8-2008

Advancing Data Analysis for Spectroscopic Imaging by Combining Wavelet Compression with Chemometrics

Robert Daryl Luttrell
University of Tennessee - Knoxville

Recommended Citation

Luttrell, Robert Daryl, "Advancing Data Analysis for Spectroscopic Imaging by Combining Wavelet Compression with Chemometrics. " PhD diss., University of Tennessee, 2008.
https://trace.tennessee.edu/utk_graddiss/467

To the Graduate Council:

I am submitting herewith a dissertation written by Robert Daryl Luttrell entitled "Advancing Data Analysis for Spectroscopic Imaging by Combining Wavelet Compression with Chemometrics." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Chemistry.

Frank Vogt, Major Professor

We have read this dissertation and recommend its acceptance:

Michael Sepaniak, Michael Best, Nicole Labbé

Accepted for the Council:

Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:

I am submitting herewith a dissertation written by Robert Daryl Luttrell, Jr. entitled "Advancing Data Analysis for Spectroscopic Imaging by Combining Wavelet Compression with Chemometrics." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Chemistry.

Dr. Frank Vogt, Major Professor

We have read this dissertation
and recommend its acceptance:

Dr. Michael Sepaniak

Dr. Michael Best

Dr. Nicole Labbé

Acceptance for the Council:

Carolyn R. Hodges

Vice Provost and Dean of the
Graduate School

(Original signatures are on file with official student records.)

Advancing Data Analysis for Spectroscopic Imaging by Combining Wavelet Compression with Chemometrics

A Dissertation
Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

Robert Daryl Luttrell, Jr.
August 2008

Abstract

Spectroscopic imaging is a vital tool for studying heterogeneous samples such as bacteria and tissue. Its ability to acquire spatially resolved information allows for identification and classification of the various constituents within a sample. Spectroscopic imagers quickly acquire thousands to tens of thousands of spectra per measurement. These data are often arranged in the form of a 3-dimensional (3D) data cube which contains two spatial dimensions and one spectral dimension. This large amount of data is beneficial for gaining a thorough understanding about the distributions of chemical information. If too little information is measured, important chemical behavior may be overlooked. Statistical analysis algorithms (chemometrics) are required to determine the relevant spectroscopic information within a data cube. Applying chemometrics to such large volumes of data presents computational difficulties regarding computer memory and processing speed. To overcome these burdens, wavelet transform compression is applied prior to chemometric evaluation to accelerate computations and reduce data storage requirements.

To optimize compression by enhancing acceleration and reducing approximation errors, different wavelets, or ‘hybrid wavelets’, can be applied to the different dimensions of a 3D data set. Determining which combination of wavelets will yield the most compression and best data representation is difficult since many possibilities exist. A compression method is presented that automatically determines the optimum wavelet combinations for a given data set. Principal component analysis (PCA) is used to demonstrate the capabilities of this new procedure, but the compression routine is advantageous for many chemometric techniques.

Although linear algorithms like PCA work well in many situations, they are not well-adapted for explaining nonlinear relationships. Kernel principal component analysis (KPCA) has recently been developed to overcome the limitations of linear algorithms. However, when applied to spectroscopic imaging, KPCA calculations require multiple gigabytes of RAM just for holding the data. Therefore, routine use of the algorithm is often prohibited on personal computers. To circumvent such situations, a wavelet compression algorithm is presented that avoids ever having to hold all data in memory at any point during the calculations. The goal is to enable the application of KPCA to large imaging data sets of heterogeneous samples.

Preface

The compression-based chemometric algorithms established in Chapters 2 - 4 are adapted from two first-author manuscripts published in the Journal of Chemometrics [1], [2] and a co-author manuscript published in the Journal of Chemical Education [3]. Appendix 1 is based on a project conducted in collaboration with the United States Department of Agriculture, Cotton Structure and Quality Research Unit. All alterations are implemented in order to reflect the appropriate style of this thesis.

Table of Contents

Chapter Page

Chapter 1 1

Introduction: Chemometrics and its Application to Spectroscopic Data Analysis 1

Chapter 2 6

Background and Theory of Chemometric Algorithms and Wavelet Transforms 6

2.1 Principles of Univariate Least Squares Regression 6

2.2 Classical Least-Squares (CLS) 11

2.2.1 CLS – Calibration 13

2.2.2 CLS – Evaluation of Unknown Samples 18

2.3 Principal Component Analysis / Principal Component Regression (PCA/PCR) 20

2.3.1 PCA – Calibration 22

2.3.2 PCR – Evaluation of Unknown Samples 24

2.4 Kernel Principal Component Analysis (KPCA) 25

2.4.1 KPCA – Calibration 26

2.4.2 KPCA – Evaluation of Unknown Samples 31

2.5 Wavelet Transforms (WTs) 32

Chapter 3 39

Composing Hybrid Wavelets for Enhanced Evaluation of N-dimensional Data Sets 39

3.1 Theory 40

3.2 Experimental 48

3.3 Results and Discussion 49

3.3.1 Finding a Representative Test Set Size 49

3.3.2 Impacts of Wavelet Combinations on Chemometric Models 51

3.4 Conclusions 61

Chapter 4 65

Accelerating Kernel Principal Component Analysis (KPCA): Applications to Spectroscopic Imaging 65

4.1 Kernel Principal Component Analysis (KPCA) Compression Method 66

4.1.1 Incorporating Data Compression into Calibration 66
4.1.2 Incorporating Mean-Centering into Wavelet Compressed KPCA 75
4.2 Experimental 79
4.2.1 “Bacteria Data” 79
4.2.2 “Remote Sensing Data” 79
4.2.3 Compression Algorithm Parameter Settings 80
4.3 Results and Discussion 80
4.3.1 Reducing Memory Requirements 81
4.3.2 Analysis of the Quality of the Compression Results 81
4.3.3 Decrease in Computation Time 86
4.4 Conclusions 87
Chapter 5 90
Summary and Conclusions 90
List of References 93
Appendix 1 98
Enhancing the Prediction of Cotton Micronaire Values from Near-infrared (NIR) Absorbance Spectra 98
A.1.1 Experimental Overview 98
A.1.2 Results without Baseline Correction 99
A.1.3 Results with Baseline Correction – Polynomial Fitting 101
A.1.4 Results with Baseline Correction – Second Derivatives 101
A.1.5 Results Utilizing Spectra Averaging 104
A.1.6 “A Closer Look” 108
A.1.7 Future Outlook 112
Appendix 2 114
Summary of Graduate School Honors, Publications, and Presentations 114
Vita 116

List of Figures

Figure Page

Figure 1: Example of a 3-dimensional spectroscopic data cube obtained with a 2-dimensional detector. 2

Figure 2: (a) A visible image of an *E. coli B* sample – no chemical information is available. 3

Figure 3: $K=5$ calibration samples have been used to determine a calibration curve by means of least-squares regression (simulated data). 9

Figure 4: Examples of various Daubechies family wavelets. 34

Figure 5: Cartoon illustrating the main steps of a WT (53). 36

Figure 6: (a) The original signal (or spectrum) in the original measurement domain (simulated data). 37

Figure 7: Flowchart outlining the major steps of the wavelet selection algorithm. 42

Figure 8: Flowchart outlining the compression of the Z-dimension; see (54) for definition of total information content and % retained information (*Q*%). 43

Figure 9: RGB score image of data set #2 (Parafilm® data cube) acquired by means of MIA [16]. 50

Figure 10: Histograms displaying the results of the wavelets chosen for each dimension of data set #1. 52

Figure 11: Acceleration factors (60) obtained for the two data sets versus the amount of retained information *Q*% (see (54)). 54

Figure 12: Average size (62) and variance (63) of compressed data set #1 versus accuracy/size importance. 56

Figure 13: Histograms of the wavelets chosen for each dimension of data set #2. 59

Figure 14: Residual images of data set #1; see text (refer to Fig. 3 (top left) in [36] and Fig. 2 in [60] for thermal images of the shown scene). 60

Figure 15: Plots comparing the first three principal components (PCs) for data set #1. .62

Figure 16: Plots comparing the first three principal components (PCs) for data set #2. .63

Figure 17: From a spectroscopic image cube (top; refer to Chapter 4.2.2) a Gram matrix $\mathbf{K}$ (47), (48) is derived (bottom, refer to Chapter 2.4). 68

Figure 18: The main steps of the compression algorithm are: (i) After determining which wavelet coefficients are to be preserved in the X-dimension, one row of the Gram matrix is built (47) at a time. 69

Figure 19: Memory requirements for storing the Gram matrix of data set #1 in the absence of compression and at five increasing compression levels (81).............82

Figure 20: Error distributions of the $j = 1 \ldots 4{,}096$ $T_{k=1}$-values (51) for data set #1 (Chapter 4.2.1) at five increasing compression levels (81). 84

Figure 21: Error distribution of the $j = 1 \ldots 4{,}096$ $T_{k=3}$-values (51) for data set #1 (Chapter 4.2.1) at five increasing compression levels (81). 85

Figure 22: Acceleration factors (84) for data set #1 (top) and data set #2 (bottom) at each of the five compression levels (81). 88

Figure 23: Predicted versus true micronaire values obtained from non-calibrated cotton spectra..100

Figure 24: (a) An example of a fourth order polynomial fitted to the first spectrum of *cot6*. 102

Figure 25: (a) The first three absorbance spectra from the *cot6* data set. 103

Figure 26: (top) Original vs. reconstructed spectrum 1 of *cot4*. 109

Figure 27: *cot4* score 1 values vs. micronaire. 111

Figure 28: Raman spectrum of a commercial cotton ball. 113

List of Notation and a Short Repetition of Linear Algebra

Any introductory linear algebra textbook [4], [5] should serve as a reference. In this brief section, some of the essential concepts that are required for basic chemometric algorithms are summarized.

x — scalar - upper case italics represent fixed values, i.e. J samples; lower case italics represent variables, i.e. the j^{th} sample or the j^{th} element x_j of a vector (see next definition).

$\mathbf{x}_{(N\times 1)}$ — a lower case bold letter indicates a (column) vector with N elements $\begin{pmatrix} x_1 \\ \vdots \\ x_N \end{pmatrix}$; the subscript is not always given.

$\mathbf{X}_{(N\times K)}$ — a capital bold letter indicates a matrix with N rows and K columns; if $N = K$ the matrix is called a square matrix; the subscript indicating the dimensions are only given where necessary.

$X_{n.k}$ — the matrix element of $\mathbf{X}$ at the position row number n and column number k.

Matrix multiplication: $\mathbf{X}_{(N\times K)} \cdot \mathbf{Y}_{(P\times Q)} = \mathbf{Z}_{(N\times Q)}$ is defined only if $K = P$. Note that usually $\mathbf{X}\cdot\mathbf{Y} \neq \mathbf{Y}\cdot\mathbf{X}$. The elements of $\mathbf{Z}$ are computed by:

$$Z_{n,q} = \sum_{k=1}^{K=P} X_{n,k} \cdot Y_{k,q}$$

$\mathbf{x}^{\mathbf{T}}$, $\mathbf{X}^{\mathbf{T}}$ — transposed vector (= row vector), transposed matrix; example:

$$\mathbf{X}_{(3\times 2)} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix} \rightarrow \mathbf{X}^{\mathbf{T}}_{(2\times 3)} = \begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

The rank of a matrix equals the number of linear independent rows (row vectors) or columns (column vectors).

$\mathbf{y}$ is linearly dependent from a set of vectors $\{\mathbf{x}_1 \quad \mathbf{x}_2 \quad \cdots\}$ if it can be expressed as a linear combination, i.e. $\mathbf{y} = c_1 \cdot \mathbf{x}_1 + c_2 \cdot \mathbf{x}_2 + \cdots$

$\mathbf{y} \neq \mathbf{0}$ is linearly independent from a set of vectors $\{\mathbf{x}_1 \quad \mathbf{x}_2 \quad \cdots\}$ if there is **no** such linear combination.

$\mathbf{X}^{-1}$ inverse matrix: $\mathbf{X}^{-1} \cdot \mathbf{X} = \mathbf{X} \cdot \mathbf{X}^{-1} = \mathbf{1}$; $\mathbf{X}$ must be square and its inverse does not always exist. It only exists if $\mathbf{M}$ has 'full rank', i.e. all rows or all columns are linearly independent from each other.

A matrix $\mathbf{X}$ is called orthogonal if:

$$\mathbf{X}^{-1} = \mathbf{X}^{\mathbf{T}} \quad \leftrightarrow \quad \mathbf{X} \cdot \mathbf{X}^{\mathbf{T}} = \mathbf{X}^{\mathbf{T}} \cdot \mathbf{X} = \mathbf{1}$$

$\hat{x}$ or $\hat{\mathbf{x}}$ least-squares estimate of the true but inaccessible value x or $\mathbf{x}$, respectively.

The scalar product (or dot product) of two vectors $\mathbf{a}$ and $\mathbf{b}$ with the same length is defined as $\langle \mathbf{a} | \mathbf{b} \rangle = \mathbf{a} \cdot \mathbf{b} = \sum_i a_i \cdot b_i$.

If $\mathbf{x}$ and $\mathbf{y}$ are orthogonal, their dot product $\langle \mathbf{x} | \mathbf{y} \rangle = 0$; such vectors are also called perpendicular.

If $\mathbf{x}$ and $\mathbf{y}$ are orthonormal, their dot product is either $\langle \mathbf{x} | \mathbf{y} \rangle = 0$ or $\langle \mathbf{x} | \mathbf{y} \rangle = 1$. The former dot product holds if $\mathbf{x} \neq \mathbf{y}$; the latter if and only if $\mathbf{x} = \mathbf{y}$.

The Euclidean length of a vector $\mathbf{x}$ is a scalar and is defined as: $\|\mathbf{x}\|_2 = \sqrt{\langle \mathbf{x} | \mathbf{x} \rangle}$. Its length is normalized to one by multiplying each element by $1/\|\mathbf{x}\|_2$.

A vector space is the 'range' a certain set of vectors can cover, i.e. all possible linear combinations of these vectors. In other words, a linear combination of vectors from a certain vector space is again a member of the vector space; any vector from the vector space times a constant scalar is also a member of that vector space. A more descriptive way is to say a set of vectors span a vector space. For example, $\mathbf{x}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\mathbf{x}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ span the X-Y plane R^2. Thus, the vector space R^2 contains (or 'covers') all linear combinations that can be made from $\mathbf{x}_1$ and $\mathbf{x}_2$.

A subvector space is introduced by means of an example: R^2 is the vector space of the 2-dimensional X-Y plane; R^3 is the vector space of the 3-dimensional X-Y-Z space, i.e. all vectors with three elements. Obviously, R^2 is a subvector space of R^3, i.e. $R^2 \subset R^3$, or in other words, R^2 is contained in R^3. These two definitions are not strict in a mathematical sense but covers what is needed here.

The basis of a vector space is a minimum set of linearly independent vectors by means of which all elements of a vector space can be expressed as a linear combination. $\begin{pmatrix} 3 \\ 1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$ are a basis for R^2. An orthonormal basis of a vector space contains basis vectors that are mutually orthonormal. $\mathbf{x}_1$ and $\mathbf{x}_2$ from above are an orthonormal basis for R^2.

A vector $\mathbf{y} = c_1 \cdot \mathbf{x}_1 + c_2 \cdot \mathbf{x}_2$ is expressed as a linear combination of (here) two basis vectors and corresponding coordinates c_1 and c_2. If the basis vectors change, the coordinates also change.

Chapter 1

Introduction: Chemometrics and its Application to Spectroscopic Data Analysis

Combining spectroscopy with imaging techniques, commonly known as spectroscopic imaging, has significantly enhanced many studies of heterogeneous samples which require high spatial resolution [6] - [8]. The data acquired from a spectroscopic imager are usually arranged in the form of a data cube (see Figure 1). The X and Y-dimensions of the cube represent the spatial dimensions of the sample and the Z-dimension contains the corresponding spectral information. Thus, one can imagine such a data set as a stack of images acquired at different wavelengths. Chemometric algorithms are often applied to data cubes to extract the desired chemical information. Chemometrics [9] - [15] is the chemical discipline that uses statistical methods to evaluate measurement data. Specifically, chemometrics can determine which analytes are present within a sample, their spatial distribution, and how much of the analyte is present. Multivariate image analysis (MIA) [16], a standard chemometric tool, is used for visualizing spectroscopic imaging data. MIA produces color coded images which represent the distributions of different spectroscopic features as different colors. As an example, MIA is applied to data acquired from a microscopic *E. coli B* sample (see Figure 2). The color image in Figure 2 (c) clearly discriminates the different spectroscopic information.

Most current imaging systems make use of a 2-dimensional focal plane array (FPA) detector that acquires a chemical spectrum with each pixel of the detector [17]. Standard sizes of modern day FPAs easily produce thousands to tens of thousands of spectra within a short amount of time (i.e. < 1 minute). This amount of information results in data sets ranging in size from several megabytes to several gigabytes. When chemometric algorithms are applied to this amount of data, serious challenges arise. Time resolution, as encountered in online applications, and computational resources, in

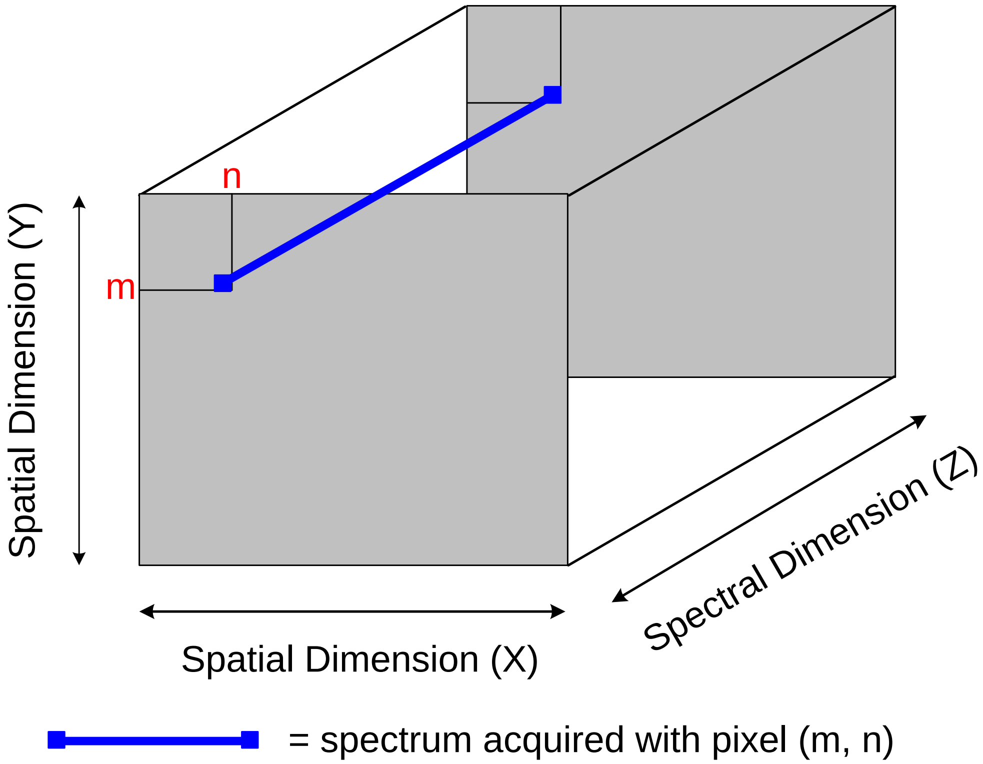


Figure 1: Example of a 3-dimensional spectroscopic data cube obtained with a 2-dimensional detector. The data cube contains two spatial dimensions, X and Y, and one spectral dimension, Z. Each pixel (m, n) of the detector measures an individual chemical spectrum from a specific sample location.

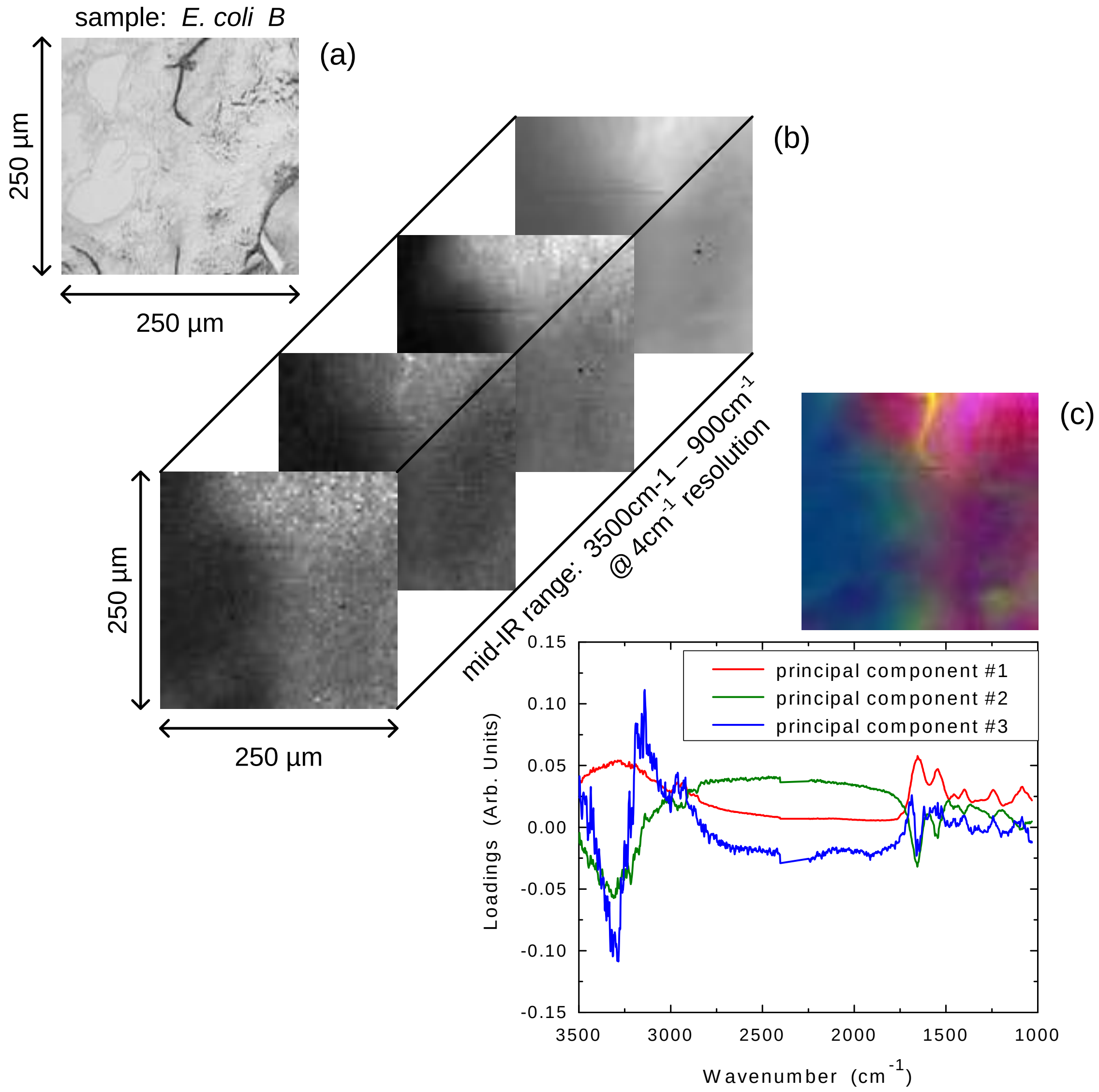


Figure 2: (a) A visible image of an *E. coli B* sample – no chemical information is available. (b) A spectroscopic data cube acquired from the sample contains 4096 spectra from different sample locations. (c) MIA extracts the relevant spectroscopic information and depicts its spatial distribution as different colors.

general, computer memory and processing speed, become greatly affected. Therefore, in order to overcome these burdens and accelerate data analysis times without sacrificing the quality of the final results, it is mandatory that compression methods be developed. These methods must extract the relevant information within a data set without introducing excessive computations.

