter dependence, *The Annals of Statistics,* **6**(6), 1337-1344.

Allen, D. M., (1974), The relationship between variable selection and data augmentation and a method of prediction, *Technometrics*, **16**, 125-127.

Almøy, T., (1996), A simulation study on comparison of prediction methods when only a few components are relevant, *Computational Statistics and Data Analysis*, **25**, 377-398.

Baffi, G., Martin, E. B. and Morris, A. J., (1999a), Non-linear projection to latent structures revisited (the quadratic PLS algorithm), *Computers and Chemical Engineering,* **23**, 395-411.

Baffi, G., Martin, E. B. and Morris, A. J., (1999b), Non-linear projection to latent structures revisited (the neural network PLS algorithm), *Computers and Chemical Engineering,* **23**, 1293-1307.

Baffi, G., Martin, E. B. and Morris, A. J., (2000), Non-linear dynamic projection to latent structures modelling, *Chemometrics and Intelligent Laboratory Systems,* **52**(1), 5-22.

Baylis, W. E. and Pradhan; A. D., (1984), Locating minima of a hypersurface: A modification of the Levenberg-Marquardt method, *Computer Physics Communications,* **31**(4), 297-301.

Banks, D., (1993), Is industrial statistics out of control?, *Statistical Science,* **8**(4), 356-409.

Bates, D. M. and Watts, D. G., (1980), Relative curvature measures of nonlinearity, *Journal of the Royal Statistical Society*, **42**, 1-25.

Bishop, C. M., (1995), Neural Networks for Pattern Recognition, Oxford University Press, New York

Bjorkstrom, A. and Sundberg, R., (1999), A generalised view on continuum regression, *Scandinavian Journal of Statistics,* **26**, 17-30.

Blanco M., Coello J., Iturriaga H., Maspoch S. and Pagès L., (2000), Calibration in non-linear near infrared reflectance spectroscopy, *Chemometrics and Intelligent Laboratory Systems*, **50**, 75-80.

Bock H. G., (1981), Numerical treatment of inverse problems in chemical reaction kinetics, Modelling of Chemical Reaction Systems, Series in Chemical Physics. Editors: Ebert K.H., Deuflhard P. and Jäger W., Springer, Heidelberg, **18**, 102-125.

Borggaard, C . and Thodberg, H. H. (1992) Optimal minimal interpretation of spectra, *Analytical Chemistry*, **64**, 545-551.

Bowman A. W, Azzalini A., (1997), Applied Smoothing Techniques for Data Analysis. Clarendon: Oxford, UK.

Breiman, L., Friedman J. H., Olshen R., and Stone C. J., (1984), Classification and regression trees, Wadsworth International Group, Belmont, California.

Butler, N. A. and Denham, M. C., (2000), The peculiar shrinkage properties of partial least squares, *Journal of the Royal Statistical Society, B (Methodology)*, **62**(3), 585-593.

Cao, R., Cuevas, A., and Gonzáles-Manteiga, W., (1994), A comparative study of several smoothing methods in density estimation, *Computational Statistics and Data Analysis*, **17**, 153-176.

Cattell, R. B., (1966), The scree test for the number of factors, *Multivariate Behavioural Research*, **1**, 245-267.

Clarke, B. A. and Disney, R. L., (1970), Probability and Random Processes, John Wiley & Sons, New York.

De Boor, C., (1978), A Practical Guide to Splines, Springer-Verlag, New York

De Jong, S., (1993a), SIMPLS: An alternative approach to partial least squares regression, *Chemometrics and Intelligent Laboratory Systems* **18**, 251-263.

De Jong, S., (1993b) PLS fits closer than PCR, *Journal of Chemometrics*, **7**(6), 551-557.

De Jong, S., (1995), PLS shrinks, *Journal of Chemometrics*, **9**(4), 323-326.

De Jong, S., (1998), Regression coefficients in multilinear PLS, *Journal of Chemometrics*, **2**(1), 77-81.

De Jong, S., Wise, B. M. and Ricker, N. L., (2001), Canonical partial least squares and continuum power regression, *Journal of Chemometrics*, **15**(2), 85-100.