To facilitate data compression, thus accelerating computation times, wavelet transforms are utilized. 1-dimensional (1D) wavelet transforms (WTs) [18] - [26] have become an important chemometric tool for denoising and data compression [23], [27] - [30]. WTs are linear transformations and preserve data structure unlike the more commonly known 'ZIP' method [31]. ZIP alters data structure such that chemometric algorithms can no longer be applied. Therefore, chemometrics can be applied directly to compressed data sets [32] - [34]. Recently, 1D WTs have been expanded to high-dimensional (or multi-dimensional) WTs [35] which allows for data compression in multiple dimensions. This is greatly beneficial for compressing 3D data cubes (Figure 1). Applying the same wavelet to each dimension of a multi-dimensional data set is not required from a mathematical perspective. Since different data set dimensions contain different types of information, different wavelets, or 'hybrid wavelets', can be combined and applied to each dimension in order to optimize compression for spectroscopic imaging [36], [37]. Additional computation expense is introduced when wavelet compression is applied; however, it is highly over-compensated during subsequent chemometric analyses which then handle much smaller data sets. The reason for this over-compensation is that the computation time required to perform wavelet compression increases linearly with data set size; computation times for a principal component analysis (PCA), for example, are decreased in second and third order [32], [38] when these smaller data sets are analyzed. It has been demonstrated that the quality of the chemometric models derived from compressed data is comparable to that of the models derived from the uncompressed data [36], [37].

To provide a solution [1], [2] for the severe computational demands produced by spectroscopic imaging data sets, two novel compression techniques utilizing multi-dimensional WTs are presented in this thesis. The first compression method describes an automated selection routine that chooses the optimum wavelet combination for any given data set [1]. PCA is applied in this method for demonstration purposes, but many other chemometric algorithms can also be employed. The second compression technique addresses the nonlinear counterpart of PCA, kernel principal component analysis (KPCA) [39], [40]. KPCA is a relatively new chemometric method and is experiencing a rapid increase in popularity given its ability to model nonlinear relationships present in complex data [41] - [46]. Unfortunately, the vigorous mechanics involved with KPCA prevent it from being easily applied to large data sets as encountered in spectroscopic imaging. Therefore, a compression algorithm is presented that accelerates KPCA, making it feasible on common desktop computers [2]. These proposed compression routines reduce data set sizes and accelerate computation speed while maintaining high-quality data representation. The capabilities of these compression-based methods are assessed through application to multiple spectroscopic imaging data sets obtained from different experimental setups.

The organization of the thesis is as follows: First, the background and theory of chemometrics and the algorithms applied in this thesis, PCA and KPCA, are established in Chapter 2 along with a qualitative description of a wavelet transform. Chapter 3 presents a selection algorithm for automatically determining the optimum wavelet combinations for any given data set. A compression routine for the nonlinear counterpart to PCA, KPCA, is outlined in Chapter 4. Conclusions and future outlook based on the research in this thesis are given in Chapter 5. Appendix 1 provides a description of a project that is conducted in collaboration with the United States Department of Agriculture (USDA), Cotton Structure and Quality Research Unit. This project involves the development of new chemometric procedures for assessing cotton quality.

Chapter 2

Background and Theory of Chemometric Algorithms and Wavelet Transforms

This chapter supplies the background and basic ideas utilized in least-squares regression. Univariate Least-Squares Regression linearly relates one response variable (i.e. concentration) to one predictor variable (i.e. absorbance). However, if the response variable (concentration) is dependent on several predictor variables (absorbance values), then the univariate model is extended to Multivariate Least-Squares Regression. Since many chemometric algorithms are based on least-squares regression, the fundamental principles of Univariate Least-Squares Regression are introduced in Chapter 2.1. The expansion towards Multivariate Least-Squares Regression (MLR) [47] - [49] is exemplified by means of Classical Least-Squares (CLS, Chapter 2.2), a generalization of Beer's Law [50]. However, CLS can only be applied in special cases and for many real-world situations a more general approach is required. For this purpose Principal Component Analysis / Regression (PCA/PCR) [11], [51] is introduced in Chapter 2.3. Chapter 2.4 describes a nonlinear form of PCA referred to as Kernel Principal Component Analysis (KPCA) [39], [40] which has been shown in recent years to be successful at modeling nonlinear behavior in various experimental studies [41] - [46]. Finally, wavelet transforms [19] - [30] are discussed in Chapter 2.5 and are used significantly in Chapters 3 and 4 [1], [2].

2.1 Principles of Univariate Least Squares Regression

Beer's Law (1) states that the absorption $A(\lambda)$ at a certain user selected wavelength λ is the product of the molar absorptivity $\varepsilon(\lambda)$ times absorption pathlength L times concentration c of an analyte [3], [50]:

$$A(\lambda) = \varepsilon(\lambda) \cdot L \cdot c$$

(1)

If only one analyte is absorbing at λ, Beer’s Law can be used to quantify this analyte. For this purpose, a calibration sample of known concentration c^{cal} is prepared. After measuring its absorbance $A^{cal}(\lambda)$ the $\varepsilon(\lambda) \cdot L$ can be determined by:

$$\varepsilon(\lambda) \cdot L = \frac{A^{cal}(\lambda)}{c^{cal}}$$

(2)

This procedure (2) is called calibration. Since $\varepsilon(\lambda) \cdot L$ does not change over time, this information (2) can be used to determine the concentration c^{meas} of an unknown sample by measuring its absorbance $A^{meas}(\lambda)$ and calculating c^{meas} by:

$$c^{meas} = \frac{A^{meas}(\lambda)}{\varepsilon(\lambda) \cdot L}$$

(3)

However, in real-world applications this procedure is not feasible because of measurement errors. If a measurement error δ impacts the measurement, (1) is not accessible and one has to deal with:

$$A(\lambda) = \varepsilon(\lambda) \cdot L \cdot c + \delta$$

(4)

This imposes an unsolvable one-equation-two-unknowns problem, i.e. either $\varepsilon \cdot L$ and δ in (2) or c^{meas} and δ in (3) are unknown. Further, preparing only one calibration sample while assuming $c = 0 \Rightarrow A = 0$ as a second point of a straight calibration curve should be avoided because measurement errors in (2) falsify the calibration curve, resulting in concentration errors. Thus, a calibration curve should be based on several calibration samples because then errors are ‘averaged out’. K calibration samples with known calibration concentrations $c_{i=1\cdots K}^{cal}$ are prepared and the corresponding absorbance values $A_{i=1\cdots K}^{cal}$ are acquired. This results in the following equations, which are all impacted by random and different measurement errors $\delta_{1\cdots K}$:

$$\begin{aligned} A_1^{cal} &= \varepsilon \cdot L \cdot c_1^{cal} + d + \delta_1 \\ &\vdots \\ A_K^{cal} &= \varepsilon \cdot L \cdot c_K^{cal} + d + \delta_K \end{aligned}$$

(5)

As shown in Figure 3, a straight line $y = m \cdot x + d$ or here $A = (\varepsilon \cdot L) c + d$ is fitted to these measurement data. Details on how the slope $\varepsilon \cdot L$ and intercept d are determined are discussed in the remainder of this chapter. Beer's Law does not incorporate an intercept; nonetheless, a straight line has two degrees of freedom, slope and intercept. In this case, $d \approx 0$ is expected because in theory $c = 0 \Rightarrow A = 0$.

Parameters $\varepsilon \cdot L$ and d that closely fulfill all K equations (5) are required. However, due to random noise this is not possible because there are K equations and $K + 2$ unknowns, i.e. $\varepsilon \cdot L$, d and $\delta_{1 \cdots K}$. Thus, one has to estimate *good overall parameters* $\hat{\varepsilon} \cdot L$ and $\hat{d}$. The symbol $\hat{}$ has been introduced to discriminate between true but inaccessible and estimated parameters. This estimation is done by a least-squares regression. The fundamental principle of least-squares regression is: ***Select model parameters*** $\hat{\varepsilon} \cdot L$ **and** $\hat{d}$ ***such that the sum of squared errors*** S ***is minimized***, *i.e.*:

$$\begin{aligned} S &= \sum_{i=1}^{K} \delta_i^2 \\ &= \sum_{i=1}^{K} \left(A_i^{cal} - \hat{\varepsilon} \cdot L \cdot c_i^{cal} - \hat{d} \right)^2 \overset{!}{=} \min \end{aligned}$$

(6)

Squared errors prevent canceling of positive and negative deviations.

In order to determine this minimum (or to be more rigorous, extrema) we compute partial derivatives with respect to the wanted parameters and set them equal to zero:

$$\begin{aligned} \frac{\partial S}{\partial \hat{d}} &= 0 = \sum_{i=1}^{K} \left[2 \cdot \left(A_i^{cal} - \hat{\varepsilon} \cdot L \cdot c_i^{cal} - \hat{d} \right) \cdot (-1) \right] \\ \frac{\partial S}{\partial (\hat{\varepsilon} \cdot L)} &= 0 = \sum_{i=1}^{K} \left[2 \cdot \left(A_i^{cal} - \hat{\varepsilon} \cdot L \cdot c_i^{cal} - \hat{d} \right) \cdot \left(-c_i^{cal} \right) \right] \end{aligned}$$

(7)

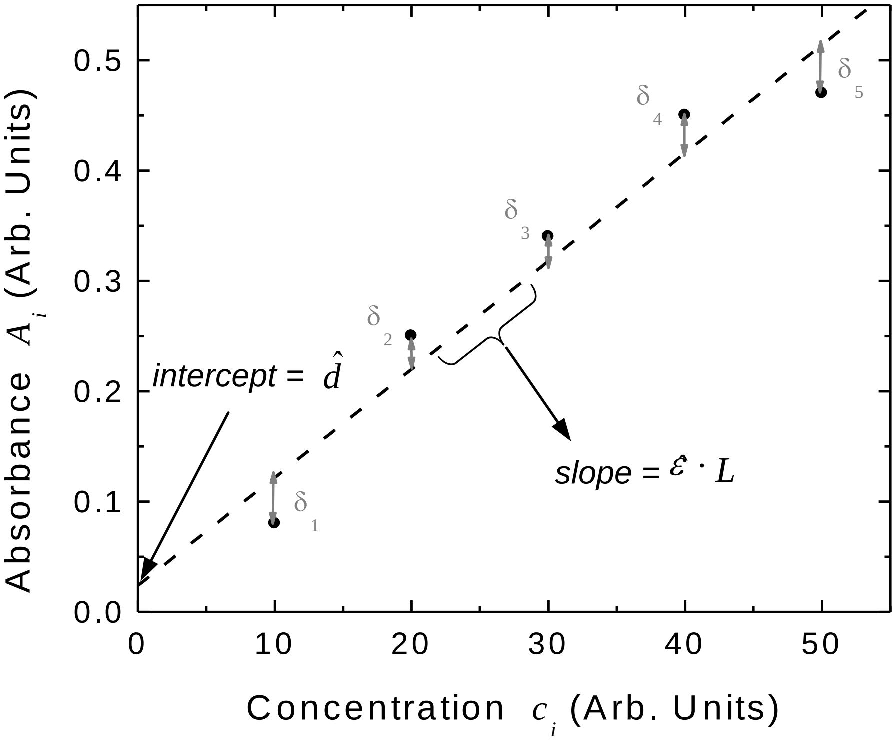


Figure 3: $K = 5$ calibration samples have been used to determine a calibration curve by means of least-squares regression (simulated data).

Strictly speaking, the parameters $\hat{\varepsilon} \cdot L$ and $\hat{d}$ that fulfill the equation (7) are extrema, i.e. minima or maxima. However, they could be chosen completely out of the way such that S (6) gets arbitrarily large. Thus, it can be *assumed* that the parameters found after solving these equations minimize S .

We now have two equations (7) and two unknown parameters. Therefore, we can begin to solve this equation system for $\hat{\varepsilon} \cdot L$ and $\hat{d}$. For the final result, refer directly to equation (9):

$$0 = \sum_{i=1}^{K} \left(A_i^{cal} - \hat{\varepsilon} \cdot L \cdot c_i^{cal} - \hat{d} \right) = \sum_{i=1}^{K} A_i^{cal} - \sum_{i=1}^{K} \hat{\varepsilon} \cdot L \cdot c_i^{cal} - \sum_{i=1}^{K} \hat{d} \cdot 1$$

$$0 = \sum_{i=1}^{K} \left(A_i^{cal} - \hat{\varepsilon} \cdot L \cdot c_i^{cal} - \hat{d} \right) \cdot c_i^{cal} = \sum_{i=1}^{K} A_i^{cal} \cdot c_i^{cal} - \sum_{i=1}^{K} \hat{\varepsilon} \cdot L \cdot \left(c_i^{cal} \right)^2 - \sum_{i=1}^{K} \hat{d} \cdot c_i^{cal}$$

The factor of 1 in the right hand side of the first equation has been introduced to make the following derivation more clear. Now we bring items we know to the right side:

$$\sum_{i=1}^{K} \hat{d} \cdot 1 + \sum_{i=1}^{K} \hat{\varepsilon} \cdot L \cdot c_i^{cal} = \sum_{i=1}^{K} A_i^{cal}$$

$$\sum_{i=1}^{K} \hat{d} \cdot c_i^{cal} + \sum_{i=1}^{K} \hat{\varepsilon} \cdot L \cdot \left(c_i^{cal} \right)^2 = \sum_{i=1}^{K} A_i^{cal} \cdot c_i^{cal}$$

The left hand sides are rewritten to make the following step more clear:

$$\hat{d} \cdot \sum_{i=1}^{K} 1 + \hat{\varepsilon} \cdot L \cdot \sum_{i=1}^{K} c_i^{cal} = \sum_{i=1}^{K} A_i^{cal}$$

$$\hat{d} \cdot \sum_{i=1}^{K} c_i^{cal} + \hat{\varepsilon} \cdot L \cdot \sum_{i=1}^{K} \left(c_i^{cal} \right)^2 = \sum_{i=1}^{K} A_i^{cal} \cdot c_i^{cal}$$

This can be formally written as a vector-matrix equation:

$$\begin{pmatrix} \sum_{i=1}^{K} 1 & \sum_{i=1}^{K} c_i^{cal} \\ \sum_{i=1}^{K} c_i^{cal} & \sum_{i=1}^{K} \left(c_i^{cal} \right)^2 \end{pmatrix} \cdot \begin{pmatrix} \hat{d} \\ \hat{\varepsilon} \cdot L \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{K} A_i^{cal} \\ \sum_{i=1}^{K} A_i^{cal} \cdot c_i^{cal} \end{pmatrix}$$

(8)

We *formally* define a vector $\mathbf{a}$ and a matrix $\mathbf{C}^{cal}$ which contain data we know because we prepare the calibration samples and measure the calibration spectra’s absorbance values:

$$\mathbf{a}^{cal}=\begin{pmatrix}A_1^{cal}\\ \vdots\\ A_K^{cal}\end{pmatrix}\qquad \mathbf{C}^{cal}=\begin{pmatrix}1 & c_1^{cal}\\ \vdots & \vdots\\ 1 & c_K^{cal}\end{pmatrix}\quad \text{and thus}\quad {\mathbf{C}^{cal}}^{\mathbf{T}}=\begin{pmatrix}1 & \cdots & 1\\ c_1^{cal} & \cdots & c_K^{cal}\end{pmatrix}$$

It is straightforward to verify that:

$$\begin{pmatrix}\sum_{i=1}^{K}1 & \sum_{i=1}^{K}c_i^{cal}\\ \sum_{i=1}^{K}c_i^{cal} & \sum_{i=1}^{K}\left(c_i^{cal}\right)^2\end{pmatrix}={\mathbf{C}^{cal}}^{\mathbf{T}}\cdot\mathbf{C}^{cal}\quad \text{and}\quad \begin{pmatrix}\sum_{i=1}^{K}A_i^{cal}\cdot 1\\ \sum_{i=1}^{K}A_i^{cal}\cdot c_i^{cal}\end{pmatrix}={\mathbf{C}^{cal}}^{\mathbf{T}}\cdot\mathbf{a}^{cal}$$

Again, a factor of 1 has been introduced to the right equation to make the following more clear. Now we can rewrite (8) as:

$${\mathbf{C}^{cal}}^{\mathbf{T}}\cdot\mathbf{C}^{cal}\cdot\begin{pmatrix}\hat{d}\\ \hat{\varepsilon}\cdot L\end{pmatrix}={\mathbf{C}^{cal}}^{\mathbf{T}}\cdot\mathbf{a}^{cal}$$

In the last step we multiply by the matrix $\left({\mathbf{C}^{cal}}^{\mathbf{T}}\cdot\mathbf{C}^{cal}\right)^{-1}$ from the left to both sides and get:

$$\underbrace{\left({\mathbf{C}^{cal}}^{\mathbf{T}}\cdot\mathbf{C}^{cal}\right)^{-1}\cdot{\mathbf{C}^{cal}}^{\mathbf{T}}\cdot\mathbf{C}^{cal}}_{=\mathbf{1}_{2x2}}\cdot\begin{pmatrix}\hat{d}\\ \hat{\varepsilon}\cdot L\end{pmatrix}=\left({\mathbf{C}^{cal}}^{\mathbf{T}}\cdot\mathbf{C}^{cal}\right)^{-1}\cdot{\mathbf{C}^{cal}}^{\mathbf{T}}\cdot\mathbf{a}$$

$$\begin{pmatrix}\hat{d}\\ \hat{\varepsilon}\cdot L\end{pmatrix}=\left({\mathbf{C}^{cal}}^{\mathbf{T}}\cdot\mathbf{C}^{cal}\right)^{-1}\cdot{\mathbf{C}^{cal}}^{\mathbf{T}}\cdot\mathbf{a}\tag{9}$$

The right side of (9) consists of known calibration information and the left side contains the estimates of the wanted parameters (Figure 3). These parameters determine the calibration curve and are optimum in the least-squares sense. Least-squares fitting assumes random, independent, and normally distributed measurement errors. If there is any kind of systematic error, the experiment or the sensor should be checked.

2.2 Classical Least-Squares (CLS)

The procedure outlined in Chapter 2.1 is a standard component of analytical chemistry. However, this approach can only be applied if only one analyte absorbs at the chosen wavelength λ (*univariate* regression). If this is not the case and several analytes absorb at the chosen wavelength λ, there will be cross-sensitivities from other analytes and the sensor's selectivity will be insufficient. The reason for this is that overlapping absorptions of several analytes add linearly (refer to (10), [50]). This is somewhat over-simplified because analyses of mixtures with overlapping absorbance bands are typical tasks chemists encounter in real-world applications.

In order to keep the following discussion concise, only two analytes are considered here; the augmentation to more analytes is straightforward. Instead of (1) one has to handle:

$$A(\lambda)=\varepsilon_1(\lambda)\cdot L\cdot c_1+\varepsilon_2(\lambda)\cdot L\cdot c_2$$

(10)

The unsolvable problem here (10) is obvious: only one absorbance $A(\lambda)$ is measured but the concentrations of two analytes, i.e. c_1 and c_2, need to be determined. To overcome this insufficient amount of information, the absorbance is measured at two different wavelengths λ_1 and λ_2. Now we have two equations and two unknowns:

$$\begin{aligned}A(\lambda_1)&=\varepsilon_1(\lambda_1)\cdot L\cdot c_1+\varepsilon_2(\lambda_1)\cdot L\cdot c_2\\ A(\lambda_2)&=\varepsilon_1(\lambda_2)\cdot L\cdot c_1+\varepsilon_2(\lambda_2)\cdot L\cdot c_2\end{aligned}$$

(11)

In order to simplify the notation we change (11) to (12) where the first index of ε refers to the wavelength position and the second to the analyte number. The index of A refers to wavelength position:

$$\begin{aligned}A_1&=\varepsilon_{11}\cdot L\cdot c_1+\varepsilon_{12}\cdot L\cdot c_2\\ A_2&=\varepsilon_{21}\cdot L\cdot c_1+\varepsilon_{22}\cdot L\cdot c_2\end{aligned}$$

(12)

Since we are dealing now with more than one predictor variable (= absorbance at different wavelengths), the following procedure is called *Multivariate* Least-Squares Regression (MLR) [47] - [49]. For solving this equation system we need to know the constant parameters $\varepsilon_{11}\cdot L$, $\varepsilon_{12}\cdot L$, $\varepsilon_{21}\cdot L$ and $\varepsilon_{22}\cdot L$. These four parameters have to be determined experimentally by means of a calibration. The following discussion is a detailed derivation of Classical Least-Squares (CLS), a MLR based approach. If only the final equations are of interest to the reader, refer to (26) and (31). Equation (26) uses calibration data to determine the parameters required in (12). After this calibration, equation (31) is used to determine the concentrations of unknown samples.

2.2.1 CLS – Calibration

To determine these four parameters (12) experimentally, at least four equations are required. To derive them we have to prepare two calibration samples of known composition; i.e. c_{11}^{cal} and c_{21}^{cal} (sample #1) and c_{12}^{cal} and c_{22}^{cal} (sample #2). The first index of c^{cal} refers to the analyte number and the second to the sample number. Then absorbance values are measured for both samples at wavelength position λ_1 and λ_2. In the following equation (13), the first index of A^{cal} and ε refers to the wavelength position; the second index of A^{cal} refers to sample number and the second index of ε to the analyte number.

$$\begin{array}{ll} \text{calibration sample \#1} & \begin{cases} A_{11}^{cal} = \varepsilon_{11} \cdot L \cdot c_{11}^{cal} + \varepsilon_{12} \cdot L \cdot c_{21}^{cal} \\ A_{21}^{cal} = \varepsilon_{21} \cdot L \cdot c_{11}^{cal} + \varepsilon_{22} \cdot L \cdot c_{21}^{cal} \end{cases} \\ \text{calibration sample \#2} & \begin{cases} A_{12}^{cal} = \varepsilon_{11} \cdot L \cdot c_{12}^{cal} + \varepsilon_{12} \cdot L \cdot c_{22}^{cal} \\ A_{22}^{cal} = \varepsilon_{21} \cdot L \cdot c_{12}^{cal} + \varepsilon_{22} \cdot L \cdot c_{22}^{cal} \end{cases} \end{array} \tag{13}$$

c_{11}^{cal} and c_{21}^{cal} in each of the top two equations (calibration sample #1) are the same because these equations are measured from the same sample. Similarly, c_{12}^{cal} and c_{22}^{cal} are the same values in the bottom two equations (calibration sample #2). For both calibration samples $\varepsilon_{11} \cdot L$, $\varepsilon_{12} \cdot L$, $\varepsilon_{21} \cdot L$ and $\varepsilon_{22} \cdot L$ are equal because they are physical properties of the analyte. Solving the equation system (13) determines the four unknowns $\varepsilon_{11} \cdot L$, $\varepsilon_{12} \cdot L$, $\varepsilon_{21} \cdot L$ and $\varepsilon_{22} \cdot L$, which are required later on for analyses of unknown samples.

However, this only works in an ideal case, i.e. in the absence of measurement errors – but there are always measurement errors. Thus, (13) is not accessible and we have to deal with the following:

$$\text{calibration sample \#1} \quad \begin{cases} A_{11}^{cal} = \varepsilon_{11} \cdot L \cdot c_{11}^{cal} + \varepsilon_{12} \cdot L \cdot c_{21}^{cal} + \delta_1 \\ A_{21}^{cal} = \varepsilon_{21} \cdot L \cdot c_{11}^{cal} + \varepsilon_{22} \cdot L \cdot c_{21}^{cal} + \delta_2 \end{cases}$$

$$\text{calibration sample \#2} \quad \begin{cases} A_{12}^{cal} = \varepsilon_{11} \cdot L \cdot c_{12}^{cal} + \varepsilon_{12} \cdot L \cdot c_{22}^{cal} + \delta_3 \\ A_{22}^{cal} = \varepsilon_{21} \cdot L \cdot c_{12}^{cal} + \varepsilon_{22} \cdot L \cdot c_{22}^{cal} + \delta_4 \end{cases}$$

(14)

In (14) there are four equations and eight unknowns; i.e. $\varepsilon_{11} \cdot L$, $\varepsilon_{12} \cdot L$, $\varepsilon_{21} \cdot L$, $\varepsilon_{22} \cdot L$ and the measurement errors $\delta_{1\cdots4}$. These random measurement errors leave us with an unsolvable problem Even measuring absorbances at one more wavelength position would add one more equation while also introducing an additional unknown measurement error δ. Thus, we are always short of information. Like in Chapter 2.1, we have to estimate *good overall parameters* $\hat{\varepsilon}_{11} \cdot L$, $\hat{\varepsilon}_{12} \cdot L$, $\hat{\varepsilon}_{21} \cdot L$ and $\hat{\varepsilon}_{22} \cdot L$. This estimation is again based on the least-squares principle of *minimizing the sum of squared errors* S (compare (6)):

$$\begin{aligned} S &= \sum_i \delta_i^2 \\ &= \left(A_{11}^{cal} - \hat{\varepsilon}_{11} \cdot L \cdot c_{11}^{cal} - \hat{\varepsilon}_{12} \cdot L \cdot c_{21}^{cal}\right)^2 + \left(A_{21}^{cal} - \hat{\varepsilon}_{21} \cdot L \cdot c_{11}^{cal} - \hat{\varepsilon}_{22} \cdot L \cdot c_{21}^{cal}\right)^2 \\ &+ \left(A_{12}^{cal} - \hat{\varepsilon}_{11} \cdot L \cdot c_{12}^{cal} - \hat{\varepsilon}_{12} \cdot L \cdot c_{22}^{cal}\right)^2 + \left(A_{22}^{cal} - \hat{\varepsilon}_{21} \cdot L \cdot c_{12}^{cal} - \hat{\varepsilon}_{22} \cdot L \cdot c_{22}^{cal}\right)^2 \end{aligned}$$

(15)

To determine values $\hat{\varepsilon}_{11} \cdot L$, $\hat{\varepsilon}_{12} \cdot L$, $\hat{\varepsilon}_{21} \cdot L$ and $\hat{\varepsilon}_{22} \cdot L$ that minimize S, partial derivatives with respect to the unknowns are computed and set equal to zero. Now we have four equations (16) and four unknowns.

$$\frac{\partial S}{\partial\left(\hat{\varepsilon}_{11} \cdot L\right)} = 0 \qquad \frac{\partial S}{\partial\left(\hat{\varepsilon}_{12} \cdot L\right)} = 0 \qquad \frac{\partial S}{\partial\left(\hat{\varepsilon}_{21} \cdot L\right)} = 0 \qquad \frac{\partial S}{\partial\left(\hat{\varepsilon}_{22} \cdot L\right)} = 0$$

(16)

From (16) we start to separate unknown and known items and we derive:

$$\begin{aligned} \hat{\varepsilon}_{11} \cdot L \cdot c_{11}^{cal} \cdot c_{11}^{cal} + \hat{\varepsilon}_{12} \cdot L \cdot c_{21}^{cal} \cdot c_{11}^{cal} + \hat{\varepsilon}_{11} \cdot L \cdot c_{12}^{cal} \cdot c_{12}^{cal} + \hat{\varepsilon}_{12} \cdot L \cdot c_{22}^{cal} \cdot c_{12}^{cal} &= A_{11}^{cal} \cdot c_{11}^{cal} + A_{12}^{cal} \cdot c_{12}^{cal} \quad (\mathrm{I}) \\ \hat{\varepsilon}_{11} \cdot L \cdot c_{11}^{cal} \cdot c_{21}^{cal} + \hat{\varepsilon}_{12} \cdot L \cdot c_{21}^{cal} \cdot c_{21}^{cal} + \hat{\varepsilon}_{11} \cdot L \cdot c_{12}^{cal} \cdot c_{22}^{cal} + \hat{\varepsilon}_{12} \cdot L \cdot c_{22}^{cal} \cdot c_{22}^{cal} &= A_{11}^{cal} \cdot c_{21}^{cal} + A_{12}^{cal} \cdot c_{22}^{cal} \quad (\mathrm{II}) \\ \hat{\varepsilon}_{21} \cdot L \cdot c_{11}^{cal} \cdot c_{11}^{cal} + \hat{\varepsilon}_{22} \cdot L \cdot c_{21}^{cal} \cdot c_{11}^{cal} + \hat{\varepsilon}_{21} \cdot L \cdot c_{12}^{cal} \cdot c_{12}^{cal} + \hat{\varepsilon}_{22} \cdot L \cdot c_{22}^{cal} \cdot c_{12}^{cal} &= A_{21}^{cal} \cdot c_{11}^{cal} + A_{22}^{cal} \cdot c_{12}^{cal} \quad (\mathrm{III}) \\ \hat{\varepsilon}_{21} \cdot L \cdot c_{11}^{cal} \cdot c_{21}^{cal} + \hat{\varepsilon}_{22} \cdot L \cdot c_{21}^{cal} \cdot c_{21}^{cal} + \hat{\varepsilon}_{21} \cdot L \cdot c_{12}^{cal} \cdot c_{22}^{cal} + \hat{\varepsilon}_{22} \cdot L \cdot c_{22}^{cal} \cdot c_{22}^{cal} &= A_{21}^{cal} \cdot c_{21}^{cal} + A_{22}^{cal} \cdot c_{22}^{cal} \quad (\mathrm{IV}) \end{aligned}$$

(17)

Consider the right side of (17) – both equations (I) and (II) must hold simultaneously. We also rewrite this sum as a multiplication of a row and column vector:

$$(\mathrm{I}) = \begin{pmatrix} A_{11}^{cal} & A_{12}^{cal} \end{pmatrix} \cdot \begin{pmatrix} c_{11}^{cal} \\ c_{12}^{cal} \end{pmatrix}$$

$$(\mathrm{II}) = \begin{pmatrix} A_{11}^{cal} & A_{12}^{cal} \end{pmatrix} \cdot \begin{pmatrix} c_{21}^{cal} \\ c_{22}^{cal} \end{pmatrix}$$

Thus, we combine them to give one equation; the left hand side of (18) is a row vector:

$$\begin{pmatrix} (\mathrm{I}) & (\mathrm{II}) \end{pmatrix} = \begin{pmatrix} A_{11}^{cal} & A_{12}^{cal} \end{pmatrix} \cdot \begin{pmatrix} c_{11}^{cal} & c_{21}^{cal} \\ c_{12}^{cal} & c_{22}^{cal} \end{pmatrix}$$

(18)

In an equivalent approach (III) and (IV) are combined:

$$\begin{pmatrix} (\mathrm{III}) & (\mathrm{IV}) \end{pmatrix} = \begin{pmatrix} A_{21}^{cal} & A_{22}^{cal} \end{pmatrix} \cdot \begin{pmatrix} c_{11}^{cal} & c_{21}^{cal} \\ c_{12}^{cal} & c_{22}^{cal} \end{pmatrix}$$

(19)

Also, both equations (18) and (19) must be true at the same time. Thus, we combine them to form one matrix equation:

$$\begin{pmatrix} (\mathrm{I}) & (\mathrm{II}) \\ (\mathrm{III}) & (\mathrm{IV}) \end{pmatrix} = \begin{pmatrix} A_{11}^{cal} & A_{12}^{cal} \\ A_{21}^{cal} & A_{22}^{cal} \end{pmatrix} \cdot \begin{pmatrix} c_{11}^{cal} & c_{21}^{cal} \\ c_{12}^{cal} & c_{22}^{cal} \end{pmatrix} = \mathbf{A}^{cal} \cdot \mathbf{C}^{cal^{\mathbf{T}}}$$

(20)

During calibration all elements of $\mathbf{A}^{cal}$ have been measured and we know the elements of $\mathbf{C}^{cal^{\mathbf{T}}}$ (see indices) due to sample preparation. Let us consider the left side of (17):

$$\begin{aligned}
(\mathrm{I}) &= \left(\hat{\varepsilon}_{11} \cdot L \cdot c_{11}^{cal} + \hat{\varepsilon}_{12} \cdot L \cdot c_{21}^{cal}\right) \cdot c_{11}^{cal} + \left(\hat{\varepsilon}_{11} \cdot L \cdot c_{12}^{cal} + \hat{\varepsilon}_{12} \cdot L \cdot c_{22}^{cal}\right) \cdot c_{12}^{cal} \\
(\mathrm{II}) &= \left(\hat{\varepsilon}_{11} \cdot L \cdot c_{11}^{cal} + \hat{\varepsilon}_{12} \cdot L \cdot c_{21}^{cal}\right) \cdot c_{21}^{cal} + \left(\hat{\varepsilon}_{11} \cdot L \cdot c_{12}^{cal} + \hat{\varepsilon}_{12} \cdot L \cdot c_{22}^{cal}\right) \cdot c_{22}^{cal} \\
(\mathrm{III}) &= \left(\hat{\varepsilon}_{21} \cdot L \cdot c_{11}^{cal} + \hat{\varepsilon}_{22} \cdot L \cdot c_{21}^{cal}\right) \cdot c_{11}^{cal} + \left(\hat{\varepsilon}_{21} \cdot L \cdot c_{12}^{cal} + \hat{\varepsilon}_{22} \cdot L \cdot c_{22}^{cal}\right) \cdot c_{12}^{cal} \\
(\mathrm{IV}) &= \left(\hat{\varepsilon}_{21} \cdot L \cdot c_{11}^{cal} + \hat{\varepsilon}_{22} \cdot L \cdot c_{21}^{cal}\right) \cdot c_{21}^{cal} + \left(\hat{\varepsilon}_{21} \cdot L \cdot c_{12}^{cal} + \hat{\varepsilon}_{22} \cdot L \cdot c_{22}^{cal}\right) \cdot c_{22}^{cal}
\end{aligned}$$

Combine (I) and (II) to form one equation; the left side is a row vector and the right side contains a row vector times a matrix:

$$\begin{pmatrix} (\mathrm{I}) & (\mathrm{II}) \end{pmatrix} = \begin{pmatrix} \hat{\varepsilon}_{11} \cdot L \cdot c_{11}^{cal} + \hat{\varepsilon}_{12} \cdot L \cdot c_{21}^{cal} & \hat{\varepsilon}_{11} \cdot L \cdot c_{12}^{cal} + \hat{\varepsilon}_{12} \cdot L \cdot c_{22}^{cal} \end{pmatrix} \cdot \begin{pmatrix} c_{11}^{cal} & c_{21}^{cal} \\ c_{12}^{cal} & c_{22}^{cal} \end{pmatrix}$$

(21)

and (III) and (IV):

$$\left((III)\quad(IV)\right)=\left(\hat{\varepsilon}_{21}\cdot L\cdot c_{11}^{cal}+\hat{\varepsilon}_{22}\cdot L\cdot c_{21}^{cal}\quad \hat{\varepsilon}_{21}\cdot L\cdot c_{12}^{cal}+\hat{\varepsilon}_{22}\cdot L\cdot c_{22}^{cal}\right)\cdot\begin{pmatrix}c_{11}^{cal} & c_{21}^{cal}\\ c_{12}^{cal} & c_{22}^{cal}\end{pmatrix}$$

(22)

Now, we can combine (21) and (22) to obtain:

$$\begin{pmatrix}(I) & (II)\\ (III) & (IV)\end{pmatrix}=\begin{pmatrix}\hat{\varepsilon}_{11}\cdot L\cdot c_{11}^{cal}+\hat{\varepsilon}_{12}\cdot L\cdot c_{21}^{cal} & \hat{\varepsilon}_{11}\cdot L\cdot c_{12}^{cal}+\hat{\varepsilon}_{12}\cdot L\cdot c_{22}^{cal}\\ \hat{\varepsilon}_{21}\cdot L\cdot c_{11}^{cal}+\hat{\varepsilon}_{22}\cdot L\cdot c_{21}^{cal} & \hat{\varepsilon}_{21}\cdot L\cdot c_{12}^{cal}+\hat{\varepsilon}_{22}\cdot L\cdot c_{22}^{cal}\end{pmatrix}\cdot\begin{pmatrix}c_{11}^{cal} & c_{21}^{cal}\\ c_{12}^{cal} & c_{22}^{cal}\end{pmatrix}$$

(23)

The left side of equation (23) is known due to (20). The right matrix on the right hand side is $\mathbf{C}^{cal\,\mathbf{T}}$, which is known due to sample preparation. The left matrix of the right hand side is only partially known and requires further splitting up. By simple inspection we see:

$$\begin{aligned}\begin{pmatrix}\hat{\varepsilon}_{11}\cdot L\cdot c_{11}^{cal}+\hat{\varepsilon}_{12}\cdot L\cdot c_{21}^{cal} & \hat{\varepsilon}_{11}\cdot L\cdot c_{12}^{cal}+\hat{\varepsilon}_{12}\cdot L\cdot c_{22}^{cal}\\ \hat{\varepsilon}_{21}\cdot L\cdot c_{11}^{cal}+\hat{\varepsilon}_{22}\cdot L\cdot c_{21}^{cal} & \hat{\varepsilon}_{21}\cdot L\cdot c_{12}^{cal}+\hat{\varepsilon}_{22}\cdot L\cdot c_{22}^{cal}\end{pmatrix}&=\begin{pmatrix}\hat{\varepsilon}_{11}\cdot L & \hat{\varepsilon}_{12}\cdot L\\ \hat{\varepsilon}_{21}\cdot L & \hat{\varepsilon}_{22}\cdot L\end{pmatrix}\cdot\begin{pmatrix}c_{11}^{cal} & c_{12}^{cal}\\ c_{21}^{cal} & c_{22}^{cal}\end{pmatrix}\\ &=\hat{\mathbf{E}}\cdot\mathbf{C}^{cal}\end{aligned}$$

(24)

Substituting (24) back into (23) results in:

$$\begin{pmatrix}(I) & (II)\\ (III) & (IV)\end{pmatrix}=\hat{\mathbf{E}}\cdot\mathbf{C}^{cal}\cdot\mathbf{C}^{cal\,\mathbf{T}}$$

(25)

The right side of (20) equals the right side of (25):

$$\mathbf{A}^{cal}\cdot\mathbf{C}^{cal\,\mathbf{T}}=\hat{\mathbf{E}}\cdot\mathbf{C}^{cal}\cdot\mathbf{C}^{cal\,\mathbf{T}}$$

We know the elements of $\mathbf{A}^{cal}$ and $\mathbf{C}^{cal}$ and we need to solve this equation for $\hat{\mathbf{E}}$. For this purpose, both sides are multiplied from the right by $\left(\mathbf{C}^{cal}\cdot\mathbf{C}^{cal\,\mathbf{T}}\right)^{-1}$:

$$\mathbf{A}\cdot\mathbf{C}^{cal\,\mathbf{T}}\cdot\left(\mathbf{C}^{cal}\cdot\mathbf{C}^{cal\,\mathbf{T}}\right)^{-1}=\hat{\mathbf{E}}\cdot\underbrace{\mathbf{C}^{cal}\cdot\mathbf{C}^{cal\,\mathbf{T}}\cdot\left(\mathbf{C}^{cal}\cdot\mathbf{C}^{cal\,\mathbf{T}}\right)^{-1}}_{=\mathbf{1}}=\hat{\mathbf{E}}=\begin{pmatrix}\hat{\varepsilon}_{11}\cdot L & \hat{\varepsilon}_{12}\cdot L\\ \hat{\varepsilon}_{21}\cdot L & \hat{\varepsilon}_{22}\cdot L\end{pmatrix}$$

(26)

The matrix equation (26) computes $\hat{\mathbf{E}}$ whose elements (also compare (1), (10)) are estimates of the four unknown parameters needed in (12). Since the absorption pathlength L has been incorporated into $\hat{\mathbf{E}}$ it must be kept constant during all calibration measurements and during acquisition of the spectra from unknown samples.

It is easy to see how (26) can be expanded to more wavelength positions. If more equations are needed, simply add more rows to $\mathbf{A}^{cal}$ and consequently $\hat{\mathbf{E}}$. In other words, calibration spectra are written into columns of $\mathbf{A}^{cal}$. Each column of $\hat{\mathbf{E}}$ contains the molar absorptivity values (times L) of a certain analyte at the selected wavelength positions. In this thesis, these columns are referred to as molar absorptivity spectra. The calibration concentrations of the analytes are written into the rows of $\mathbf{C}^{cal}$. If more calibration samples are available, more columns are appended to $\mathbf{C}^{cal}$. If more analytes are to be included, rows are added to $\mathbf{C}^{cal}$. In general, a MLR estimate is more precise when more information is used in the procedure. However, adding information that has strong errors has detrimental effects and should be excluded - only high quality information helps to increase the accuracy.

In order to prevent the failure of a calibration set, the following important warning needs to be considered while preparing calibration samples: Calibration concentrations of the individual analytes are written in the rows of the matrix $\mathbf{C}^{cal}$. In (26) the matrix $\left(\mathbf{C}^{cal}\cdot\mathbf{C}^{cal\,\mathbf{T}}\right)^{-1}$ is computed. From a mathematical perspective $\mathbf{C}^{cal}$ must be non-singular for this inverse to exist. This means that it must be avoided to prepare calibration samples with concentrations such as $\mathbf{C}^{cal}=\begin{pmatrix} 0.1\,\text{mol L}^{-1} & 0.2\,\text{mol L}^{-1} & 0.3\,\text{mol L}^{-1} \\ 0.2\,\text{mol L}^{-1} & 0.4\,\text{mol L}^{-1} & 0.6\,\text{mol L}^{-1} \end{pmatrix}$. In this example, the concentrations of the second analyte (second row) are always twice as much as the concentrations of analyte #1; these rows are linearly dependent vectors and the matrix is singular. From a spectroscopic perspective, the spectra look like they are due to only one analyte. In other words, one must avoid preparing calibration samples such that the calibration concentrations of one analyte are a linear combination of the concentrations of other analytes.

After computing $\hat{\mathbf{E}}$ (26) we can determine concentrations of an unknown sample by solving (12) or an appropriately expanded equation system. For this purpose, an equation of similar type like (26) will be used in the following prediction step (Chapter 2.2.2).

2.2.2 CLS – Evaluation of Unknown Samples

Measurement errors Δ_1 and Δ_2 are also contained in the spectrum of an unknown sample. Thus, instead of (12) we have to solve the following equation system:

$$\begin{aligned} A_1 &= \varepsilon_{11} \cdot L \cdot c_1 + \varepsilon_{12} \cdot L \cdot c_2 + \Delta_1 \\ A_2 &= \varepsilon_{21} \cdot L \cdot c_1 + \varepsilon_{22} \cdot L \cdot c_2 + \Delta_2 \end{aligned} \qquad (27)$$

Similar to the calibration procedure (14) we have fewer equations (here: two) than unknowns: c_1, c_2, Δ_1 and Δ_2. Again, there is no way to solve this system analytically. Thus, we cannot determine the true concentrations $\mathbf{c} = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$; instead we have to estimate a vector $\hat{\mathbf{c}} = \begin{pmatrix} \hat{c}_1 \\ \hat{c}_2 \end{pmatrix}$ by means of MLR from a measured spectrum $\mathbf{a} = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix}$. The least-squares principle is applied again, which minimizes the sum of squared errors:

$$\begin{aligned} S &= \sum_i \Delta_i^2 \overset{!}{=} \min \\ &= \left(A_1 - \hat{\varepsilon}_{11} \cdot L \cdot \hat{c}_1 - \hat{\varepsilon}_{12} \cdot L \cdot \hat{c}_2\right)^2 + \left(A_2 - \hat{\varepsilon}_{21} \cdot L \cdot \hat{c}_1 - \hat{\varepsilon}_{22} \cdot L \cdot \hat{c}_2\right)^2 \end{aligned}$$

We compute the partial derivatives with respect to the unknowns (here: $\hat{c}_1$ and $\hat{c}_2$):

$$\frac{\partial S}{\partial \hat{c}_1} = 0 \qquad \textit{and} \qquad \frac{\partial S}{\partial \hat{c}_2} = 0$$

$$\begin{aligned} 0 &= 2 \cdot \left(A_1 - \hat{\varepsilon}_{11} \cdot L \cdot \hat{c}_1 - \hat{\varepsilon}_{12} \cdot L \cdot \hat{c}_2\right) \cdot \left(-\hat{\varepsilon}_{11} \cdot L\right) + 2 \cdot \left(A_2 - \hat{\varepsilon}_{21} \cdot L \cdot \hat{c}_1 - \hat{\varepsilon}_{22} \cdot L \cdot \hat{c}_2\right) \cdot \left(-\hat{\varepsilon}_{21} \cdot L\right) \\ 0 &= 2 \cdot \left(A_1 - \hat{\varepsilon}_{11} \cdot L \cdot \hat{c}_1 - \hat{\varepsilon}_{12} \cdot L \cdot \hat{c}_2\right) \cdot \left(-\hat{\varepsilon}_{12} \cdot L\right) + 2 \cdot \left(A_2 - \hat{\varepsilon}_{21} \cdot L \cdot \hat{c}_1 - \hat{\varepsilon}_{22} \cdot L \cdot \hat{c}_2\right) \cdot \left(-\hat{\varepsilon}_{22} \cdot L\right) \end{aligned}$$

We divide both equations by -2, separate known (right hand side) and unknown items (left hand side) and rewrite the right hand side as a matrix times a vector:

$$\left.\begin{aligned} \left(\hat{\varepsilon}_{11} \cdot L \cdot \hat{c}_1 + \hat{\varepsilon}_{12} \cdot L \cdot \hat{c}_2\right)\hat{\varepsilon}_{11} \cdot L + \left(\hat{\varepsilon}_{21} \cdot L \cdot \hat{c}_1 + \hat{\varepsilon}_{22} \cdot L \cdot \hat{c}_2\right)\hat{\varepsilon}_{21} \cdot L = A_1 \cdot \hat{\varepsilon}_{11} \cdot L + A_2 \cdot \hat{\varepsilon}_{21} \cdot L \\ \left(\hat{\varepsilon}_{11} \cdot L \cdot \hat{c}_1 + \hat{\varepsilon}_{12} \cdot L \cdot \hat{c}_2\right)\hat{\varepsilon}_{12} \cdot L + \left(\hat{\varepsilon}_{21} \cdot L \cdot \hat{c}_1 + \hat{\varepsilon}_{22} \cdot L \cdot \hat{c}_2\right)\hat{\varepsilon}_{22} \cdot L = A_1 \cdot \hat{\varepsilon}_{12} \cdot L + A_2 \cdot \hat{\varepsilon}_{22} \cdot L \end{aligned}\right\} = \hat{\mathbf{E}}^{\mathbf{T}} \cdot \mathbf{a} \qquad (28)$$