De Maesschalck, R., Jouan-Rimbaund, D. and Massart, D. L., (2000), The Mahalanobis distance: Tutorial, *Chemometrics and Intelligent Laboratory Systems* **50**, 1-18.

De Vena L., Mastretta M. and Ricciardello L., (1995), Neural network architectures for industrial applications, *Biosensors and Bioelectronics*, **10**(1-2), 231-236.

Dey, A. K., Ruymgaart, F. H. and Mair, B. A., (1996), Cross-validation for parameter selection in inverse estimation problems, *Scandinavian Journal of Statistics*, **23**, 609-620.

Dobson, A. J., (1997), Introduction to Generalised Linear Models, Chapman and Hall, Padstow, Cornwall.

Dong, D. and McAvoy, T. J., (1996), Nonlinear principal component analysis — based on principal curves and neural networks, *Computers and Chemical Engineering*, **20**, 65-78.

Draper, N. H. and Smith, H., (1998), Applied Regression Analysis, John Wiley & Sons, New York.

Dunia, R., Quin, J. S., Edgar, T. F. and McAvoy, T. J., (1996), Use of principal component analysis for sensor fault identification, *Computers and Chemical Engineering*, **20**, S713-S718.

Fearn T., (2000), On orthogonal signal correction, *Chemometrics and Intelligent Laboratory Systems*, **50**, 47-52.

Ferre, L., (1995), Selection of components in principal component analysis : A comparison of methods, *Computational Statistics and Data Analysis*, **19**, 669-682.

Frank, I. E. and Friedman, J. H., (1993), A statistical view of some chemometrics regression tools, *Technometrics*, **35**(2), 109-148.

Friedman, J. E. and Tukey J. W., (1974), A projection pursuit algorithm for exploratory data analysis. *IEEE Trans. Comp.*, **23**(9), 881-889.

Friedman, J. E., and Stuetzle W., (1981), Projection pursuit regression, *Journal of the American Statistical Association*, **76**, 817-823.

Friedman, J. E., (1985), Classification and multiple regression through projection pursuit, (Department of Statistics, Stanford University, Report LCS 12).

Friedman, J. E., (1991), Multivariate adaptive regression splines, *Annals of Statistics*, **19**, 1-141.

Friedman, J. E., (1997), Data mining and statistics: What's the connection?, *http://www-stat.stanford.edu/~jhf/#Reports* (2002).

Gallagher, N. B., Wise, B. M., Butler, S. W., White, D. D., Jr. and Barna, G. G., (1997), Development and benchmarking of multivariate statistical process control tools for a semiconductor etch process : Improving robustness through model updating. *IFAC ADCHEM '97*, Banff, Canada, 78-83.

Geladi, P. and Kowalski, B., (1986), Partial least squares regression : A tutorial, *Analytica Chimica Acta*, **185**, 1-17.

Geladi, P. and Kowalski, B. R., (1986), An example of 2-block predictive partial least-squares regression with simulated data, *Analytica Chimica Acta*, **185**, 19-32.

Geladi, P., (1988), Notes on the history and nature of partial least squares (PLS) modelling, *Journal of Chemometrics*, **2**, 231-246.

Geladi, P. and Esbensen K., (1990), The start and early history of chemometrics, selected interviews, part 1 and 2, *Journal of Chemometrics*, **4**, 337-354, 389-399.

Gemperline, P. J., (1992), Developments in nonlinear multivariate calibration, *Chemometrics and Intelligent Laboratory Systems*, **15**(2-3), 115-126.

Goutis, C., (1996), Partial least squares algorithm yields shrinkage estimators, *The Annals of Statistics*, **24**(2), 816-824.

Hassel P. A., Martin E. B. and Morris A. J., (2002), Non-linear partial least squares. Estimation of the weight vector, *Journal of Chemometrics*, **16**, 419-426.

Hastie, T. J. and Tibshirani, R. J., (1990), Generalized Additive Models, Chapman and Hall, London.

Helland, I. S., (1988), On the structure of partial least squares regression, *Communications in Statistics - Simulations*, **17**(2), 581-607.