The left side of this equation system can be re-organized as:

$$
\begin{aligned}
&= \hat{\varepsilon}_{11}\cdot L\cdot\left(\hat{\varepsilon}_{11}\cdot L\cdot\hat{c}_1+\hat{\varepsilon}_{12}\cdot L\cdot\hat{c}_2\right)+\hat{\varepsilon}_{21}\cdot L\cdot\left(\hat{\varepsilon}_{21}\cdot L\cdot\hat{c}_1+\hat{\varepsilon}_{22}\cdot L\cdot\hat{c}_2\right)\\
&= \hat{\varepsilon}_{12}\cdot L\cdot\left(\hat{\varepsilon}_{11}\cdot L\cdot\hat{c}_1+\hat{\varepsilon}_{12}\cdot L\cdot\hat{c}_2\right)+\hat{\varepsilon}_{22}\cdot L\cdot\left(\hat{\varepsilon}_{21}\cdot L\cdot\hat{c}_1+\hat{\varepsilon}_{22}\cdot L\cdot\hat{c}_2\right)\\
&= \begin{pmatrix}\hat{\varepsilon}_{11}\cdot L & \hat{\varepsilon}_{21}\cdot L\\ \hat{\varepsilon}_{12}\cdot L & \hat{\varepsilon}_{22}\cdot L\end{pmatrix}\cdot\begin{pmatrix}\hat{\varepsilon}_{11}\cdot L\cdot\hat{c}_1+\hat{\varepsilon}_{12}\cdot L\cdot\hat{c}_2\\ \hat{\varepsilon}_{21}\cdot L\cdot\hat{c}_1+\hat{\varepsilon}_{22}\cdot L\cdot\hat{c}_2\end{pmatrix}\\
&= \hat{\mathbf{E}}^{\mathbf{T}}\cdot\begin{pmatrix}\hat{\varepsilon}_{11}\cdot L\cdot\hat{c}_1+\hat{\varepsilon}_{12}\cdot L\cdot\hat{c}_2\\ \hat{\varepsilon}_{21}\cdot L\cdot\hat{c}_1+\hat{\varepsilon}_{22}\cdot L\cdot\hat{c}_2\end{pmatrix}
\end{aligned}
$$

(29)

We have to further split up the right vector in (29) in order to separate known and unknown items. We simply see that:

$$
\begin{pmatrix}\hat{\varepsilon}_{11}\cdot L\cdot\hat{c}_1+\hat{\varepsilon}_{12}\cdot L\cdot\hat{c}_2\\ \hat{\varepsilon}_{21}\cdot L\cdot\hat{c}_1+\hat{\varepsilon}_{22}\cdot L\cdot\hat{c}_2\end{pmatrix}=\begin{pmatrix}\hat{\varepsilon}_{11}\cdot L & \hat{\varepsilon}_{12}\cdot L\\ \hat{\varepsilon}_{21}\cdot L & \hat{\varepsilon}_{22}\cdot L\end{pmatrix}\cdot\begin{pmatrix}\hat{c}_1\\ \hat{c}_2\end{pmatrix}=\hat{\mathbf{E}}\cdot\hat{\mathbf{c}}
$$

(30)

Inserting (30) into (29) and setting the result equal to (28) gives:

$$
\hat{\mathbf{E}}^{\mathbf{T}}\cdot\mathbf{a}=\hat{\mathbf{E}}^{\mathbf{T}}\cdot\hat{\mathbf{E}}\cdot\hat{\mathbf{c}}
$$

In order to solve for $\hat{\mathbf{c}}$, $\left(\hat{\mathbf{E}}^{\mathbf{T}}\cdot\hat{\mathbf{E}}\right)^{-1}$ is multiplied from the left to both sides of the matrix equation:

$$
\left(\hat{\mathbf{E}}^{\mathbf{T}}\cdot\hat{\mathbf{E}}\right)^{-1}\cdot\hat{\mathbf{E}}^{\mathbf{T}}\cdot\mathbf{a}=\underbrace{\left(\hat{\mathbf{E}}^{\mathbf{T}}\cdot\hat{\mathbf{E}}\right)^{-1}\cdot\hat{\mathbf{E}}^{\mathbf{T}}\cdot\hat{\mathbf{E}}}_{=\mathbf{1}}\cdot\hat{\mathbf{c}}=\hat{\mathbf{c}}
$$

(31)

In conclusion, the calibration step (Chapter 2.2.1) uses calibration spectra and the corresponding calibration concentrations to estimate the matrix $\hat{\mathbf{E}}$ (26). $\hat{\mathbf{E}}$ is then utilized to predict the concentrations $\hat{\mathbf{c}}$ (31) of an unknown sample from the corresponding spectrum $\mathbf{a}$.

Equations (26) and (31) are very general types of equations encountered during MLR procedures. The origin of this equation type lies in the least-squares principle of minimizing the sum of squared errors.

A warning regarding (31) should be mentioned here: $\left(\hat{\mathbf{E}}^{\mathbf{T}}\cdot\hat{\mathbf{E}}\right)^{-1}$ needs to be computed, which only exists if all molar absorptivity spectra have different signatures and no one spectrum is a linear combination of the others. Otherwise, $\hat{\mathbf{E}}$ would be singular and the aforementioned inverse would not exist. This requirement is violated in situations where selectivity is insufficient, thereby prohibiting the parallel quantification of these analytes.

2.3 Principal Component Analysis / Principal Component Regression (PCA/PCR)

So far, calibration and evaluation algorithms have been discussed from a multivariate regression perspective. Since a spectrum consists of absorbance values measured at discrete wavelength positions a spectrum is represented by a vector. After expanding (12) to N wavelength positions and Q analytes, CLS expresses a spectrum $\mathbf{a}$ as the following linear combination of vectors $\mathbf{e}_1 \cdots \mathbf{e}_Q$:

$$\mathbf{a} = \begin{pmatrix} A_1 \\ \vdots \\ A_N \end{pmatrix} = \underbrace{L \cdot \begin{pmatrix} \varepsilon_{11} \\ \vdots \\ \varepsilon_{N1} \end{pmatrix}}_{=\mathbf{e}_1} \cdot c_1 + \cdots + \underbrace{L \cdot \begin{pmatrix} \varepsilon_{1Q} \\ \vdots \\ \varepsilon_{NQ} \end{pmatrix}}_{=\mathbf{e}_Q} \cdot c_Q = c_1 \cdot \mathbf{e}_1 + \cdots + c_Q \cdot \mathbf{e}_Q$$

(32)

While the molar absorptivity spectra $L \cdot \boldsymbol{\varepsilon}_{1 \cdot\cdot Q} = \mathbf{e}_{1 \cdot\cdot Q}$ determine the general shape of the absorbance spectrum, concentration values weigh these molar absorptivity spectra. This will be discussed from a linear algebra perspective in order to introduce PCR. The vectors $\mathbf{e}_{1 \cdot\cdot Q}$ are now interpreted as basis vectors spanning a $Q \leq N$-dimensional subvector space of R^N (usually: $Q << N$). Since Q independently changing analytes are present in the samples, Q basis vectors are sufficient to describe the entire system mathematically. If, for instance, the concentration c_1 is doubled, the absorbance contribution of analyte #1 at all N wavelengths also doubles. Thus, the absorbance values at all N wavelength positions are not independent. In (32) the concentrations $c_1 \cdots c_Q$ are the coordinates of $\mathbf{a}$ in the basis $\mathbf{e}_{1 \cdot\cdot Q}$.

CLS's requirement of knowing all analytes and all calibration concentrations becomes obvious. If something is not known, the molar absorptivity spectra are determined incorrectly in (26). If these basis vectors are incorrect, they are not equal to the expected physical properties of the analytes. Consequently, the coordinates of an unknown spectrum in this incorrect basis are not the concentrations of the wanted analytes in (31) and (32). Depending on the level of the disturbance, concentration results can range from being slightly off to being random numbers. In conclusion, the origin of distorted basis vectors $\mathbf{e}_{1\cdots Q}$ is that a wrong assumption about the number of present analytes (= number of basis vectors) has been made. If $\mathbf{e}_{1\cdots Q}$ are wrong, the concentration prediction in equation (31), (32) will be erroneous as well.

This is where PCA/PCR introduces a different approach. PCA/PCR first determines the rank $\mathrm{r}(\mathbf{A}) = R$ of the calibration spectra matrix $\mathbf{A}$ (a generalized version of the matrix introduced in (20)). The rank of a matrix is the number of linearly independent row vectors or column vectors [4]. In a further step, PCA/PCR extracts R basis vectors (typically $R \geq Q$), which will replace the $\mathbf{e}_{1\cdots Q} = L \cdot \boldsymbol{\varepsilon}_{1\cdots Q}$ in (26), (31) and (32). The superiority of PCA/PCR over CLS stems from the fact that PCA/PCR determines mathematically the appropriate number $R \geq Q$ of basis vectors and does not rely on a possibly incorrect user assumption about the number of present analytes. This user-independent approach ensures that the complete spectroscopic information contained in the calibration spectra is incorporated into the calibration model. Hence, unanticipated analytes will not lead to an incomplete calibration model and concentration errors. The price one has to pay for this is the loss of straightforward interpretability of the calibration model. The basis vectors $\mathbf{e}_{1\cdots Q}$ (32) will be replaced by $R \geq Q$ 'principal components' (PCs). The coordinates of a spectrum in the PC basis are called 'scores' which replace the concentrations $c_1 \cdots c_Q$ in (32). Nonetheless, the PCs model the spectroscopic features of the analytes contained in the calibration samples; however, the PCs comprise a mixture of features from different analytes. Consequently, the scores are linear combinations of the concentrations. At the end of the PCA/PCR calibration a 'translation' step from the PC coordinate system to the chemically meaningful properties, i.e. concentrations, is derived. This translation step remains valid during calibration and evaluation of unknown samples. If $R < Q$ is found during PCA/PCR calibration, at least one analyte has a spectrum that is either buried in noise or is a linear combination of the others. In such a situation, the calibration fails for CLS and PCA/PCR and a different experimental approach is required. This problem would not be detected by CLS.

A word of warning is required here to make the full extent of these techniques clear. CLS requires that calibration concentrations for all analytes in all calibration samples to be known. PCA/PCR can handle such situations but will fail if an unknown absorber emerges after finalizing PCA/PCR calibration. In that case, additional spectral features are contained in the spectra of unknown samples which cannot be described by the PCA/PCR calibration model because they are not contained in the PCs. It is mandatory for both CLS **and** PCA/PCR that **no** new absorbers are contained in the unknown samples **after** the calibration has been finalized.

2.3.1 PCA – Calibration

One common way to compute the PCs is to use a *singular value decomposition* (SVD) ([4], [38], [52], [53]) on the matrix $\mathbf{A}$ whose columns contain the K calibration spectra of length N. A SVD decomposes an arbitrary matrix into two orthogonal matrices $\mathbf{U}$ and $\mathbf{V}$ as well as a diagonal matrix $\mathbf{S} = \mathbf{diag}\left(s_1 \quad \cdots \quad s_K\right)$ such that:

$$
\begin{aligned}
\mathbf{A}_{(N\times K)} &= \mathbf{U}_{(N\times K)} \cdot \mathbf{S}_{(K\times K)} \cdot \mathbf{V}^{\mathbf{T}}_{(K\times K)} \\
&= \mathbf{U}_{(N\times R_{\text{PCA}})} \cdot \underbrace{\mathbf{S}_{(R_{\text{PCA}}\times R_{\text{PCA}})} \cdot \mathbf{V}^{\mathbf{T}}_{(R_{\text{PCA}}\times K)}}_{=\mathbf{T}^{\mathbf{T}}_{(R_{\text{PCA}}\times K)}} \\
&= \mathbf{U}_{(N\times R_{\text{PCA}})} \cdot \mathbf{T}^{\mathbf{T}}_{(R_{\text{PCA}}\times K)}
\end{aligned}
\qquad (\ 33\ )
$$

Usually the so-called singular values $s_{1\cdots K}$ are ordered decreasingly by convention. The columns of $\mathbf{U}$ and the rows of $\mathbf{V}^{\mathbf{T}}$ are ordered accordingly. If $\mathbf{A}$ does not have full rank, i.e. $\text{rank}\left(\mathbf{A}\right) = R_{\text{PCA}} < \min\left(N, K\right)$, only R_{PCA} singular values $s_{R_{\text{PCA}}<\min(N,K)} \neq 0$. If a singular value equal to zero is multiplied with the corresponding column of $\mathbf{U}$ or row of $\mathbf{V}^{\mathbf{T}}$, the resulting vector is zero. This singular value as well as the corresponding column of $\mathbf{U}$ or row of $\mathbf{V}^{\mathbf{T}}$ can be discarded without loss of information (second row in (33)). In the third row of (33), the columns of $\mathbf{U}$ contain the R_{PCA} relevant principal components (PCs) and the columns of $\mathbf{T}^{\mathbf{T}}$ (or the rows of $\mathbf{T}$) contain the R_{PCA} so-called scores of the K calibration spectra. The PCs replace the basis vectors $\mathbf{e}_{1\cdots Q}$ (32) and the scores $\mathbf{T}^{\mathbf{T}}$ replace the calibration concentrations $\mathbf{C}^{cal}$. Each column of $\mathbf{T}^{\mathbf{T}}$ corresponds to one calibration spectrum, i.e. a column of $\mathbf{A}$.

Because of random measurement noise, two spectra of the same sample measured back-to-back will never be identical. One consequence of noise is that in real-world applications there will be small singular values but none that are exactly zero. Thus, we need to determine a threshold below which a singular value can be *assumed* to be approximately equal to zero. Selecting the appropriate number of PCs is very important because ‘overfitting’ [53] must be avoided; however, this is a non-trivial task. Refer to the literature ([12], [13], [54]) for more detailed information. Thus, due to noise we have to consider the following equation instead of (33):

$$\begin{aligned}\mathbf{A}_{(N\times K)} &\approx \mathbf{U}_{(N\times R_{\text{PCA}})}\cdot \underbrace{\mathbf{S}_{(R_{\text{PCA}}\times R_{\text{PCA}})}\cdot \mathbf{V}^{\mathbf{T}}_{(R_{\text{PCA}}\times K)}}_{=\mathbf{T}^{\mathbf{T}}_{(R_{\text{PCA}}\times K)}} \\ &\approx \mathbf{U}_{(N\times R_{\text{PCA}})}\cdot \mathbf{T}^{\mathbf{T}}_{(R_{\text{PCA}}\times K)}\end{aligned} \tag{34}$$

Here, we apply an empirical approach that is found to be feasible in many real-world applications. The singular values $s_{1\cdots K}$ in (33) are ordered decreasingly. A value ≈ 0 represents irrelevant information which must be discarded. In this thesis, the ratio $s_1/s_{r=2\cdots K}$ serves as a figure of merit for finding the threshold. r is increased stepwise until the ratio crosses a user-defined value. For many of our applications, an empirical value of 1000 is often sufficient. R_{PCA} indicates the smallest singular value for which $s_1/s_r < 1000.$

The procedure just described is referred to as Principal Component Analysis (PCA). In order to predict concentrations from unknown spectra, PCA must be expanded to a Principal Component Regression (PCR). This is discussed in the next paragraph.

The R_{PCA} PCs (qualitative information) contain the entire relevant spectroscopic information contained in $\mathbf{A}$. The scores (quantitative information) represent how strong a PC contributes to the corresponding calibration spectrum. The PCs have the same length (or the same discrete wavelength positions) as the calibration spectra; the numbers contained in a PC are called loadings. The PCs and scores have no direct chemical/physical meaning but are linear combinations of all the chemical/physical properties contained in the calibration samples. Later, the unknown spectra will be expressed as a linear combination of the PCs. In other words, their coordinates (scores) in the PC space will be determined. In order to transform these scores into chemical information (=concentrations) we need to determine a 'translation' from scores to concentrations. This transform from scores to concentrations will never change and will be derived from the calibration information. The scores $\mathbf{T}^{\mathrm{T}}$ (34) of the calibration spectra and the calibration concentrations $\mathbf{C}^{cal}$ are known. Further, the calibration concentrations are linear combinations of the scores aside from measurement errors or other deviations $\boldsymbol{\varepsilon}$. A transform matrix $\hat{\mathbf{B}}$ has to be determined via MLR (compare (26) and (31)):

$$\mathbf{C}^{cal}_{(Q\times K)} = \mathbf{B}_{(Q\times R_{\mathrm{PCA}})} \cdot \mathbf{T}^{\mathrm{T}}_{(R_{\mathrm{PCA}}\times K)} + \boldsymbol{\varepsilon}$$
$$\mathbf{C}^{cal} \cdot \mathbf{T} \cdot \left(\mathbf{T}^{\mathrm{T}} \cdot \mathbf{T}\right)^{-1} = \hat{\mathbf{B}}$$

(35)

This finalizes the PCA/PCR calibration. The R_{PCA} PCs have been determined during calibration (33) which contain all spectroscopic information of the calibration spectra. A future unknown spectrum will be decomposed into these PCs. The scores, or the coordinates in the PC basis of this unknown spectrum, will then be translated into concentrations via the matrix $\hat{\mathbf{B}}$ (35).

Sometimes in the literature the calibration spectra are written in the *rows* of $\mathbf{A}$ (33). In this case, the previous discussion remains valid, only the meaning and dimensions of $\mathbf{U}$ and $\mathbf{V}^{\mathrm{T}}$ (33) are exchanged. The PCs are then contained in the rows of $\mathbf{V}^{\mathrm{T}}$ and the scores in the columns of $\mathbf{U}\cdot\mathbf{S}$. Refer to [51] for more details on this approach. Here, the calibration spectra have been written into the columns of $\mathbf{A}$ in order to remain consistent with the discussion of CLS (Chapter 2.2).

2.3.2 PCR – Evaluation of Unknown Samples

In CLS a spectrum is represented by a linear combination of concentration and molar absorptivity spectra (compare (32) and first line in (36)). In PCA/PCR an unknown spectrum is decomposed into PCs, the new basis vectors, and scores (second line in the following equation (36)). $\boldsymbol{\delta}$ and $\boldsymbol{\Delta}$ indicate measurement errors that cannot be modeled:

$$\begin{aligned}
\mathbf{a}_{(N\times 1)} &= \sum_{i=1}^{Q} \boldsymbol{\varepsilon}_i \cdot L \cdot c_i + \boldsymbol{\delta} \\
\mathbf{a}_{(Nx1)} &= \sum_{i=1}^{R} \mathbf{u}_i \cdot t_i + \boldsymbol{\Delta} \\
&= \mathbf{U}_{(N\times R_{\mathrm{PCA}})} \cdot \mathbf{t}_{(R_{\mathrm{PCA}}\times 1)} + \boldsymbol{\Delta}
\end{aligned}$$

(36)

In (36), the top equation represents the CLS model, which may not be correct if any unknown absorbers are present. The middle equation represents the PCA/PCR model, which will be correct despite any unknown information. Again, MLR (compare (26) and (31)) is used here to estimate $\mathbf{t}$ of the unknown spectrum $\mathbf{a}_{(Nx1)}$. The orthogonality of $\mathbf{U}$ simplifies the computations:

$$\hat{\mathbf{t}} = \underbrace{(\mathbf{U}^{\mathbf{T}} \cdot \mathbf{U})^{-1}}_{=\mathbf{1}} \cdot \mathbf{U}^{\mathbf{T}} \cdot \mathbf{a} = \mathbf{U}^{\mathbf{T}} \cdot \mathbf{a}$$

(37)

In a second step, the scores vector $\hat{\mathbf{t}}$ is linearly transformed into the concentrations $\mathbf{c}^{unknown}$ by using the result of (35):

$$\mathbf{c}^{unknown} = \hat{\mathbf{B}} \cdot \hat{\mathbf{t}}$$

(38)

This finalizes PCA/PCR evaluation. PCA/PCR is utilized in Chapter 3 as a tool to assess the capabilities of a new compression-based chemometric algorithm.

2.4 Kernel Principal Component Analysis (KPCA)

Common linear algorithms, such as PCA/PCR (Chapter 2.3), are frequently used to model chemical systems. These linear methods have been successful for countless applications but can be ill-suited for modeling nonlinear behavior within data. For the latter applications, a method named ‘kernel principal component analysis’ (KPCA) [39], [40] has been recently developed. KPCA has been successfully applied to a number of fields including nonlinear process monitoring for failure detection in waste water treatment plants [41] - [43], data denoising [44], recognition of handwritten digits [45], and classification of genetic data [46].

For the introduction of KPCA, it will be helpful to derive the PCs and scores (see (33) in Chapter 2.3) in a different way. Using the calibration spectra, a covariance matrix $\mathbf{C}$ will be computed which then undergoes an eigenvalue decomposition (EVD). This EVD can be realized via an SVD (see (33)):

$$\begin{aligned}
\mathbf{C}_{(N\times N)} &= \mathbf{A}^{\mathbf{T}}\cdot\mathbf{A} = \sum_{j=1}^{M}\mathbf{x}_j\cdot\mathbf{x}_j^{\mathbf{T}} \\
&= (\mathbf{U}\cdot\mathbf{S}\cdot\mathbf{V}^{\mathbf{T}})^{\mathbf{T}}\cdot(\mathbf{U}\cdot\mathbf{S}\cdot\mathbf{V}^{\mathbf{T}}) = \mathbf{V}\cdot\mathbf{S}^2\cdot\mathbf{V}^{\mathbf{T}} = \mathbf{V}\cdot\Lambda\cdot\mathbf{V}^{\mathbf{T}} \\
&\Downarrow \\
\mathbf{C}\cdot\mathbf{v}_i &= \lambda_i\cdot\mathbf{v}_i \qquad \text{with : } i=1,\cdots,N
\end{aligned} \tag{39}$$

After an EVD, the PCs are contained in the eigenvector matrix $\mathbf{V}$ and the eigenvalues, which are the squared singular values, are contained in the diagonal matrix $\Lambda = \mathbf{S}^2$. The scores are obtained by projecting the unknown measurement spectra onto the PCs. Calculating the scores for KPCA will be discussed in more detail in Chapter 2.4.2.

2.4.1 KPCA – Calibration

The KPCA algorithm is briefly concluded in this section. References [39] and [40] provide a more detailed discussion. If the reader is only interested in a basic summary of the KPCA algorithm and wants to skip some of the lengthy derivations, then proceed directly to the summary paragraph at the end of this section and then to Chapter 2.4.2. However, some important details describing the computational difficulties associated with KPCA are discussed below. These computational burdens are the main focus of the research presented in Chapter 4. The compression algorithm established in Chapter 4 provides a means to handle these demanding computations, thus making KPCA feasible on common workstations [2].

From a linear algebra perspective, spectra $\mathbf{x}_{(N\times 1)}$ are vectors in an N-dimensional vector space X. Since the chemical system from which spectra are acquired typically generates a certain number of spectroscopic signatures, X is usually a sub-vector space of R^N, i.e. the space of N-dimensional real vectors. The first step of KPCA is to non-linearly transform all M spectra into vectors of dimension $N_H >> N$. After applying this non-linear transform Φ (or mapping function or kernel), vectors $\Phi(\mathbf{x}_1),\cdots,\Phi(\mathbf{x}_M)$ are members of the so-called 'feature (vector) space' H which is a $R \leq M$-dimensional sub-vector space of R^{N_H} (compare to (34) but note that $R \neq R_{\mathrm{PCA}}$):

$$\Phi : X \subset R^N \to H \subset R^{N_H}, \quad \mathbf{x} \mapsto \Phi(\mathbf{x})$$

(40)

If, for example, a 'polynomial kernel' of order $d = 2$ is applied to a spectrum $\mathbf{x} = (x_1 \quad x_2)^T$ containing absorbance values of two wavelengths, (40) becomes:

$$\Phi : X \subset R^2 \to H \subset R^3, \quad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto \Phi(\mathbf{x}) = \begin{pmatrix} x_1^2 \\ x_2^2 \\ 2 \cdot x_1 \cdot x_2 \end{pmatrix}$$

(41)

For the investigations in this study, mainly the polynomial kernel (41) of user-specified order d is used. Using $d = 1$ makes KPCA equivalent to the conventional linear PCA algorithm.

For $N >> 2$ and higher polynomial orders d, the explicit mapping Φ of data into H introduces serious computational challenges because N_H becomes extremely large. For polynomial kernels, the dimension of the feature space H has the dimension [40]:

$$N_H = \frac{(d + N - 1)!}{d! \cdot (N - 1)!}$$

(42)

For example, if a spectrum $\mathbf{x}$ contains $N = 1000$ wavelength positions and a polynomial of degree $d = 4$ is applied, the feature space H has a dimension of $N_H \sim 4 \cdot 10^{10}$. Storing one vector $\Phi(\mathbf{x})$ in double precision would require ~300 GB. Thus, at all steps of the KPCA algorithm it is mandatory to avoid explicit handling of vectors in feature space. Consequently, a PCA in feature space, similar to the approach in (33), is not feasible and an alternative formalism has to be utilized. For this purpose, a covariance matrix $\mathbf{C}^H_{(N_H \times N_H)}$ similar to the one in (39) can *formally* be defined by:

$$\mathbf{C}^H_{(N_H \times N_H)} = \sum_{j=1}^{M} \Phi(\mathbf{x}_j) \cdot \Phi(\mathbf{x}_j)^{\mathbf{T}}$$
$$\Downarrow$$
$$\mathbf{C}^H \cdot \mathbf{v}_i = \lambda_i \cdot \mathbf{v}_i \qquad \text{with: } i = 1, \cdots, N_H \quad \text{but} \quad \lambda_i = 0 \quad \text{for} \quad i > M$$

(43)

The dimensions (42) of $\mathbf{C}^H_{(N_H \times N_H)}$ (43) prohibit direct computation of all eigenvectors $\mathbf{v}_i$ and eigenvalues λ_i; however, a new method is presented in Chapter 4 that can handle such situations. Nonetheless, the vectors $\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_M)$ span the feature space H (40) and since all $i = 1, \cdots, M$ eigenvectors $\mathbf{v}_i$ (43) are members of H, there exists a set of M coefficients $\alpha_{i=1,\ldots,M,\ k=1,\ldots,M}$ such that:

$$\mathbf{v}_i = \sum_{k=1}^{M} \alpha_{i,k} \cdot \Phi(\mathbf{x}_k)$$

(44)

In other words, the eigenvectors, which will now be determined, are linear combinations of the calibration spectra in feature space. In the following discussion, the coefficients $\alpha_{i=1,\ldots,M,\ k=1,\ldots,M}$ will be determined. Using (44) in the second line of (43) results in:

$$\mathbf{C}^H \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot \Phi(\mathbf{x}_k) = \lambda_i \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot \Phi(\mathbf{x}_k)$$
$$\sum_{j=1}^{M} \Phi(\mathbf{x}_j) \cdot \Phi(\mathbf{x}_j)^{\mathbf{T}} \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot \Phi(\mathbf{x}_k) = \lambda_i \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot \Phi(\mathbf{x}_k)$$

(45)

In the next step, all M vectors $\Phi(\mathbf{x}_{1 \cdots M})^{\mathbf{T}}$ are consecutively multiplied from the left to the second row of equation (45). This results in the M equations:

$$\Phi(\mathbf{x}_1)^{\mathbf{T}} \cdot \left(\sum_{j=1}^{M} \Phi(\mathbf{x}_j) \Phi(\mathbf{x}_j)^{\mathbf{T}} \right) \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot \Phi(\mathbf{x}_k) = \lambda_i \cdot \Phi(\mathbf{x}_1)^{\mathbf{T}} \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot \Phi(\mathbf{x}_k)$$
$$\vdots$$
$$\Phi(\mathbf{x}_M)^{\mathbf{T}} \cdot \left(\sum_{j=1}^{M} \Phi(\mathbf{x}_j) \Phi(\mathbf{x}_j)^{\mathbf{T}} \right) \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot \Phi(\mathbf{x}_k) = \lambda_i \cdot \Phi(\mathbf{x}_M)^{\mathbf{T}} \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot \Phi(\mathbf{x}_k)$$

Rearranging the summations over k and utilizing dot products results in:

$$\sum_{k=1}^{M} \alpha_{i,k} \cdot \sum_{j=1}^{M} \langle \Phi(\mathbf{x}_1), \Phi(\mathbf{x}_j) \rangle \cdot \langle \Phi(\mathbf{x}_j), \Phi(\mathbf{x}_k) \rangle = \lambda_i \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot \langle \Phi(\mathbf{x}_1), \Phi(\mathbf{x}_k) \rangle$$
$$\vdots$$
$$\sum_{k=1}^{M} \alpha_{i,k} \cdot \sum_{j=1}^{M} \langle \Phi(\mathbf{x}_M), \Phi(\mathbf{x}_j) \rangle \cdot \langle \Phi(\mathbf{x}_j), \Phi(\mathbf{x}_k) \rangle = \lambda_i \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot \langle \Phi(\mathbf{x}_M), \Phi(\mathbf{x}_k) \rangle$$

(46)

Now, the so-called Gram matrix $\mathbf{K}_{(M \times M)}$ (Figure 17) is defined which contains the elements:

$$K_{p,q} = \langle \Phi(\mathbf{x}_p), \Phi(\mathbf{x}_q) \rangle$$

(47)

In the next equation, it will be shown that M column vectors $\boldsymbol{\alpha}_{i=1,\cdots,M}$, which contain the coefficients $\alpha_{i=1,..,M,\ k=1,..,M}$, are eigenvectors of $\mathbf{K}_{(M \times M)}$. Substituting this definition along with (47) into (46) derives $(i = 1,\cdots,M)$:

$$\sum_{k=1}^{M} \alpha_{i,k} \cdot \sum_{j=1}^{M} K_{1,j} \cdot K_{j,k} = \lambda_i \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot K_{1,k}$$
$$\vdots$$
$$\sum_{k=1}^{M} \alpha_{i,k} \cdot \sum_{j=1}^{M} K_{M,j} \cdot K_{j,k} = \lambda_i \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot K_{M,k}$$
$$\Downarrow$$
$$\mathbf{K} \cdot \mathbf{K} \cdot \boldsymbol{\alpha}_i = \lambda_i \cdot \mathbf{K} \cdot \boldsymbol{\alpha}_i$$
$$\mathbf{K} \cdot \boldsymbol{\alpha}_i = \lambda_i \cdot \boldsymbol{\alpha}_i$$

(48)

Solving the eigenvalue problem (48) for all $i = 1,\cdots,M$ results in eigenvalues $\lambda_{1,\cdots,M}$ and eigenvectors $\boldsymbol{\alpha}_{1,..,M}$ of the Gram matrix $\mathbf{K}$. The latter, $\boldsymbol{\alpha}_{1,..,M}$, will be used to determine the eigenvectors $\mathbf{v}_{i=1,\cdots,M}$ (44) of the covariance matrix $\mathbf{C}^H$. The $\mathbf{v}_{i=1,\cdots,M}$ will then play the role of principal components as in PCA (see (44)). Typically, there are only $R < M$ relevant eigenvalues. The remaining eigenvalues are so close to zero that they are negligible from a spectroscopic perspective. Thus, only $\lambda_{1,\cdots,R}$ and $\boldsymbol{\alpha}_{1,\cdots,R}$ are kept. In a last step, the vectors $\mathbf{v}_{i=1,\cdots,R}$ are normalized to Euclidean length of one by dividing each $\boldsymbol{\alpha}_i$ by $\sqrt{\lambda_i}$:

$$
\begin{aligned}
1 &= \langle \mathbf{v}_i, \mathbf{v}_i \rangle \quad \text{for all} \quad i = 1, \cdots, R \\
&= \left\langle \sum_{k=1}^{M} \alpha_{i,k} \cdot \Phi(\mathbf{x}_k), \sum_{j=1}^{M} \alpha_{i,j} \cdot \Phi(\mathbf{x}_j) \right\rangle = \sum_{k=1}^{M} \alpha_{i,k} \cdot \sum_{j=1}^{M} \alpha_{i,j} \cdot \langle \Phi(\mathbf{x}_k), \Phi(\mathbf{x}_j) \rangle \\
&= \sum_{k=1}^{M} \sum_{j=1}^{M} \alpha_{i,k} \cdot K_{k,j} \cdot \alpha_{i,j} = \langle \boldsymbol{\alpha}_i, \mathbf{K} \cdot \boldsymbol{\alpha}_i \rangle = \langle \boldsymbol{\alpha}_i, \lambda_i \cdot \boldsymbol{\alpha}_i \rangle \\
&= \left\langle \sqrt{\lambda_i} \cdot \boldsymbol{\alpha}_i, \sqrt{\lambda_i} \cdot \boldsymbol{\alpha}_i \right\rangle
\end{aligned}
$$

(49)

Since the $\boldsymbol{\alpha}_{1,\cdots,R}$ are eigenvectors of the symmetric Gram matrix they are orthogonal. Similarly, the $\mathbf{v}_i$ are eigenvectors of the covariance matrix $\mathbf{C}^H$ and after dividing all $\boldsymbol{\alpha}_i$ by $\sqrt{\lambda_i}$ the $\mathbf{v}_i$ will be an orthonormal basis of the vector space H (40).

However, even if the Gram matrix $\mathbf{K}(M \times M)$ would have dimensions small enough to store the matrix in a workstation's memory, it can often not be generated directly. The reason for this is that each matrix element is a dot product of two vectors of length N_H (42). In order to circumvent unreasonable computation times, the so-called "kernel trick" [39], [40] is applied. It can be shown that for some kernels dot products in the high-dimensional feature space can be determined based on the dot product in the much lower-dimensional input space. This approach is much more computationally efficient. For instance, for polynomial kernels of order d (41):

$$\langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle = \langle \mathbf{x}, \mathbf{y} \rangle^d$$

(50)

In summary, the KPCA algorithm first calculates the Gram matrix (47), (48) which is then decomposed into eigenvalues $\lambda_{1,\cdots,R}$ and eigenvectors $\boldsymbol{\alpha}_{1,\cdots,R}$ (48). From

$\boldsymbol{\alpha}_{1,\cdots,R}$, vectors $\mathbf{v}_{1,\cdots,R}$ (44), (49) are derived that are KPCA's equivalent to PCA's principal components.

2.4.2 KPCA – Evaluation of Unknown Samples

In order to evaluate an unknown spectrum in feature space, a vector $\mathbf{t} = (t_1 \quad \cdots \quad t_R)^{\mathbf{T}}$ is determined. This $\mathbf{t}$-vector is the equivalent to PCA's score vector of unknown samples (compare to (37)). For instance, these $\mathbf{t}$-vectors represent how strong a (KPCA) PC (44) contributes to the corresponding calibration spectra. For this purpose, an unknown data vector $\Phi(\mathbf{x}_{\text{unknown}})$ is projected (51) onto the vectors $\mathbf{v}_{i=1,\cdots,R}$ (44). The same kernel trick (50) helps, again, to keep computation efforts reasonable:

$$\begin{aligned} t_{i=1,\ldots,R} &= \left\langle \Phi(\mathbf{x}_{\text{unknown}}), \mathbf{v}_i \right\rangle = \frac{1}{\sqrt{\lambda_i}} \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot \left\langle \Phi(\mathbf{x}_{\text{unknown}}), \Phi(\mathbf{x}_k) \right\rangle \\ &= \frac{1}{\sqrt{\lambda_i}} \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot \left\langle \mathbf{x}_{\text{unknown}}, \mathbf{x}_k \right\rangle^d \qquad \text{for polynomial kernels of order } d \\ &= \frac{1}{\sqrt{\lambda_i}} \cdot \sum_{k=1}^{M} \alpha_{i,k} \cdot K_{q,k}^{\text{unknown}} \qquad \text{with}: q = 1, \cdots, Q \text{ unknown data vectors,} \\ & \qquad \qquad \qquad \qquad \qquad \text{compare } K_{q,k}^{\text{unknown}} \text{ with equ. (47)} \end{aligned} \tag{51}$$

If Q unknown data vectors $\Phi(\mathbf{x}_{\text{unknown}})_{1,\cdots,Q}$ are evaluated, (51) has to be expanded, as shown in this paragraph. For data evaluation, the R score values of each of the Q unknown data vectors can be combined to a *matrix* $\mathbf{T}_{(Q\times R)}$. Thus, each row of $\mathbf{T}_{(Q\times R)}$ is derived from one of the unknown samples:

$$\mathbf{T}_{(Q\times R)} = \begin{pmatrix} \left\langle \Phi(\mathbf{x}_{\text{unknown}})_1, \mathbf{v}_1 \right\rangle & \cdots & \left\langle \Phi(\mathbf{x}_{\text{unknown}})_1, \mathbf{v}_R \right\rangle \\ \vdots & \ddots & \vdots \\ \left\langle \Phi(\mathbf{x}_{\text{unknown}})_Q, \mathbf{v}_1 \right\rangle & \cdots & \left\langle \Phi(\mathbf{x}_{\text{unknown}})_Q, \mathbf{v}_R \right\rangle \end{pmatrix}_{(Q\times R)}$$

$$
\begin{aligned}
&= \begin{pmatrix}
\frac{1}{\sqrt{\lambda_1}}\cdot\sum_{k=1}^{M}\alpha_{1,k}\cdot\left\langle\Phi(\mathbf{x}_{\text{unknown},1}),\Phi(\mathbf{x}_k)\right\rangle & \cdots & \frac{1}{\sqrt{\lambda_R}}\cdot\sum_{k=1}^{M}\alpha_{R,k}\cdot\left\langle\Phi(\mathbf{x}_{\text{unknown},1}),\Phi(\mathbf{x}_k)\right\rangle \\
\vdots & \ddots & \vdots \\
\frac{1}{\sqrt{\lambda_1}}\cdot\sum_{k=1}^{M}\alpha_{1,k}\cdot\left\langle\Phi(\mathbf{x}_{\text{unknown},Q}),\Phi(\mathbf{x}_k)\right\rangle & \cdots & \frac{1}{\sqrt{\lambda_R}}\cdot\sum_{k=1}^{M}\alpha_{R,k}\cdot\left\langle\Phi(\mathbf{x}_{\text{unknown},Q}),\Phi(\mathbf{x}_k)\right\rangle
\end{pmatrix}_{(Q\times R)} \\
&= \begin{pmatrix}
\frac{1}{\sqrt{\lambda_1}}\cdot\sum_{k=1}^{M}\alpha_{1,k}\cdot K_{1,k}^{\text{unknown}} & \cdots & \frac{1}{\sqrt{\lambda_R}}\cdot\sum_{k=1}^{M}\alpha_{R,k}\cdot K_{1,k}^{\text{unknown}} \\
\vdots & \ddots & \vdots \\
\frac{1}{\sqrt{\lambda_1}}\cdot\sum_{k=1}^{M}\alpha_{1,k}\cdot K_{Q,k}^{\text{unknown}} & \cdots & \frac{1}{\sqrt{\lambda_R}}\cdot\sum_{k=1}^{M}\alpha_{R,k}\cdot K_{Q,k}^{\text{unknown}}
\end{pmatrix}_{(Q\times R)} \\
&= \begin{pmatrix}
K_{1,1}^{\text{unknown}} & \cdots & K_{1,M}^{\text{unknown}} \\
\vdots & \ddots & \vdots \\
K_{Q,1}^{\text{unknown}} & \cdots & K_{Q,M}^{\text{unknown}}
\end{pmatrix}_{(Q\times M)}
\cdot
\begin{pmatrix}
\frac{1}{\sqrt{\lambda_1}}\cdot\alpha_{1,1} & \cdots & \frac{1}{\sqrt{\lambda_R}}\cdot\alpha_{R,1} \\
\vdots & \ddots & \vdots \\
\frac{1}{\sqrt{\lambda_1}}\cdot\alpha_{1,M} & \cdots & \frac{1}{\sqrt{\lambda_R}}\cdot\alpha_{R,M}
\end{pmatrix}_{(M\times R)} \\
&= \mathbf{K}_{(Q\times M)}^{\text{unknown}}\cdot\begin{pmatrix}
\frac{1}{\sqrt{\lambda_1}}\cdot\boldsymbol{\alpha}_1 & \cdots & \frac{1}{\sqrt{\lambda_R}}\cdot\boldsymbol{\alpha}_R
\end{pmatrix}_{(M\times R)}
\end{aligned}
\tag{52}
$$

Concluding the evaluation step, t-values (KPCA), which are equivalent to PCA's scores, of one (51) or several unknown samples (52) are determined by projecting unknown data vectors in the feature space, $\Phi(\mathbf{x}_{\text{unknown}})$, onto KPCA's principal components $\mathbf{v}_{1,\cdots,R}$ (44).

2.5 Wavelet Transforms (WTs)

Throughout the following two chapters wavelet transforms (WTs) are used as a tool to accelerate (Chapter 3) and make feasible (see Chapter 4) chemometric calculations to analyze spectroscopic imaging data. WTs have gained substantial interest in recent years in the field of analytical chemistry by providing a means for denoising and compressing data [1], [2], [23], [27] - [30], [32] - [34]. This section is intended to offer a qualitative overview on the concept of a WT and give insight on the unique characteristics of this efficient form of data representation. Refer to [20] for a general description, or first read, of WTs. For a more detailed discussion, refer to [18], [19], [22].

To begin explaining the advantages of a WT, a comparison is made to Fourier Transform (FT). FT decomposes a signal into sine and cosines waves, thus decomposing a signal into its frequency components (Fourier coefficients). Each coefficient is specifically derived from a single frequency and provides information about the contribution of this frequency to the overall signal. No time localization information is obtained from a signal during FT. Since most signals contain some degree of localization in time *and* frequency, a certain set of basis functions is required that is well localized in both the time and frequency domains in order to represent the signal efficiently. This is where the advantage of a WT becomes apparent. A WT uses wavelets, $\psi_{a,b}(x)$ (also see Figure 4), which are well localized in both time and frequency, as its basis functions to decompose a signal $f(x)$ into its corresponding wavelet coefficients $c_{a,b}$:

$$c_{a,b} = \int_{-\infty}^{\infty} f(x)\,\psi_{a,b}(x)\,\mathrm{d}x = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} f(x)\,\psi\left(\frac{x-b}{a}\right)\mathrm{d}x \tag{53}$$

a and b in (53) describe the scale (resolution or width) and translation (location in time) of the wavelet, respectively.

There are many different types of wavelets; however, the research outlined in this thesis makes use of the popular Daubechies family wavelets [19]. Daubechies wavelets decompose a signal perfectly; i.e. no loss (or addition) of information is introduced during the transform. Figure 4 displays a few examples of some the various Daubechies family wavelets. Each wavelet contains its own individual mathematical properties. This characteristic is addressed in more detail in Chapter 3. Other than the Haar wavelet (Figure 4 (a)), the remanding Daubechies wavelets are simply referred to as Daub4, Daub8, etc.

The unique feature of wavelets is that they are ≠ 0 only within a small window known as the ‘support’. During a WT, the scale (width) of the support for a particular wavelet is adjusted, via a (see (53)), and translated over the entire signal, via b (see

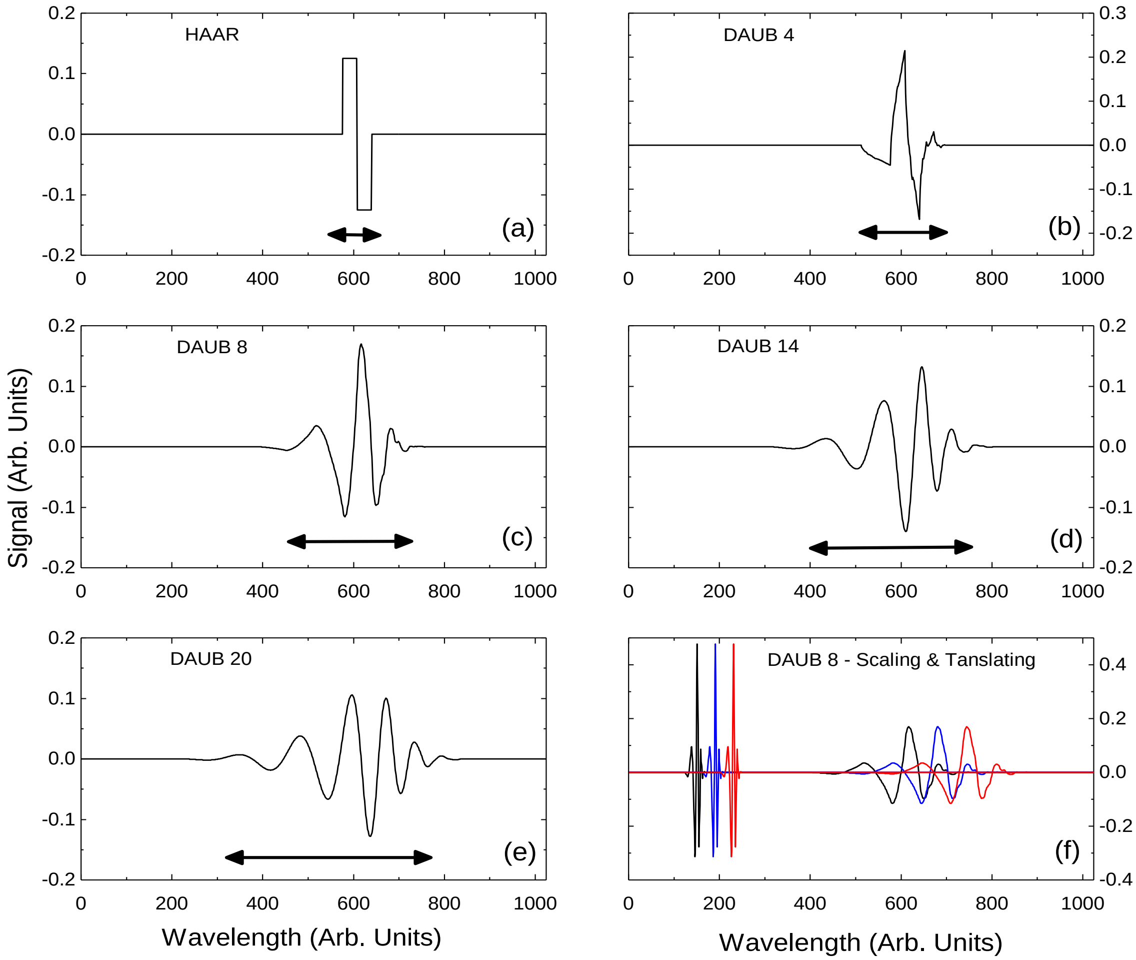


Figure 4: Examples of various Daubechies family wavelets. (a) Haar, (b) Daub4, (c) Daub8, (d) Daub14, and (e) Daub20 wavelets are shown, respectively. Each wavelet has its own mathematical properties. (f) The Daub8 wavelet represented at different scales (width) and translations (also see (53)).