Henry, R. C., Park, E. S. and Spiegelman, C. H., (1999), Comparing a new algorithm with the classic methods for estimating the number of factors, *Chemometrics and Intelligent Laboratory Systems*, (**48**), 91-97.

Hestenes, M. R. and Stiefel, E. (1952), Methods of conjugate gradients for solving linear systems, *Journal of Research of the National Bureau of Standards*, **49**, 409-436.

Hill, R. C., Formby, T. B., and Johnson, S. R. (1977), Component selection norms for principal component regression, *Communications in Statistics - Theory and Methods*, **6**, 309-334.

Hoaglin D.C., Mosteller F, Tukey JW., (1983), Understanding Robust and Exploratory Data Analysis, Wiley, New York.

Hoerl, A. E., (1962), Application of ridge analysis to regression problems, *Chemical Engineering Progress*, **58**, 54-59.

Hoerl A. E., Kennard R. W. and Baldwin K., (1975), Ridge regression: Some simulations, , *Communications in Statistics - Theory and Methods*, **4**, 105-123.

Hoerl, A. E. and Kennard, R. W., (1970), Ridge regression: Biased estimation for nonorthogonal problems, *Technometrics,* **42**(1), 80-86.

Hotelling, H., (1931), The generalisation of student's ratio, *Annals of Statistics,* **2**(3), 360-378.

Hotelling, H., (1933), Analysis of a complex of statistical variables into principal components, *The Journal of Educational Psychology,* **24**, 417-441, 498-520.

Hoyle, R. H. (Ed.) (1995). Structural Equation Modelling: Concepts, Issues, and Applications. Sage Publications, Thousand Oaks.

Höskuldsson, A., (1988), PLS regression methods, *Journal of Chemometrics,* **2**, 211-228.

Höskuldsson, A., (1992), Quadratic PLS regression, *Journal of Chemometrics,* **6**, 307-334.

Höskuldsson, A., (1996), Prediction Methods in Science and Technology: Vol 1. Basic Theory. Thor Publishing, Denmark.

Höskuldsson, A., (1992), The Heisenberg modelling procedure and application to nonlinear modelling, *Chemometrics and Intelligent Laboratory Systems*, **44**, 15-30.

Jackson, D. A., (1993), Stopping rules in principal components analysis: A comparison of heuristical and statistical approaches, *Ecology,* **74**(8), 2204-2214.

Jackson, J. E., (1985), Multivariate quality control, *Communications in Statistics - Theory and Methods,* **14**(11), 2657-2688.

Jackson, J. E., (1991), A User's Guide to Principal Components, Wiley, New York.

Jackson, J. E. and Hearne, F. T., (1979), Hotelling's T^2 for principal components - what about absolute values?, *Technometrics,* **21**(2), 253-255.

Jackson, J. E. and Mudholkar, G. S., (1979), Control procedures for residuals associated with principal components analysis, *Technometrics,* **21**(3), 341-349.

Jia, F., Martin E. B. and Morris, A. J. (1998), Non-linear Principal Components Analysis for Process Fault Detection, *Computers & Chemical Engineering*, **22**(1), S851-S854.

Johnson R. A. and Wichern D. W., (1998) Applied Multivariate Statistical Analysis. Prentice-Hall, New Jersey.

Jolliffe, I. T., (1986), Principal Component Analysis. Springer-Verlag, New York.

Jonathan, P., Krzanowski, W. J. and McCarthy, W. V., (2000), On the use of cross-validation to assess performance in multivariate prediction, *Statistics and Computing*, **10**, 209-229.

Joshi, V. N. and Tewarson, R. P., (1972) On solving ill-conditioned systems of linear equations, *Transactions of the New York Academy of Science*, **2**(34), 565-571.

Kaspar, M. H. and Ray, W. H., (1993), Dynamic PLS modelling for process control, *Chemical Engineering Science,* **48**(20), 3447-3461.

Kaspar, M. H. and Ray, W. H., (1993), Partial least squares modelling as successive singular value decompositions, *Computers and Chemical Engineering,* **17**(10), 985-989.