(53)). In other words, once a scale is set it is held constant while it is translated over the entire signal. This process is continued until all wavelet coefficients (53) are calculated. Only the spectral features present inside the support (see (53)) impact the wavelet coefficients. Since a wavelet is equal to zero everywhere outside of the support, all spectral features outside of the support are set equal to zero and contribute nothing to the corresponding wavelet coefficients. Figure 4 (f) displays different widths and translations of the Daub8 wavelet. Scaling and translating wavelets over a signal acts as a filter. Information is gathered about the high and low frequency signal components when narrow and wide wavelets are applied, respectively. The next paragraph, along with Figure 5, provides an example to illustrate, in more detail, the scaling/translating process of a WT.

Utilizing a simulated spectrum (Figure 5, top left) of 512 wavelength positions a WT is carried out by first scanning over this spectrum with the highest scale (narrowest) of the applied wavelet. In this example, a Daub8 wavelet (Figure 4 (c) and (f)) is used. As the wavelet scale widens, the number of wavelet coefficients calculated is down-sampled by a factor of 2. For instance, the first scanning step will yield 512 / 2 = 256 wavelet coefficients (refer to Figure 5, top right and bottom left). These coefficients characterize the high frequency components of the spectrum (i.e. mostly noise). Next, the subsequent wavelet scale (refer to Figure 5, bottom right) is translated over the spectrum producing the next 256 / 2 = 128 wavelet coefficients. This procedure is continued until the widest wavelet scale is used and the final wavelet coefficients are calculated, thus resulting in a WT spectrum (refer to Figure 5, bottom left). Again, this WT spectrum (Figure 5, bottom left) is completely equivalent to the original simulated spectrum (Figure 5, top left). The large positive and negative values of the lower order wavelet coefficients in the WT spectrum clearly indicate their greater significance (relevance) over the other coefficients. A large portion of the higher order coefficients are $\cong$ 0. Therefore, setting the small valued wavelet coefficients equal to 0 will essentially ‘denoise’ a signal (or spectrum). Completely removing these ‘irrelevant’ coefficients will compress the original spectrum. An example of signal compression and denoising using WTs is described next.

The compression and denoising capabilities of WTs are depicted in Figure 6. Using the simulated spectrum (Figure 5 and Figure 6 (a)), a WT is applied using a Daub8 wavelet resulting in the WT spectrum in Figure 5 and Figure 6 (b). Several

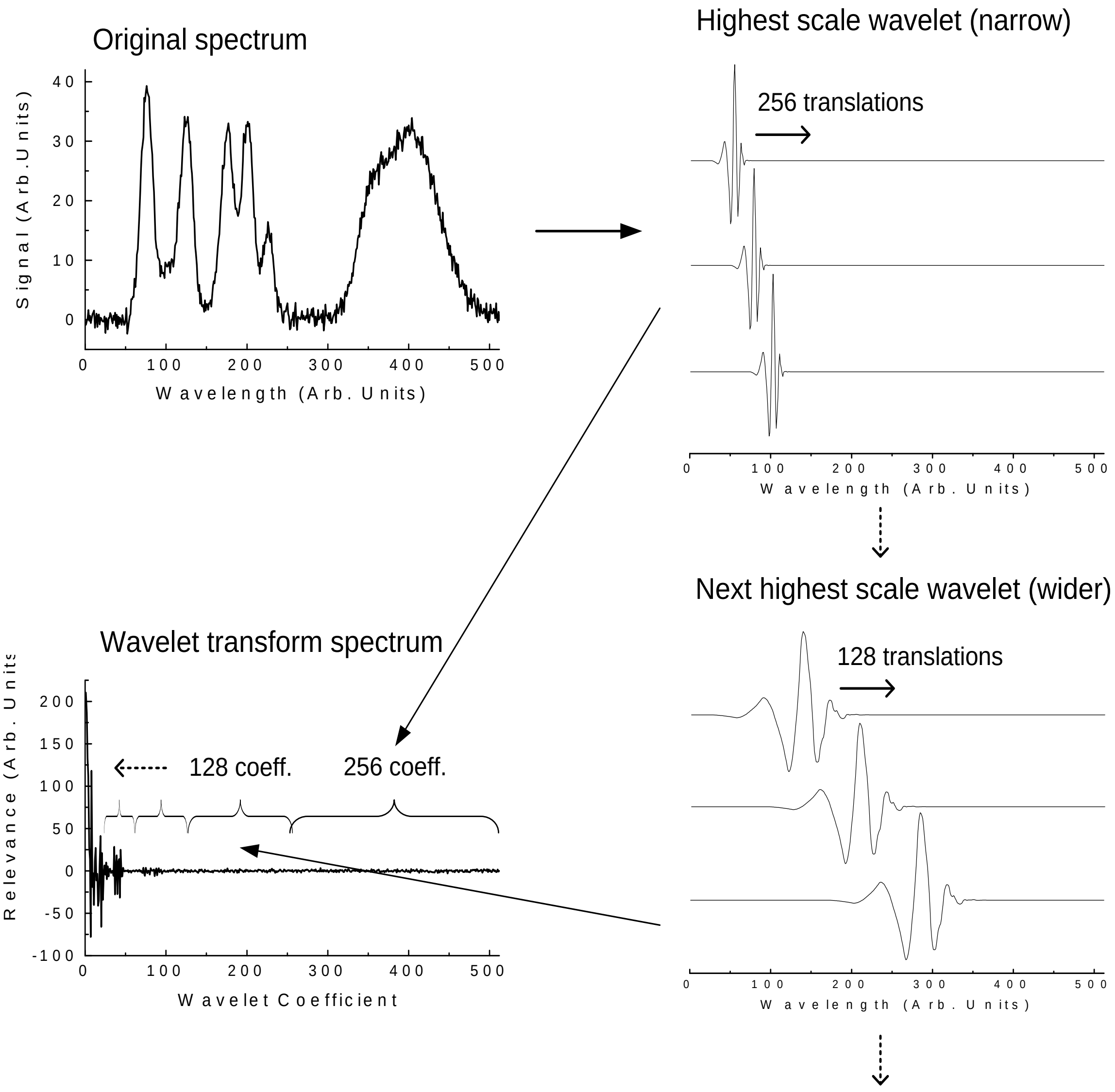


Figure 5: Cartoon illustrating the main steps of a WT (53). The spectrum (top left, generated data) is scanned by a narrow Daub8 wavelet (top right) at 512 / 2 = 256 different positions to yield 256 wavelet coefficients (bottom left). These coefficients characterize the high frequency components (mostly noise) of the spectrum. The next widest wavelet (bottom right) is scanned over the same spectrum at 256 / 2 = 128 different positions to give the next set of (128) wavelet coefficients. This process is continued for the remaining wavelets until all coefficients are obtained, concluding the WT (bottom left).

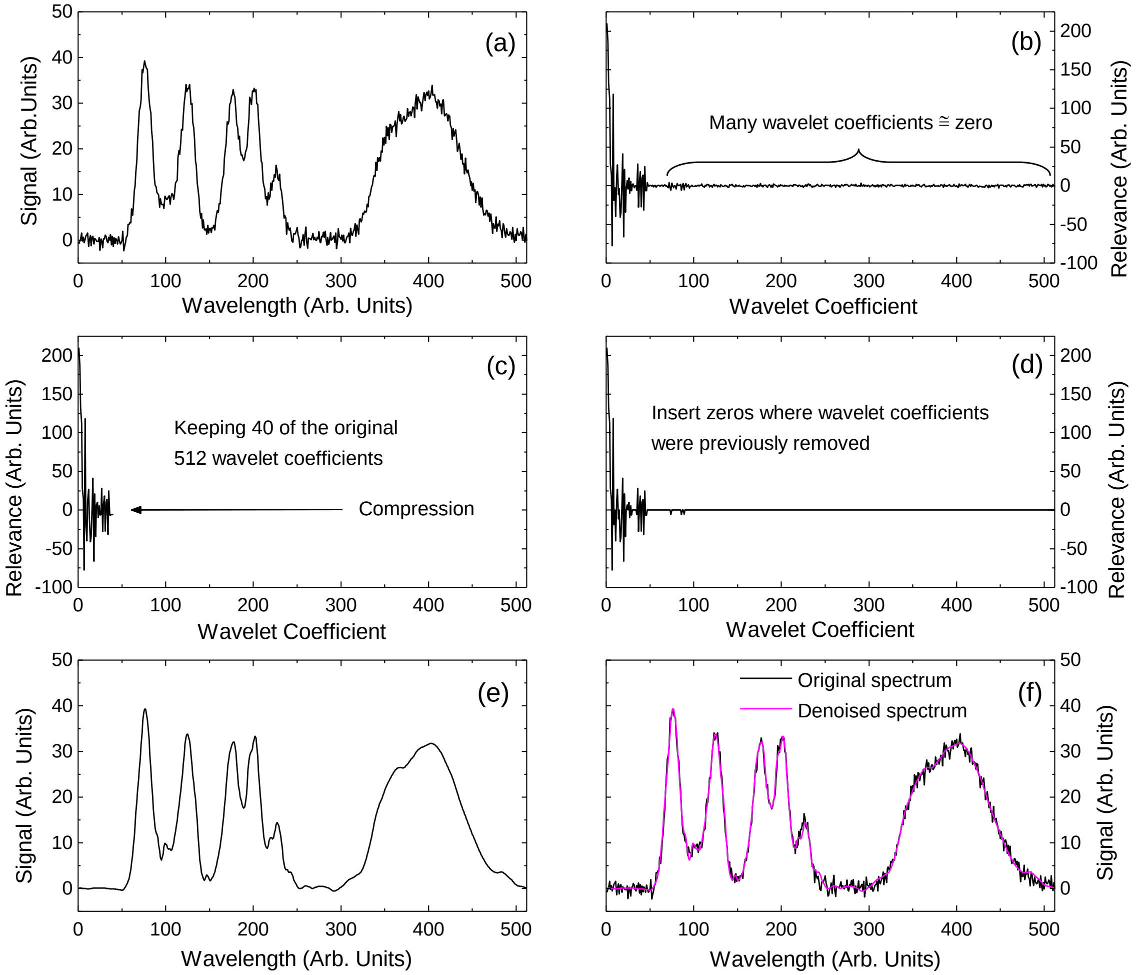


Figure 6: (a) The original signal (or spectrum) in the original measurement domain (simulated data). (b) The same signal transformed into the wavelet domain using a Daub8 wavelet. Both representations of the signal in (a) and (b) are equivalent; i.e. no information is lost (or added) during the transform. (c) The compressed version of the signal in (b). After thresholding, only 40 of the original 512 wavelet coefficients are retained. At this point, chemometrics can be applied to these compressed data to greatly accelerate computations. (d) Before reconstruction (via iWT) of the compressed signal in (c), zeros are inserted into the locations where wavelet coefficients were previously removed. (e) The denoised spectrum is calculated after performing an iWT on the signal in (d). (f) The denoised spectrum (e) overlaid on the original spectrum (a). The main 'spectral' features are retained in the denoised spectrum; however, the noise has successfully been removed.

wavelet coefficients in this WT spectrum (Figure 6 (b)) are $\cong$ 0. Since large positive and negative wavelet coefficients are more relevant than coefficients that have a small value, a threshold range can be established and used to eliminate any coefficients falling within this range. In this example, an empirical range of ± 5 'relevance units' is used to distinguish between relevant and irrelevant wavelet coefficients. More sophisticated approaches are discussed in Chapters 3 and 4. After applying this threshold only 40 of the original 512 wavelet coefficients are considered 'relevant'. This compressed wavelet spectrum (shown in Figure 6 (c)) is < 8% of its original length. At this point, chemometrics can be applied to these compressed data, thus achieving large increases in computation speed. This topic is discussed in more detail throughout Chapters 3 and 4. For denoising purposes, zeros can be inserted into the exact locations where 'irrelevant' wavelet coefficients were previously removed (compare Figure 6 (b) and Figure 6 (d)). In order to obtain the denoised, reconstructed spectrum in Figure 6 (e), an inverse wavelet transform (iWT) needs to be applied to the spectrum in Figure 6 (d). After the iWT is applied, the resulting denoised signal is obtained. If all Daubechies wavelet coefficients are kept during compression/denoising, the denoised signal would exactly equal $f(x)$ after the iWT. Also, the same wavelet *must* be applied during the iWT that was applied during the original WT. If not, the results will be meaningless. Figure 6 (f) compares the denoised spectrum (Figure 6 (e)) to the original noisy spectrum (Figure 6 (a)). In the denoised spectrum, the noise has been successfully removed and the main 'spectral' features have been preserved. The example above demonstrates that WTs are capable of modeling the relevant information within a signal using only a small number of wavelet coefficients. This is not possible with FT.

So far, only 1-dimensional (1D) WTs have been discussed. It has been shown [35] - [37], [55] that 1D WTs can easily be expanded to multi-dimensional WTs to enhance compression of multi-dimensional data sets (i.e. spectroscopic data cubes). Multi-dimensional WTs will be discussed to a greater extent and applied in Chapters 3 and 4 [1], [2].

Chapter 3

Composing Hybrid Wavelets for Enhanced Evaluation of N-dimensional Data Sets

Each dimension of a multi-dimensional data set contains different types of information. For example, a 3D data cube acquired by means of spectroscopic imaging contains two spatial dimensions and one spectral dimension (Figure 1). From a mathematical perspective, different wavelets, whichever is best, can be used to transform the different dimensions of this data set as long as the same wavelet is used during the inverse transform (see Chapter 2.5). In reference [37], 'hybrid wavelets' have been introduced which utilize combinations of different wavelets. This enhances acceleration and reduces approximation errors when different types of data are contained in multi-dimensional data cubes. However, there are many different wavelet types which lead to several possible wavelet combinations. Therefore, an important question has to be addressed: which combination of wavelets is optimum for a specific data set? This selection has to be made without *a priori* information and without adding excessive computational burden in order not to jeopardize the acceleration.

In this study, a pool of 10 different Daubechies wavelets are used, i.e. Haar, Daub4, Daub6, Daub8, …, Daub18 and Daub20 [19] (see Chapter 2.5 and Figure 4). Because spectroscopic imaging generates 3D data sets (Figure 1) there are 10 different wavelet options for each dimension. Thus, for this group of wavelets, there are 10 x 10 x 10 = 1000 possible wavelet combinations which can be used to transform the X-Y-Z dimensions of a data cube. In the absence of compression, all possible wavelet combinations provide perfect *and* reconstructable data representation. However, when compression is implemented, not all of these combinations provide an equally acceptable representation of the data [1], [37]. This chapter presents an automated compression method that selects optimum and near-optimum hybrid wavelet combinations for N-dimensional data sets. Since accelerating overall calculation times is highly desired, the selection is made fast enough to avoid burdensome calculations.

The computational methods discussed in the following chapter have been developed and implemented into C++ source code. All of the results presented in Chapter 3.3 were obtained using this code. All calculations were performed under Windows on a 32-bit Intel machine with 4 GB of RAM utilizing Microsoft Visual C++.

3.1 Theory

This section describes how we build optimum 3D hybrid-WTs for 3D data sets [1]. The wavelet selection is based on three figures of merit: acceleration of the chemometric computations, quality (accuracy) of the resulting model, and size of the compressed data cubes. In previous studies [36], [37], [55], it has been shown that shorter wavelets (i.e. Haar, Daub4, Daub6, etc.) can result in higher acceleration factors because they require fewer computations for performing WTs and iWTs. So, if the algorithm finds several wavelet combinations of equal capability, the shortest wavelets are selected in order to optimize computation speed. However, a decent amount of acceleration can be achieved even with the longest wavelets (i.e. Daub16, Daub18, Daub20). Therefore, the algorithm *explicitly* incorporates model accuracy and data set size. Calculation speed is gained as a result of performing chemometrics on compressed data sets *and* by utilizing shorter wavelets, whenever possible. In Chapter 3.3.2, an acceleration factor is defined to determine the amount of computation speed that is achieved as a result of utilizing the compression algorithm. The figures of merit that are *explicitly* incorporated into the compression algorithm are defined as:

1. **Accuracy** compares the closeness (or quality) of the chemometric model derived from the compressed data to its counterpart computed from the original measurement data. Such a comparison is only intended to assess the compression-based method and will not be performed in routine computations.
2. The **Size** of a compressed data set is important for accelerating chemometric calculations and for data storage requirements, especially when handling a continuous stream of incoming data, as encountered in online sensing. Smaller data sets yield faster computation times and require less storage space.

The selection of the best wavelet is performed subsequently for each dimension of a data (hyper)cube. In this study, the wavelet for the Z-dimension (see insert in Figure 7) is chosen first, then for the Y-dimension and finally for the X-dimension; however, the order does not matter since the transforms in different dimensions are independent from each other [52]. The automated hybrid wavelet compression is outlined in the following steps (see corresponding steps in Figure 7) followed by a more detailed discussion based on Figure 8. This algorithm can be applied to any data cube. The cube shown in the top left of Figure 7 is used only for visual purposes and is described further in Chapter 3.2.

(1) The fundamental idea is to randomly choose a small subset of test vectors in the Z-dimension (or XY plane; see insert in top right of Figure 7) of the data set. The size of this subset, labeled as *N*% of the total number of vectors in the Z-dimension, is discussed in Chapter 3.3.1. *N*% has to be as small as possible to keep computation efforts limited, but it must be large enough to ensure that the subset is *representative* of the entire Z-dimension. A copy of this subset is made for each wavelet type that is to be tested. Since ten Daubechies wavelets are included in this study, ten copies of the subset are made.

(2) The best wavelet type for the Z-dimension is chosen after analyzing the wavelets' performance on the subset regarding the aforementioned figures of merit; this step is outlined in more detail below and in Figure 8.

(3) The wavelet found to be optimum for this subset is then used to transform the Z-dimension of the original cube, vector by vector. Next, compression of the original cube in the Z-dimension is performed. The compression process is discussed further below.

(4) Steps 1 and 2 are repeated to find the best wavelet type for the Y-dimension (or XZ-plane).

> *Comment*: The Y-selection is done after Z-compression because it makes no sense to select test vectors in the Y-dimension that are removed after compression in the Z-dimension. These vectors are considered to be dispensable and thus are not representative of the Y-dimension. Also, the selection of the Y-wavelet is performed after compressing the Z-dimension since the numbers along the Z-axis of the data cube are altered due to wavelet transformation.

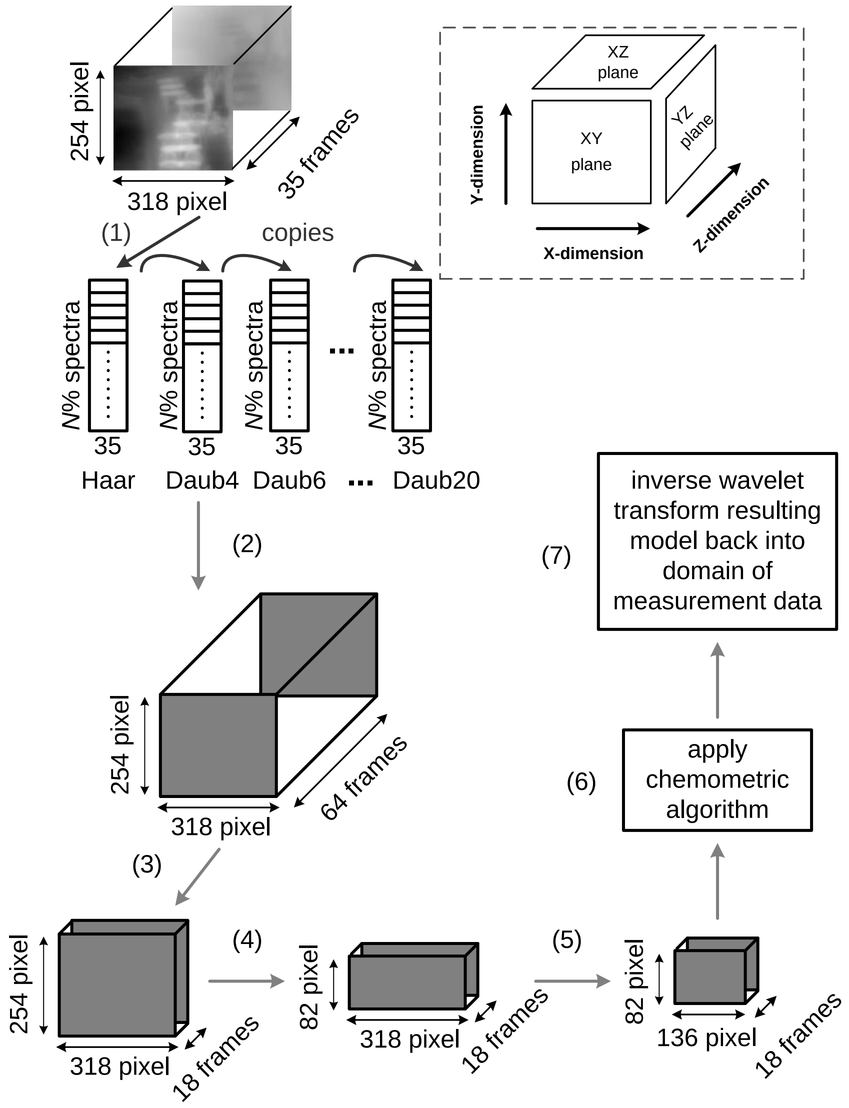


Figure 7: Flowchart outlining the major steps of the wavelet selection algorithm. (insert top right: definition of data cube dimensions and planes) (1) *N*% of the test vectors in the Z-dimension are randomly selected and stored in a matrix. This matrix is then copied as many times as there are different wavelet types being considered (i.e. ten). Next, 1D wavelet transforms are applied to the individual vectors (shown as rows) of the subsets and the best wavelet type for this subset is determined (see text). (2) Since the subset is considered to be representative of the entire Z-dimension of the original data cube, this best wavelet type is applied to transform the entire Z-dimension. (3) Compression is applied in the Z-dimension (see text). Steps (4) and (5) repeat this procedure for the Y- and X-dimension until the final compressed data cube is obtained. (6) This cube is analyzed by means of the chemometric method of choice. (7) Zeros are inserted into the final results, for example, PCs and score images, at the locations where wavelet coefficients were removed during compression. Inverse WTs of appropriate dimensionality (e.g. 1D for PCs and 2D for score images) finalize the computations and derive the wanted results in the original measurement domain.

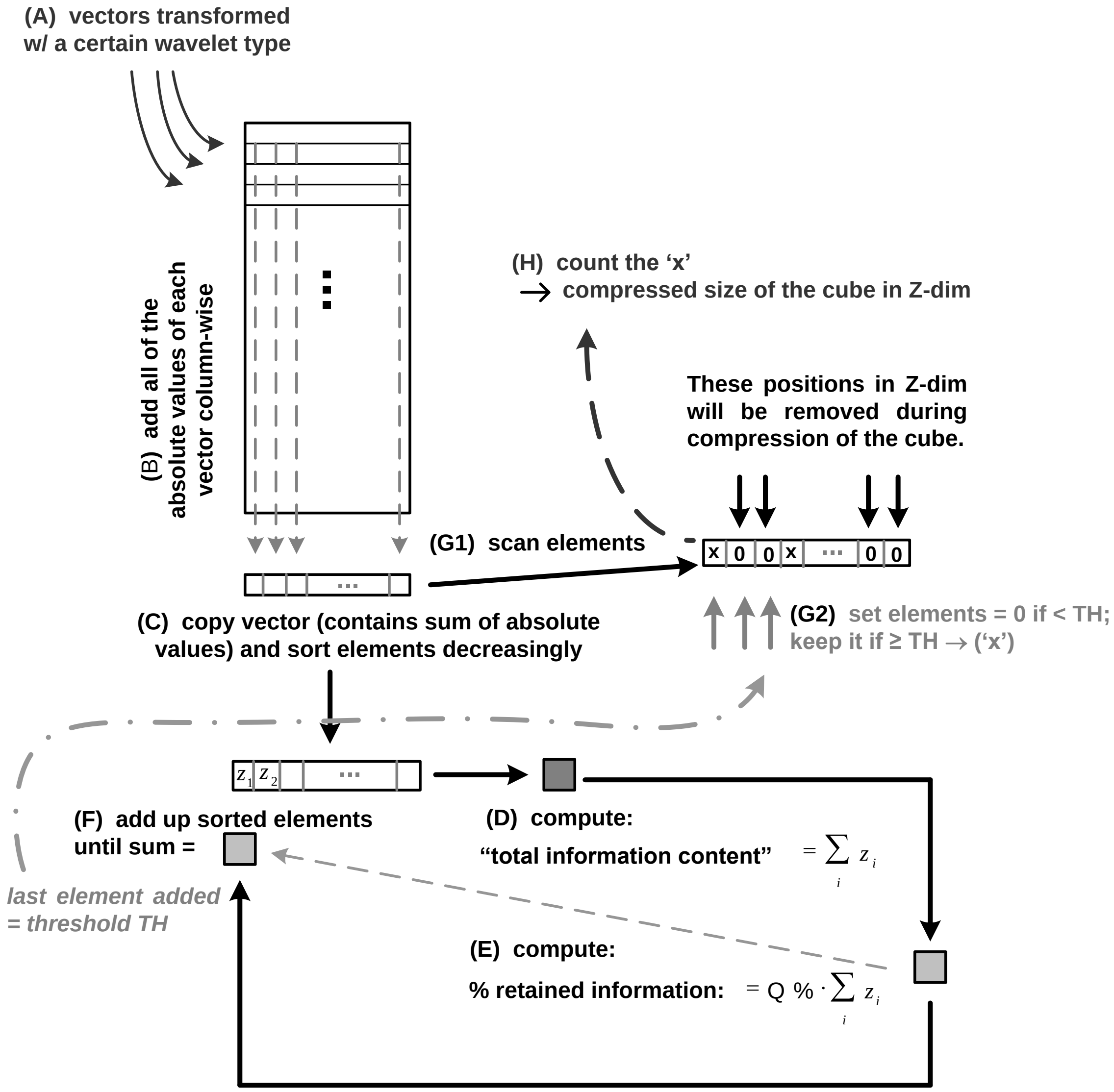

(A) vectors transformed w/ a certain wavelet type
(B) add all of the absolute values of each vector column-wise
(H) count the 'x' → compressed size of the cube in Z-dim
These positions in Z-dim will be removed during compression of the cube.
(G1) scan elements
x 0 0 x ⋯ 0 0
(C) copy vector (contains sum of absolute values) and sort elements decreasingly
(G2) set elements = 0 if < TH; keep it if ≥ TH → ('x')
z_1 z_2 ⋯
(F) add up sorted elements until sum =
last element added = threshold TH
(D) compute: "total information content" $= \sum_i z_i$
(E) compute: % retained information: $= Q\ \% \cdot \sum_i z_i$

Figure 8: Flowchart outlining the compression of the Z-dimension; see (54) for definition of total information content and % retained information (*Q*%). See text for step-by-step discussion of the procedure. The same method is applied subsequently to the Y- and X-dimensions.

Use this wavelet to transform all of the cube's vectors in the Y-dimension. Next, perform compression in the Y-dimension of the original cube.

(5) Steps 1 and 2 are repeated to find the best wavelet type for the X-dimension (or YZ-plane); see ***comment*** in step 4. Use this wavelet to transform all of the cube's vectors in the X-dimension. Next, perform compression in the X-dimension of the original cube.

(6) After the Z, Y, and X dimensions of the original data cube are compressed, the chemometric algorithm of choice (here, PCA) is applied.

(7) Last, zeros are inserted into the compressed chemometric results at the locations where wavelet coefficients were removed during compression (see Chapter 2.5). This step allows for an inverse WT (iWT) to be applied to these results, thus finalizing the computations. After the iWT, the results are now in the original measurement domain. In this study, since PCA is used, a 2D iWT and a 1D iWT are applied to the score images and principal components (PCs), respectively.

This procedure can be restricted to steps (4) – (7) for compressing (2D) matrices or to steps (5) – (7) if a number of (1D) vectors are compressed. Of course, this procedure can be easily expanded to N ≥ 4-dimensional hypercubes.

The next paragraph discusses how the randomly selected subsets are transformed and compressed using different wavelets (steps (1) – (3)). Also, details are given about how the two main figures of merit, size and accuracy, are incorporated in the selection of the best wavelet for a specific dimension (Z, Y, and X).

In step (1) (see Figure 7, top left), a representative subset of vectors in the Z-dimension is extracted and copied as many times as there are wavelet types incorporated into the analysis (i.e. ten). All of the vectors in each copy are then 1D wavelet transformed, one-by-one, with one of the wavelet types. The resulting vectors are stored in the rows of new matrices (see Figure 8 – step (A)). In step (B), the absolute values of a matrix are added column-wise; each column corresponds to a particular wavelet coefficient. Absolute values are added to avoid having large positive and negative wavelet coefficients from canceling. As was discussed in Chapter 2.5, both large positive *and* negative wavelet coefficients are considered relevant and need to be preserved to accurately represent the data. The size of the resulting sum of absolute values is an indication of how important a certain wavelet coefficient is for a particular subset and, also, for the entire Z-dimension. Step (C) copies this vector and orders its elements in decreasing order. To assess the relative importance of the individual vector elements, step (D) adds up all vector elements, thus resulting in a value labeled “total information content”. The user has to define how much (*Q*%) of the “total information content” shall be retained during the compression (step E), i.e.:

$$\text{\% retained information} = Q\% \cdot \text{total information content} \tag{54}$$

Once “% retained information” is derived, a threshold indicating the least relevant wavelet coefficient is determined (step (F)). This is done by first adding the elements of the sorted vector that was determined in step (C). The last added element that makes this sum ≥ “% retained information” is the smallest relevant element. This element is named TH (for threshold, see step (F)). In step (G1) and (G2) the elements of the vector determined in step (C) are compared to the threshold TH (step (F)). Values smaller than TH indicate irrelevant wavelet coefficients and will later be removed from the wavelet transformed data cube. In order to preserve this information, irrelevant elements of this vector are set to zero. The last step (H) counts how many relevant (non-zero) wavelet coefficients have been found. The number of non-zero elements represents the size of the compressed cube for the specific dimension that is being analyzed. After performing steps (A) – (H) for all ten copies of the test matrix (selected subset), the best wavelet type for this dimension is determined.

The figure of merit ‘size’ will be derived from the information gained in step (H) (see Figure 8):

$$\text{size\_Z}(\text{wavelet}[i]) = \#(\text{wavelet coefficients} \neq 0) \quad \text{with:} \quad i = 1 \cdots 10 \tag{55}$$

Since the same percentage *Q*% (54) is used for all wavelet types, the wavelet retaining the smallest number of non-zero wavelet coefficients (55) is considered the best regarding compressed cube size. This procedure (A) – (H) is performed for all ten copies of the test matrix.

Now, a figure of merit for accuracy is defined which is based on the results obtained during step (G2) in Figure 8. There, wavelet coefficients, or more precisely, positions in the wavelet transforms, have been classified as relevant (‘x’) or irrelevant (‘0’). Entire columns in the matrix shown in Figure 8 (top left) that correspond to ‘0-positions’ in step (G2) are set to zero; this is essentially a denoising step. Following this, the individual rows of all ten matrices are 1D inverse wavelet transformed using the same wavelet that was used for the previous transforms. This step produces the denoised versions of the original vectors. These denoised vectors are approximations of the original vectors. Next, the original vectors are compared to their denoised counterparts. The better a certain wavelet type is able to model the original vectors, the closer the denoised vectors will be to the originals. Thus, a figure of merit for assessing the accuracy of a specific wavelet is defined which utilizes squared residuals:

$$\text{accuracy\_Z}\left(\text{wavelet}\left[i\right]\right) = \sum_{\text{all test vectors}} \sum_{n} \left(\text{original\_vector}\left[i\right]\left[n\right] - \text{denoised\_vector}\left[i\right]\left[n\right]\right)^2$$

$$i = 1 \cdots 10 \quad \text{and} \quad n = 1, \cdots, \#\ \text{of vector elements}$$

(56)

Squared values are used to prevent positive and negative differences from canceling.

Since ten wavelet types have been employed size_Z (55) and accuracy_Z (56) are vectors containing ten elements each. In order to make both figures of merit of comparable weight, both need to be normalized. The following procedures, (57) and (58), will ensure that the smallest element, indicating the optimum wavelet type, for both size_Z and accuracy_Z will have a value of one. The larger elements within size_Z and accuracy_Z correspond to the less appropriate wavelets and will result in values much smaller than one. Thus, the figures of merit for size (55) and accuracy (56) for all $i = 1 \cdots 10$ wavelet types are computed to be:

$$\text{Figure\_of\_Merit\_size\_Z}\left(\text{wavelet}\left[i\right]\right) = \frac{\min\left(\text{size\_Z}\right)}{\text{size\_Z}\left(\text{wavelet}\left[i\right]\right)}$$

(57)

$$\text{Figure\_of\_Merit\_accuracy\_Z}\left(\text{wavelet}\left[i\right]\right) = \frac{\min\left(\text{accuracy\_Z}\right)}{\text{accuracy\_Z}\left(\text{wavelet}\left[i\right]\right)}$$

(58)

For some applications, such as data storage, small cube size may be more advantageous than high accuracy. In other applications, this may be the opposite. Thus, a weighing of the two figures of merit is introduced by defining a final selection criterion:

$$\begin{aligned}\text{selector\_Z}(i) = {} & \text{importance\_size} \cdot \text{Figure\_of\_Merit\_size\_Z}(\text{wavelet}(i)) + \\ & \text{importance\_accuracy} \cdot \text{Figure\_of\_Merit\_accuracy\_Z}(\text{wavelet}(i)) \\ & \text{with: } \text{importance\_accuracy} + \text{importance\_size} = 100\%\end{aligned} \qquad (59)$$

The ten elements ($i = 1 \cdots 10$) of selector_Z have values between zero and one. The largest element indicates the optimum wavelet type regarding a user's size and accuracy preference. Now the best wavelet for the Z-dimension has been determined and will be used to transform all of the vectors in the Z-dimension (see step (2) in Figure 7). Compression can be performed quickly if the decision regarding relevant and irrelevant wavelet coefficients utilizes the results found during wavelet selection. For each wavelet type, this classification has been determined in step (G2) (see Figure 8). Since the results are different for each wavelet, it is important to use the classification belonging to the chosen wavelet. Because the test set is considered to be representative of the entire dimension, the same selection of relevant wavelet coefficients can be applied to the entire Z-dimension of the original data cube. In other words, a '0' found in step (G2) of Figure 8 indicates an XY-plane (Figure 7, top right) that will be removed from the data cube after wavelet transformation. Entire planes have to be removed from the cube in order to prevent mixing of wavelet coefficients in different vectors.

After compressing the Z-dimension, the same procedure is applied to the cube's Y-dimension (step (4), Figure 7) and X-dimension (step (5), Figure 7). After all dimensions are wavelet transformed and compressed, the resulting smaller data cube is then processed by the chemometric method of choice (step (6), Figure 7). After finalizing chemometric computations, zeros are filled into the positions where wavelet coefficients have been removed (=uncompressing) and an inverse WT is performed using the correct wavelet to bring the final results into the original measurement domain (step (7), Figure 7).

This algorithm allows for the quick determination of an optimum wavelet combination for N-dimensional data sets. As will be presented in Chapter 3.3, there are typically several appropriate combinations. It should be stated that the wavelet combinations selected by this algorithm are typically sub-optimal since the choice of wavelets is based on a randomly selected sub-set of the original data. If, for instance, at a given compression level, the subset were to contain 100% of all the available data vectors, then the selected wavelet combination is considered optimum. Using all data vectors as the subset needs to be avoided since this would not benefit computation speed. Again, the test subset needs to be large enough to be representative of the entire data set, yet be as small as possible to keep computation efforts to a minimum. Nevertheless, deviations from the true optimum wavelet combination are possible. The results that will be presented in Chapter 3.3 demonstrate the capabilities of this methodology. Wavelet combinations chosen by the algorithm clearly contain strong advantages over combinations that are never selected. Compared to previous wavelet compression strategies [36], [37], [55], this algorithm invests some additional computation time for selecting an appropriate wavelet combination. This additional time is still overcompensated by the amount time that is saved by performing chemometric calculations on compressed data sets. The advantage of this proposed selection method is that it provides enhanced model quality and minimum data set sizes although it loses some computation speed due to the selection process.

3.2 Experimental

Two different data sets acquired with two different experimental setups are used to develop and evaluate this algorithm. The main purpose of using these different data is to show that the algorithm is applicable for different data types.

Data set #1 is a 3-D data cube that was acquired with a mid-infrared (3 - 5µm) imaging sensor described in previous publications [56] - [59]. The dimensions (X (spatial), Y (spatial), Z (spectral)) of this data set are (318, 254, 35), i.e. 318 x 254 = 80772 rather short spectra of 35 wavelength positions. The purpose of this experiment was to demonstrate the feasibility of passive remote sensing and to prove the ability to classify different materials based on their respective IR emission spectra. In this study, IR emission spectra of plastic furniture in a natural setting were acquired. A PCA was applied to these spectra and red-green-blue (RGB) encoded score images were calculated using multivariate image analysis (MIA) [16] (refer to Chapter 1 and Figure 2). These RGB images show different materials being represented as different colors (the corresponding RGB score images were published in [36]).

Data set #2 (X = 64, Y = 64, Z = 311) was acquired by means of FTIR microscopy (Bruker Optics Vertex 70 coupled with a Bruker Hyperion 1000 IR/VIS microscope featuring a 64 x 64 pixel MCT focal plane array, Santa Barbara Focal Plane, Inc). The sample used in this data set is a piece of Parafilm®. Its RGB score image was calculated by MIA [16] (also see Figure 2 and related discussion) and is shown in Figure 9. This data set contains fewer spectra than data set #1, but its spectra are much longer covering a wavelength range of ~1450 – 850 cm^{-1}. In the following section the consequence of a longer wavelength range will be shown. Both data sets were mean centered [9] prior to all calculations to remove any common background that may be present.

3.3 Results and Discussion

3.3.1 Finding a Representative Test Set Size

First, it is required to determine how many test spectra (*N*% of the total number of vectors in each dimension) are needed to reliably represent the overall information contained in a data cube (refer to (1) in Figure 7). The number of test spectra must be large enough for the test set to be representative, yet be as small as possible in order to limit computational expense for finding the best wavelet. To determine the appropriate size of the test subset, several preliminary analyses were performed using different percentages of the available test spectra. These tests were performed on data set #1 at an accuracy/size ratio of 50/50 (59) and a rather low compression level (*Q*% = 90% retained information (54)).

As a benchmark, 100% of the test spectra were chosen to determine the optimum wavelet combination at a specific compression level. Since all vectors were used for this analysis the resulting wavelet combination represents the optimum wavelet combination for this data set at a compression level of *Q*% = 90%. In this particular case, computational expense is of no importance because the goal is to simply find the optimum combination. The optimum combination for data set #1 was found to be X dim. = Daub4, Y-dim. = Daub12 and Z-dim. = Daub8. This combination is labeled

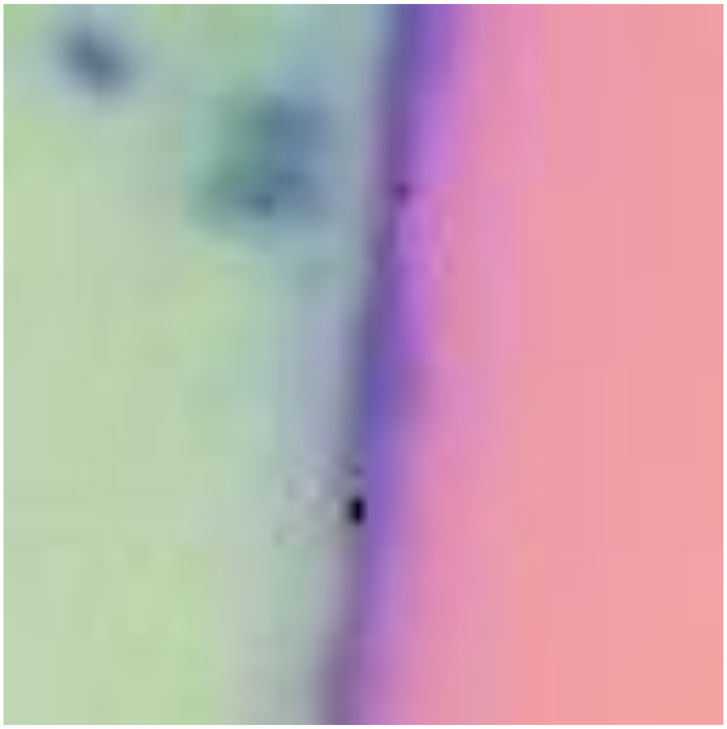


Figure 9: RGB score image of data set #2 (Parafilm® data cube) acquired by means of MIA [16]. On the left is a piece of Parafilm®; empty background is shown on the right.

opt_WT(4, 12, 8). As expected, this optimum wavelet combination was reproducible after several test trials.

Next, *N*% = 50%, 25%, 10% and 5% of the available test vectors were randomly chosen and used as the subsets in the algorithm. In order to assess the reproducibility of the selection routine, the algorithm was performed 50 times for each of the subset sizes. As less test vectors are included in the subset, deviations from the “benchmark” combination (opt_WT(4, 12, 8)) are used as an indication of a test set becoming too small to be representative. Histograms are plotted (Figure 10) to display how many times the different wavelet types were chosen after the 50 calculations. At *N*% = 50% the optimum combination was selected 25 times out of the 50 trials. Another wavelet combination (X-dim. = Daub4, Y-dim. = Daub8 and Z-dim. = Daub8 or WT(4, 8, 8)) occurred 14 times showing that competition between the wavelets exists when implementing smaller test subsets. This competition indicates that more than one good wavelet combination is possible.

As the percentage of test vectors included in the subset decreases, more wavelet types are chosen and the optimum wavelet combination is found less frequently. For instance, *N%* = 25%, WT(4, 8, 8) occurred 15 times and opt_WT(4, 12, 8) occurred 10 times. For *N%* = 10%, WT(4, 8, 8) was chosen 13 times and opt_WT(4, 12, 8) was found 7 times. Based on these results, *N%* = 10% was chosen as the appropriate size of the subset to be tested by the algorithm. Using *N%* = 5%, a much broader distribution of wavelet types for all dimensions was observed. Such a limited subset size was considered to be too small to be representative of the data cube.

3.3.2 Impacts of Wavelet Combinations on Chemometric Models

The main goal of this study is to select an optimum combination of wavelets regarding computation speed, model accuracy and compressed data set size. The task at hand is to pick one out of 10 x 10 x 10 = 1000 possible wavelet combinations that ensures good performance. Further, this selection must be performed without introducing large computational burdens.

Based on the approach of this algorithm, wavelet combinations that are not selected are considered to be poor choices of wavelets in which to represent the data. Since this algorithm analyzes a subset of the original data, the chosen wavelet

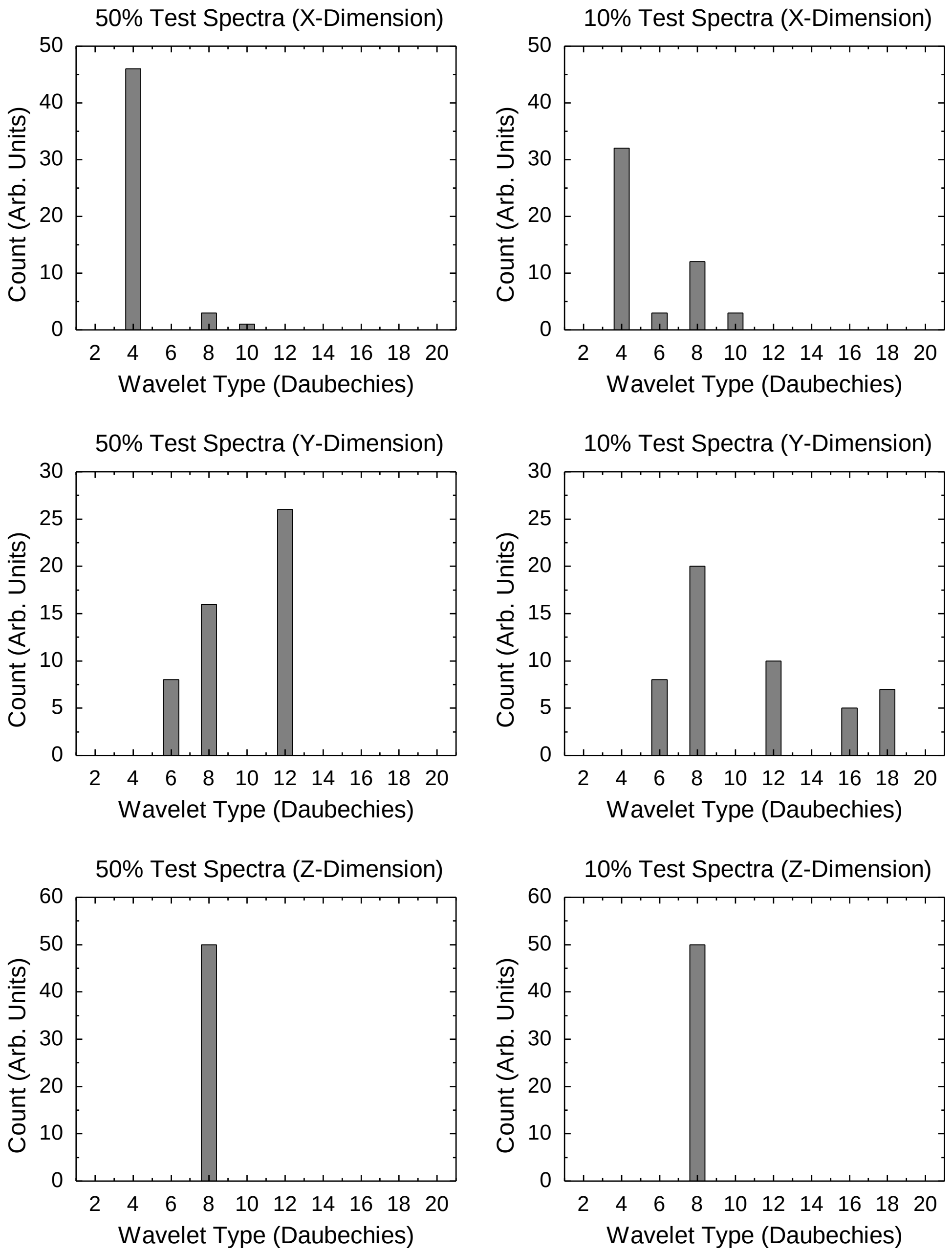


Figure 10: Histograms displaying the results of the wavelets chosen for each dimension of data set #1. Here, the algorithm picked *N*% = 50% (left) and 10% (right) test vectors and was performed 50 times for each case (see text). A compression level of 90% retained information was applied here.

combination is an estimate of the true, optimum combination. Thus, selected combinations are considered to be near-optimum. In other words, the combinations that are chosen by the algorithm are not necessarily the best (optimum) combinations; however, as shown below, they are superior to the combinations that are never selected. Several parameters (figures of merit, see Chapter 3.1) will be analyzed to determine whether or not a selected wavelet combination provides results that are superior to those obtained using a non-selected combination. In this section, residual images are calculated using both selected and non-selected wavelet combinations and these images are compared. Also, a comparison is given of the compressed and uncompressed principal components (PCs, see Chapter 2.3) obtained with different wavelet combinations.