Kourti, T., Nomikos, P. and MacGregor, J. F., (1995), Analysis, monitoring and fault diagnosis of batch processes using multi-block and multi-Way PLS, *Journal of Process Control,* **5**(4), 277-284.

Kresta, J. V., Marlin, T. E. and MacGregor, J. F., (1994), Development of inferential process models using PLS, *Computers and Chemical Engineering,* **18**(7), 597-611.

Krzanowski, W. J., (1987), Cross-validation in principal component analysis, *Biometrics,* **43**, 575-584.

Kurtanjek, Z., (1995), Bioreactor modelling and control by principal component based neural networks, *Computer Applications in Biotechnology*, 6th. May 14-17, Germany.

Kvalheim O. M., and Karstang, T. V., (1989), Interpretation of latent-variable regression models, *Chemometrics and Intelligent Laboratory Systems,* **7**, 39-51.

Lanczos, C., (1952), Solution of systems of linear equations by minimized iterations. *Journal of Research of the National Bureau of Standards*, **45**, 255-282.

Lennox B., Montaque G. M., Frith A.. M., Gent C. and Bevan V., (2001), Industrial application of neural networks - an investigation. *Journal of Process Control*, **11**(5), 497-507.

Lewis, R. M., Torczon V. and Trosset M. W., (2000), Direct search methods: then and now, *Journal of Computational and Applied Mathematics*, **124**(1-2), 191-207.

Li, B., Hassel P. A., Martin E. B. and Morris A. J., (2001), Nested PLS, *PLS and related methods*, Proceedings of the PLS 01' International Symposium, Ed. Vinzi, V., Morineau, L. M. and Tennenhaus, M., CISIA-CERESTA, Montreal, France.

Lindgren, F., Geladi , P. and Wold, S., (1993), The kernel algorithm for PLS, *Journal of Chemometrics*, **7**, 45-49.

Lingjaerde, O. C. and Christophersen, N., (2000), Shrinkage structure of partial least squares, *Scandinavian Journal of Statistics*, **27**, 459-473.

Little, R. J. A. and Rubin, D. B., (1987), Statistical Analysis with Missing Data. Wiley, New York.

Lippmann, R. P., (1987), An introduction to computing with neural nets, *IEEE, Acoustic Speech Signal Process*, **4**, 4-22.

Ljung, L., (1987), System Identification : Theory for the User. Englewood, Ciffs, New Jersey.

Lorber, A., Wangen, L. E. and Kowalski, B. R., (1987), A theoretical foundation for the PLS algorithm, *Journal of Chemometrics*, **1**, 19-31.

MacGregor, J. F., (1988), On-line statistical process control, *Chemical Engineering Progress*, **84**(10), 21-31.

MacGregor J. F., Marlin T. E., Kresta J. V., and Skageberg B. (1991), Multivariate statistical methods in process analysis and control, *American Institute of Chemical Engineers Journal*, **67**, 17-99.

MacGregor, J. F., (1994), Statistical process control of multivariate processes. *IFAC, ADCHEM Advanced Control of Chemical Processes*, Kyoto, Japan.

MacGregor, J. F., Jaeckle, C., Kiparissides, C. and Koutoudi, M., (1994), Process monitoring and diagnosis by mulitblock PLS Methods, *American Institute of Chemical Engineers Journal,* 40(5), 826-838.

MacGregor, J. F. and Kourti, T., (1995), Statistical process control of multivariate processes, *Control Engineering Practice,* 3(3), 403-414.

Madansky, A., (1988), Prescriptions for Working Statisticians. R.R Donnelley & Sons, Harrisonburg, Virginia.

Manne, R., (1987), Analysis of two PLS algorithms for multivariate calibration, *Chemometrics and Intelligent Laboratory Systems,* 2, 187-197.

Mardia, K. V., Kent, J. T., and Bibby, J. M., (1979), Multivariate Analysis. Academic Press, London.

Martens, H. and Næs, T., (1989), Multivariate Calibration, Chichester, England.

Martens, H., and Dardenne, P., (1998), Validation and verification of regression in small data sets. *Chemometrics and Intelligent Laboratory Systems,* 44, 99-121.

Mejdel, T. and Skogestad, S., (1991), Estimation of distillation compositions from multiple temperatures measurements using partial-least-squares regression, *Industrial Engineering Chemical Research,* 30, 2543-2555.

Miller, P., Swanson, R. E. and Heckler, C. E., (1998), Contribution plots: A missing link in multivariate quality control, *Applied Mathematics and Computer Science,* 8(4), 775-792.

Montgomery, D. C., (1991), Introduction to Statistical Quality Control. John Wiley & Sons, Singapore.

Montgomery, D. C. and Mastrangelo, C. M., (1991), Some statistical process control methods for autocorrelated data, *Journal of Quality Technology*, **23**(3), 179-204.

Morris, P., (1993), The breakout method for escaping from local method, *In proceedings of the 11th National Conference on Artificial Intelligence, AAAI-93*, 40-45. The MIT press.

Negiz, A. and Cinar, A., (1997), Statistical monitoring of multivariable dynamic processes with state-space models, *American Institute of Chemical Engineers Journal*, **43**(8), 2002-2020.

Nelder J. A. and Mead R., (1965), A simplex method for function minimization, *Computer Journal*, 7, 308-313.

Noble, B. (1969), Applied Linear Algebra, Prentice-Hall, Englewood Cliffs, NJ.

Næs, T., Isaksson, T., Fearn T. and Davies T., (2002), A User-friendly Guide to Multivariate Calibration and Classification. NIR Publications, Chichester.

Næs, T., Kvaal. K., Isaksson T. and Miller C. (1993), Artificial neural networks in multivariate calibration. *Journal of Near Infrared Spectroscopy*, **1**, 1-11.

Næs, T. and Isaksson T. (1995), Adjusting for non-linearities in calibration using principal components, *NIR news*, **6**(4), 4-5.

Oja E., (1989), Neural networks, principal components, and subspaces. *International Journal of Neural Systems*, **1**, 61-68.

Osborne B. G. and Fearn T., (1988), Near Infrared Spectroscopy in Food Analysis, Wiley, New York.

Pearson, K., (1901), On lines and planes of closest fit to systems of points in space, *Phylosophical Magazine*, **2**, 559-572.

Phatak, A., Penlidis A. and Reilly P., (1992), The geometry of 2-block partial least squares regression, *Communications in Statistics - Theory and Methods*, **21**, 1517-1533.

Phatak, A. and De Jong, S., (1997), The geometry of partial-least-squares, *Journal of Chemometrics*, **11**, 311-338.

Picard, R. R., and Cook, R. D., (1984), Cross validation of regression models, *Journal of the American Statistical Association*, **87**(418), 575-583.

Rivals, I., and Personnaz, L., (1999), On cross-validation for model selection, *Neural Computation*, **11**, 871-901.

Rocke, D. M. and Wooddruff, D. L., (1996), Identification of outliers in multivariate data, *Journal of the American Statistical Association*, **91**(435), 1047-1061.

Roels, J. A., (1982), Mathematical models and the design of biochemical reactors, *Journal of Chemical Technology and Biotechnology*, **32**, 59-72.

Rosenbrock, H. H. (1960), An automatic method for finding the greatest or least value of a function, *Computer Journal*, **3**, 175-185.

Runger, G. C. and Alt, F. B., (1996), Choosing principal components for multivariate statistical process control, *Communications in Statistics - Theory and Methods*, **25**(5), 909-922.

Seber, G. A. F., (1977), Linear Regression Analysis. John Wiley & Sons, New York.

Sekulic, S., Seasholtz, M. B., Wang, Z., Kowalski, B. R., Lee, S. E. and Holt B. R. (1993), Nonlinear multivariate calibration methods in analytical chemistry, *Analytical Chemistry*, **65**(19), A835-845.

Shao, R., Jia, F., Martin, E. B. and Morris, A. J., (1999), Wavelets and non-linear principal components analysis for process monitoring, *Control Engineering Practice*, 7(7), 865-879.

Simonoff J. S., (1996), Smoothing Methods in Statistics. Springer: New York, USA.

Stoika, P. and Soderstrom, T., (1998), Partial least squares : A first order analysis, *Scandinavian Journal of Statistics*, **25**, 17-24.

Stone, M., (1974), Cross-validation choice and assessment of statistical predictions, *Journal of the Royal Statistical Society*, B, **36**, 111-113.

Stone, M., (1977a), An asymptotic equivalence of choice of model by cross validation and Akaike's criterion, *Journal of the Royal Statistical Society*, B, **39**, 44-47.

Stone, M., (1977b), Asymptotic for and against cross-validation, *Biometrika*, **64**, 29-35.

Stone, M. and Brooks, R. J., (*1990*), *Continuum* regression : Cross-validated sequentially constructed prediction embracing ordinary least squares, partial least squares and principal components regression, *Journal of the Royal Statistical Society*, B, **52**(2), 237-269.

Stone, M., (1974), Cross-validation choice and assessment of statistical predictions (with discussion). *Journal of the Royal Statistical Society*, B, **36**, 111-147.

Stordrange, L., Kvalheim, O. M. Hassel, P. A., Sørensen D. M. and Libnau, F. O., (2003), A comparison of techniques for modelling data with non-linear structure, *Journal of Near Infrared Spectroscopy*, **11**, 55 - 70.

Stuart, A. and Ord, J. K., (1987), Kendall's Advanced Theory of Statistics: Distribution Theory. Charles Griffin & Co, Belfast N. Ireland.

Sundberg, R., (1993), Continuum regression and ridge regression, *Journal of the Royal Statistical Society*, B, **55**, 653-659.

Sundberg, R., (1999), Multivariate calibration - Direct and indirect regression methodology, *Scandinavian Journal of Statistics*, **26**, 161-207.

Taylor J. R., (1997), Introduction to Error Analysis (2nd Ed.), University Science Books, Sausalito, California, USA.

Ter Braak, C. J. F. and De Jong, S., (1998), The objective function of partial least squares regression, *Journal of Chemometrics*, **12**(1), 41-54.

Thomas, C., Wada, T. and Seborg, D. E., (1996), Principal component analysis applied to process monitoring of an industrial distillation column. *IFAC, 13th Triennial World Congress*, San Francisco , USA, 61-65.

Tibshirani, R., (1987), Estimating transformations for regression via additivity and variance stabilization, *Journal of the American Statistical Association*, **83**, 394-405.

Tremblay, D. A. (1999), Using simulation technology to improve profitability in the polymer industry, *http://www.aspentech.com/ap/downloads/simulation_polymer.pdf* (2002).

Tucker, W. T., Faltin, F. W. and van der Wiel, S. A., (1993), Algorithmic statistical process control: An elaboration, *Technometrics,* **35**, 363-375.

Turner, P., Montaque, G. and Morris J., (1996), Dynamic neural networks in non-linear predictive control (an industrial application), *Computers and Chemical Engineering*, **20**, (972), S937-S942.

Ulrich, K (1986): <<http://scalab.uc3m.es/~ftp/uci_database/servo/servo.data>> (2001).

Van Huffel, S. and Vandevalle, J., (1991), The Total Least Squares Problem: Computational Aspects and Analysis, Frontiers in Applied Math, 9. Siam, Philadelphia, Pennsylvania.

Van der Wiel, S. A., Tucker, W. T., Faltin, F. W. and Doganaksoy, N., (1992), Algorithmic statistical process control: Concepts and an application, *Technometrics,* **34**(3), 286-297.

Vogt, N. B., (1989), Polynomial principal component regression: an approach to analysis and interpretation of complex mixture relationship in multivariate environmental data, *Chemometrics and Intelligent Laboratory Systems,* **7**, 119-130.

Wachs, A. and Lewin, D. R., (1999), Improved PCA methods for process disturbance and failure identification, *American Institute of Chemical Engineers Journal,* **45**(8), 1688-1700.

Wand M. P., Jones M. C., (1995), Kernel Smoothing, Chapman & Hall: London, UK.

Weisberg, S., (1985), Applied Linear Regression, John Wiley & Sons, New York.

West, M. and Harrison, J., (1997), Bayesian Forecasting and Dynamic Models, Springer-Verlag, New York.

Westerhuis, J. A., Gurden, S. P. and Smilde, A. K., (2000), Generalized contribution plots in multivariate statistical process monitoring, *Chemometrics and Intelligent Laboratory Systems*, **51**, 95-114.

Wetherill, G. B. and Brown, D. W., (1991), Statistical Process Control : Theory and Practice, Chapman and Hall, Great Britain.

Widrow, B. and Lehr M., (1990), A 30 years of adaptive neural networks: perception, madeline, and backpropagation, *Proceedings of IEEE 78*, **9**, 1415-1442 .

Wikstrom, C., Albano, C., Erikson, L., Friden, H., Johansson, E., Nordahl, A., Rannar, S., Sandberg, M., Kettaneh-Wold, N. and Wold, S., (1998), Multivariate process and quality monitoring applied to an electrolysis process, part I. Process supervision with multivariate control charts, *Chemometrics and Intelligent Laboratory Systems,* **42**, 221-231.

Wikstrom, C., Albano, C., Erikson, L., Friden, H., Johansson, E., Nordahl, A., Rannar, S., Sandberg, M., Kettaneh-Wold, N. and Wold, S., (1998), Multivariate process and quality monitoring applied to an electrolysis Process, part II. multivariate time-series analysis of lagged latent variables, *Chemometrics and Intelligent Laboratory Systems,* **42**, 233-240.

Wise, B. M. and Gallagher, N. B., (1997), Development and benchmarking of multivariate statistical process control tools for a semiconductor etch process: Impact of measurement selection and data treatment on sensitivity, *IFAC Conference SAFEPROCESSES'97,* Hull U.K., 35-42.

Wise, B. M., Veltcamp, D. J., Davis, B., Ricker, N. L. and Kowalski, B. R., (1988), Principal Components Analysis for Monitoring the West Valley Liquid Fed Ceramic Melter, *Waste Management Proceedings*, **2**, 811-818.

Wise B. M., Richet N. L., Veltkamp D. F., and Kowalski B. R. (1990), A theoretical basis for the use of principal component models for monitoring multivariate processes, P*rocess Control and Quality*, **1**(1), 41-51.

Wold, H., Editor, (1966a), Estimation of Principal Components and Related Models by Iterative Least Squares, *Multivariate Analysis*, Academic Press, New York.

Wold, H., Editor, (1966b), Nonlinear Iterative Partial Least Squares (NIPALS) Modelling: Some Current Developments, *Multivariate Analysis*, Academic Press, New York.

Wold, S., (1978), Cross-validatory estimation of the number of components in factor and principal components models, *Technometrics,* **20**, 397-405.

Wold, S., (1992), Nonlinear partial least squares modelling II. Spline inner relation, *Chemometrics and Intelligent Laboratory Systems,* **14**, 71-84.

Wold, S., Kettaneh, N. and Tjessem, K., (1996), Hierarchical multiblock PLS and PC models for easier model interpretation and as an alternative to variable selection, *Journal of Chemometrics,* **10**, 463-482.

Wold, S., Antii H,. Lindgren F. and Öhman J., (1998), Orthogonal signal correction of near-infrared spectra, *Chemometrics and Intelligent Laboratory Systems,* **44**, 175-185.

Wold, S., Kettaneh-Wold, N. and Skagerberg, B., (1989), Nonlinear PLS Modelling, *Chemometrics and Intelligent Laboratory Systems,* 7(1-2), 53-65.

Wold, S., Martens, H. and Wold, H., Editors, (1983), The Multivariate Calibration Problem in Chemistry Solved by the PLS Method, Proceedings Conference Matrix Pencils. Springer Verlag, Heidelberg.

Wu, W. and Manne, R., (2000), Fast regression methods in a Lanczos (or PLS-1) basis. Theory and applications, *Chemometrics and Intelligent Laboratory Systems*, **51**, 145-161.