Speed: To evaluate the increase in computation speed, an acceleration factor is defined:

$$\text{acceleration\_factor} = \frac{\text{comp.time(conventional analysis)}}{\text{comp.time(wavelet determination} + \text{compression} + \text{analysis of compressed cube} + \text{inverse WTs)}} \qquad (60)$$

Figure 11 shows a plot of the acceleration factors of the two data sets versus the amount of retained information (*Q*%). As the amount of retained information increases the acceleration decreases. Data cubes that are compressed less retain more information, therefore they are larger in size and more time is needed to complete the desired computations. For each data set, ten acceleration factors were calculated at each compression level (*Q*% = 95% - 75%). More than one acceleration factor was calculated at each compression level since it is possible that different wavelet combinations can be determined during each trial. The different combinations can lead to slightly different computation times. Therefore, average acceleration factors are determined and plotted along with their corresponding error bars (Figure 11). Error bars were obtained by calculating the standard deviation from the ten calculated acceleration factors. An acceleration factor of ~21 was reached for data set #2 whereas for data set #1 an acceleration factor of only ~2.4 was obtained. The considerably larger acceleration factor for data set #2 can be explained by the following formula. This formula relates the amount of floating point operations (flops) required to compute a SVD to the dimensions of the data matrix $\mathbf{X}_{(K\times N)}$ [38]:

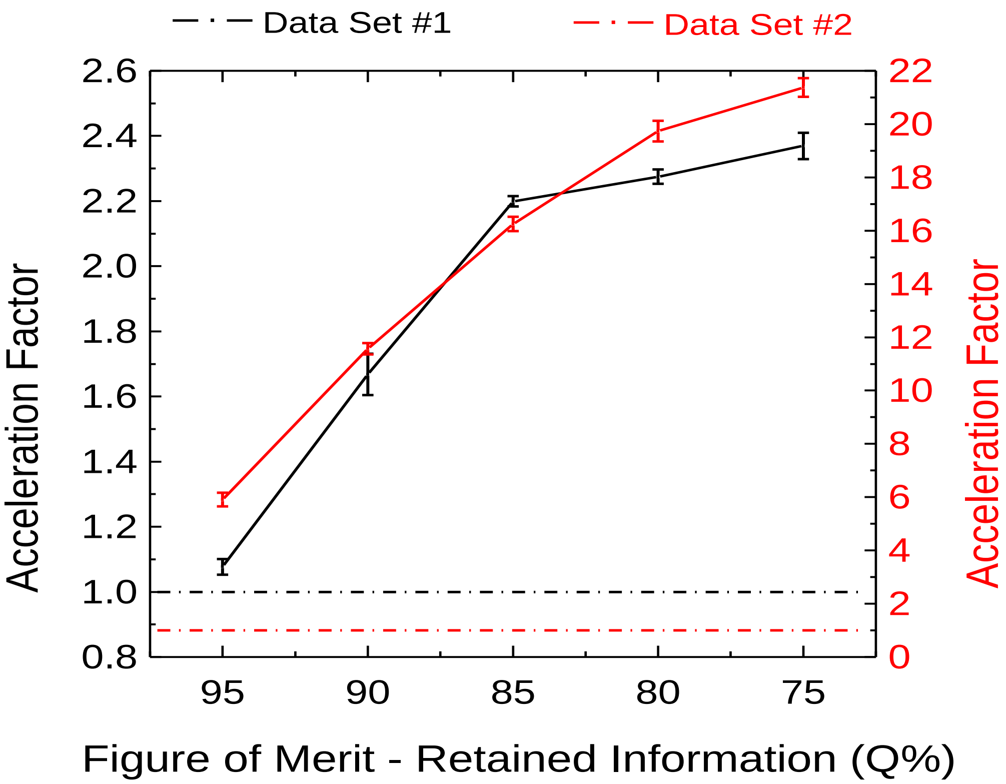


Figure 11: Acceleration factors (60) obtained for the two data sets versus the amount of retained information *Q*% (see (54)). As expected, the more information that is retained, the less acceleration that can be achieved. For both y-scales, the dash dotted lines indicate acceleration factor = 1. In order for the algorithm to be advantageous, the acceleration needs to be > 1. Error bars were obtained by calculating the standard deviation from the multiple acceleration factors (see text).

$$\text{flops}\left(\text{SVD}\left(X_{(K\times N)}\right)\right) \cong 14\cdot K\cdot N^{2}+8\cdot N^{3}$$

(61)

The dimensions (X, Y, Z) of the uncompressed data sets are (see Chapter 3.2): #1 = (318, 254, 35) and #2 = (64, 64, 311). After unfolding these cubes into data matrices their dimensions $(K\times N)$ become (80772 x 35) and (4096 x 311) for data sets #1 and #2, respectively. K is the total number of spectra contained in the original data cube and N is the number of wavelength positions for each of the spectra. Although data set #1 is much larger in K than data set #2, data set #2 is much longer in the spectral dimension $(N_{\#2} >> N_{\#1})$. Therefore, data set #2 requires many more flops ($\text{flops}_{\#2} \cong 5.8\times 10^{9}$) to perform a SVD (61) than data set #1 ($\text{flops}_{\#1} \cong 1.4\times 10^{9}$). Hence, a slight compression in the spectral dimension (N) for data set #2 can result in potentially higher acceleration factors compared to data set #1 since N decreases in second and third order (61); this was observed (see Figure 11).

Compressed data set size: To assess how much compression is achieved for a data set at a given compression level, size is determined by calculating the total dimension of the final compressed cube:

$$\text{size\_compressed\_cube} = \text{dim\_X}\cdot\text{dim\_Y}\cdot\text{dim\_Z}$$

(62)

Three different accuracy/size ratios (59) were chosen (90/10, 50/50 and 10/90) to demonstrate the algorithm's ability to tailor the optimum wavelet combination according to the selected ratio. The 90/10 ratio has a strong emphasis on accuracy whereas size is of small importance. At 50/50 there is an equal emphasis on both figures of merit and 10/90 has a slight emphasis on accuracy and a strong emphasis on size. In order to check the reproducibility, all calculations were repeated 25 times at each ratio using data set #1. 25 $\text{size\_compressed\_cube}$ (62) values for each accuracy/size ratio were averaged to give a total of three $\text{size\_compressed\_cube}$ values. These values along with their respective error bars are plotted in Figure 12. These results show a trend following the expectations regarding cube size; however, in this limited study, the differences appear not to be significant.

Accuracy: A measure for information preservation is defined to determine how much approximation is introduced by the compression step. Referring again to Chapter 2.5,

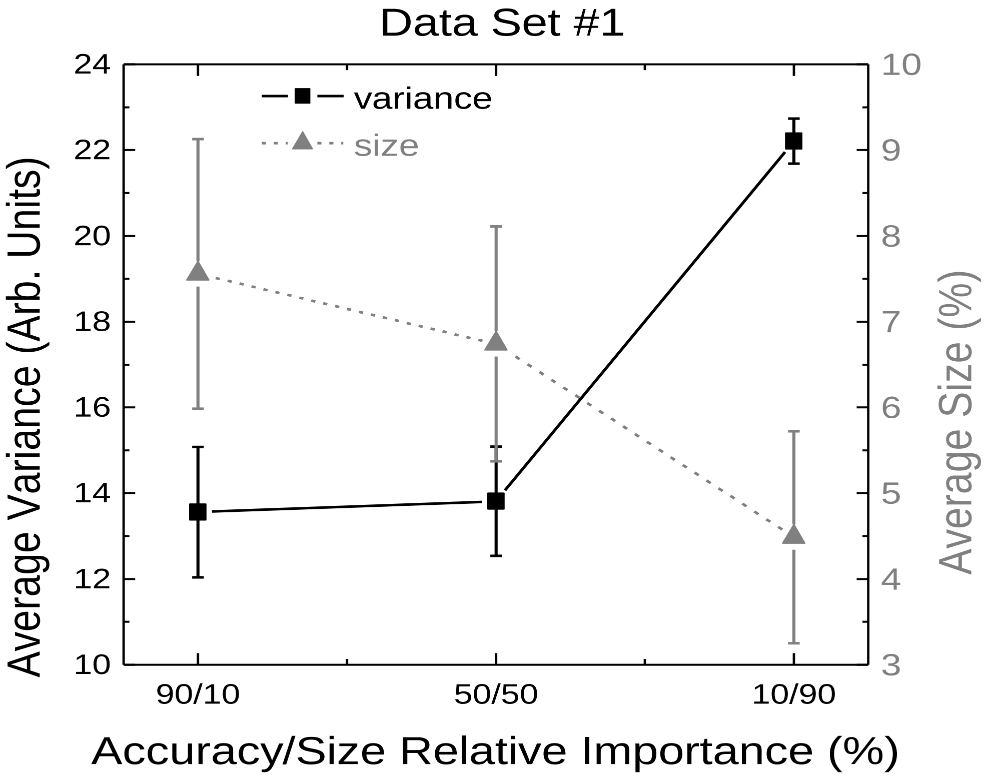


Figure 12: Average size (62) and variance (63) of compressed data set #1 versus accuracy/size importance. Error bars are obtained by calculating the standard deviation for each data trial (see text).

this step is essentially a denoising process and is required to properly perform the inverse transform. Next, a 3D inverse WT is completed using the same hybrid wavelet that was used during compression. Now, the original and reconstructed data cubes can be compared. All deviations are due to approximations made during compression. To measure the model accuracy achieved with each of the different accuracy/size ratios, the following variance is calculated (63).

$$\text{variance} = \frac{\sum_{i=1}^{\text{allelements}} \left(\text{element\_a}_i - \text{element\_b}_i\right)^2}{\text{dimension\_X} \cdot \text{dimension\_Y} \cdot \text{dimension\_Z} - 1}$$

(63)

In (63) element_a_i indicates the i^{th} element of the original data cube; element_b_i is the i^{th} element of the reconstructed data cube. What is expected is that when a strong emphasis is placed on accuracy (i.e. accuracy/size ratio = 90/10), small variance values should be calculated. These values should increase as more emphasis is taken away from accuracy and placed on compressed data set size (i.e. accuracy/size ratio = 10/90). These computations were performed only to assess the quality of the compression-based models; computation speed was of no importance here.

In 25 repetitions, the average variance values calculated for each of the three accuracy/size ratios are plotted in Figure 12. The error bars indicate the corresponding standard deviations. Again, average variance values are calculated since it is possible for different wavelet combinations to be determined by the algorithm during each repetition. The different wavelet combinations can result in slightly different variance values. According to Figure 12 a trend between variance and size is evident; however, two ratios may not be significantly different from each other.

Now we compare results obtained with a wavelet combination selected by this novel algorithm to a combination that was never selected. The benchmark is a conventionally computed PCA which does not include a compression step. This comparison is based on so-called ‘residual images’ which contain information removed from the data sets during compression. Also, principal components (PCs) obtained from the compressed data sets will be compared to those calculated using the uncompressed data.

It is demonstrated that the proposed wavelet algorithm selects optimum wavelet combinations and rejects less optimal wavelets. Wavelet combinations that are never selected by the algorithm are considered to be non-optimal. To compare the performance between optimal vs. non-optimal combinations, representative combinations are manually chosen based on the histograms in Figure 10 and Figure 13 for data sets #1 and #2, respectively. Again, wavelet combinations returned by the algorithm are considered to be better for accurate data representation and compression than non-selected combinations.

At 90% retained information (54) using 50/50 accuracy/size weighing, the combination WT(4, 8, 8) is chosen as an optimal combination for data set #1 since it is selected more often than any other wavelet combination (Figure 10). A non-optimal combination for data set #1 is selected to be WT(2, 10, 6) because neither of these wavelets are picked for the corresponding data set dimensions (Figure 10). For data set #2, an optimal combination of WT(2, 4, 2) is chosen for a compression level of 80% retained information and 50/50 accuracy/size weighing. A non-optimal combination for data set #2 is selected to be WT(6, 12, 4).

Optimal and non-optimal wavelet combinations have different capabilities of preserving information within a data set at a given compression level. This is visualized in Figure 14 by means of residual images derived for data set #1. A residual image is computed in three steps: first, a reconstructed data cube is subtracted from the original cube (compare discussion leading to (63)). Next, all elements of this resulting cube are squared to prevent positive and negative elements from canceling in the next step. Finally, all frames of this cube are added up along the Z-dimension resulting in a residual image. This procedure was performed after compression utilizing both an optimal (Figure 14, left) and non-optimal (Figure 14, right) hybrid wavelet combination. In both images, dark areas represent points of small differences between the original and reconstructed cube; white areas represent strong deviations. It is obvious that using a non-optimal combination introduces larger deviations from the original data than when using an optimal combination. Unlike the image on the left, the right image contains several noticeable white areas which provide evidence of various structures (refer to Fig. 3 (top left) in ref [36] and Fig. 2 in ref [60] for the original thermal (or IR) image). Thus, at the same compression level, the non-optimal wavelet combination cannot preserve as much information as the optimal combination.

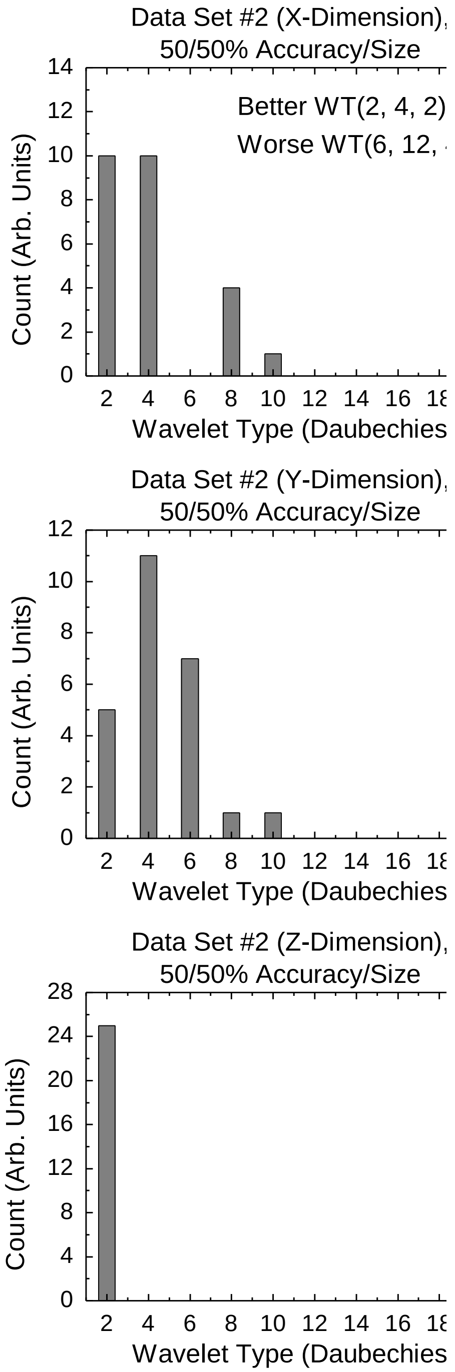


Figure 13: Histograms of the wavelets chosen for each dimension of data set #2. From these histograms it can be deduced that one optimal wavelet combination is WT(2, 4, 2). The wavelet combination WT(6, 12, 4) is one that is never chosen by the algorithm, i.e. a non-optimal combination. A compression level of 80% retained information is applied here.

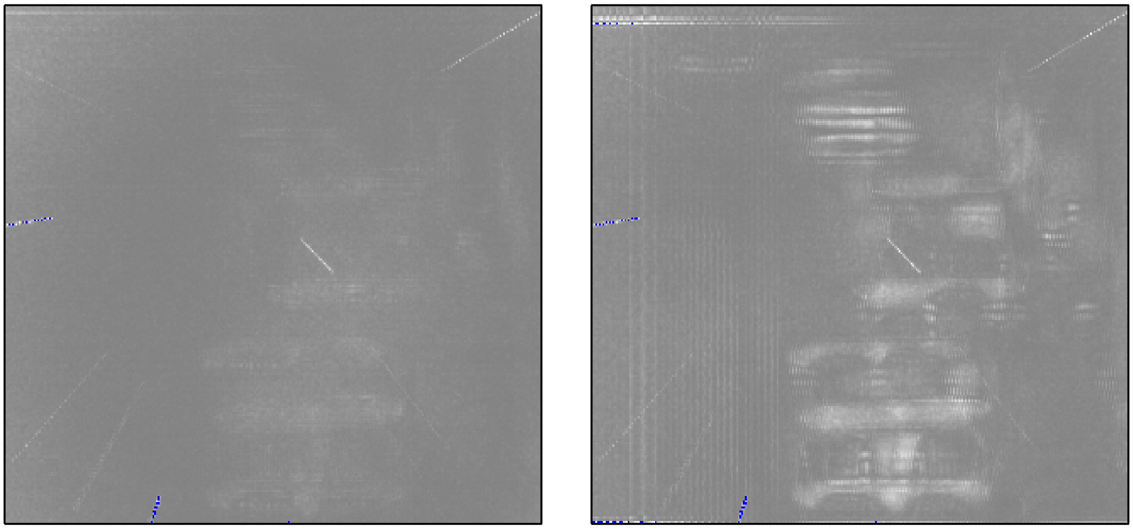


Figure 14: Residual images of data set #1; see text (refer to Fig. 3 (top left) in [36] and Fig. 2 in [60] for thermal images of the shown scene). The image on the left was obtained by using the optimal wavelet combination WT(4, 8, 8) chosen by the algorithm (*N*% = 10% test vectors, 50/50 accuracy/size ratio, *Q*% = 90% retained information (54)). Darker regions in the images represent areas of small differences between the compressed model and the original data. Lighter regions indicate areas of stronger differences. The right image was generated from a wavelet combination WT(2, 10, 6) not selected by the algorithm (under the same compression conditions as the left image). There are noticeable advantages (smaller deviations) when using the selected (optimal) hybrid wavelet (left) as opposed to using a combination (right) never chosen by the selection algorithm.

To further assess the effects a wavelet combination has on a chemometric model, the PCs obtained from a conventional PCA are compared to their compression-based counterparts. The compression based PCs were obtained from data set #1 by both an optimal (Figure 15, left) and non-optimal (Figure 15, right) wavelet combination. There is enhanced agreement between each PC when the optimal combination is used. However, for the non-optimal combination, stronger deviations become visible in the 2nd and 3rd PCs (indicated by arrows). A similar comparison is made for the first three PCs of data set #2 and again good agreement between the first two PCs is observed for the optimal combination (Figure 16, left). However, for the third PC more deviations are present. These deviations are attributed to the greater compression level (80% retained information) used during evaluation of data set #2. When the non-optimal wavelet combination is applied (Figure 16, right) additional features are introduced in the first two PCs (indicated by arrows).

These examples demonstrate the superior results obtained when using an optimal hybrid wavelet combination. This combination is automatically determined by means of this new selection algorithm.

3.4 Conclusions

Incorporating wavelet transform compression into chemometric data analyses is an efficient and practical method to compress large data sets and accelerate computations. As a result, decreased data storage requirements, reduced overall computation times, and enhanced time resolution, in chemical sensing, can be achieved. Hybrid wavelet transforms have been implemented to further fine-tune data compression and model accuracy. However, even though hybrid wavelet transforms can be beneficial for data analysis purposes, one main question still remained: which wavelet combination is best for a data set regarding computation speed, compressed data set size, and accuracy of the computed chemometric models?

In this research, an algorithm is proposed that selects the optimum wavelet combination based on a desired compression level and two figures of merit: model accuracy and resulting data set size. It was found that test subsets comprising of only 10% of the data vectors contained within a certain data set dimension are considered to

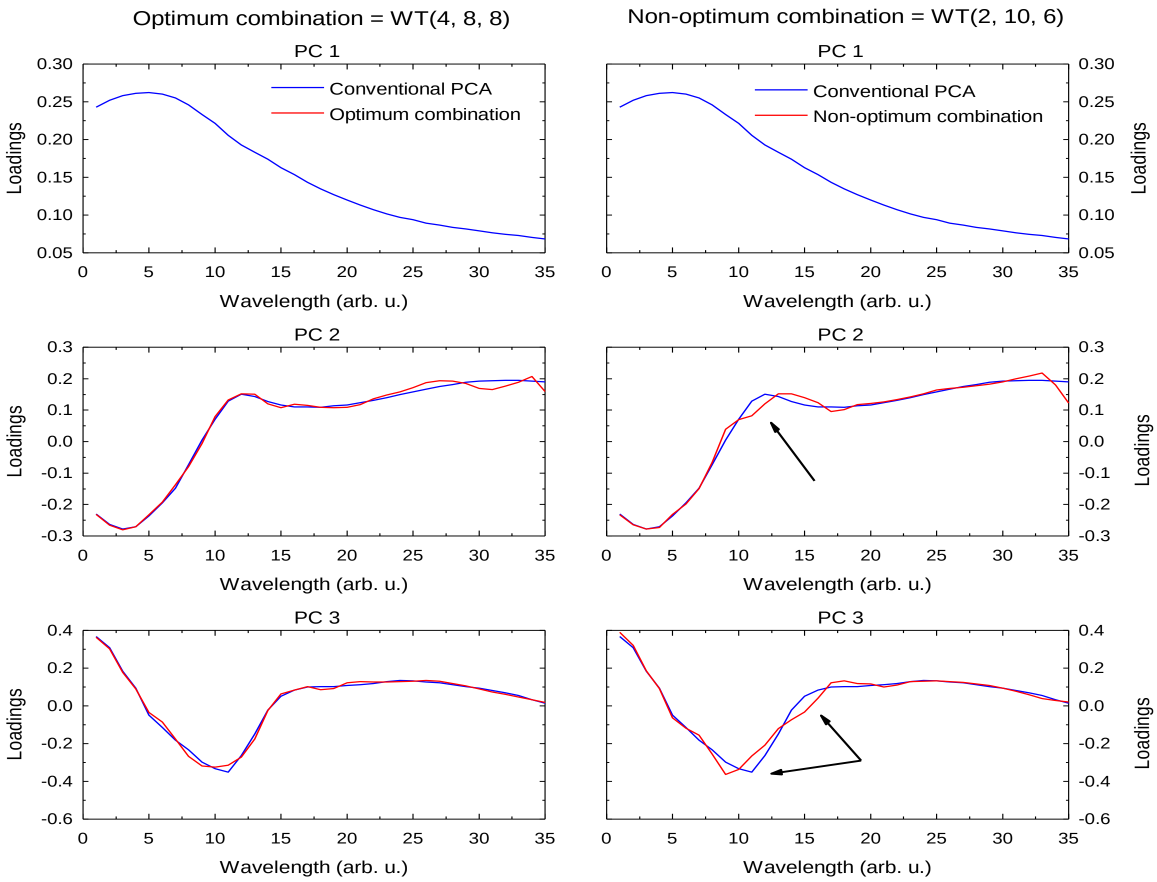


Figure 15: Plots comparing the first three principal components (PCs) for data set #1. The left column is a comparison between the PCs obtained from the conventional PCA method with no applied compression and the PCs obtained by using an optimal wavelet combination. A 50/50 accuracy/size ratio and a compression level of 90% retained information (54) were used for data set #1. The right column shows a similar comparison, but instead, using PCs obtained from a non-optimal wavelet combination. Arrows indicate stronger deviations from the true PCs when a non-optimal wavelet combination is utilized.

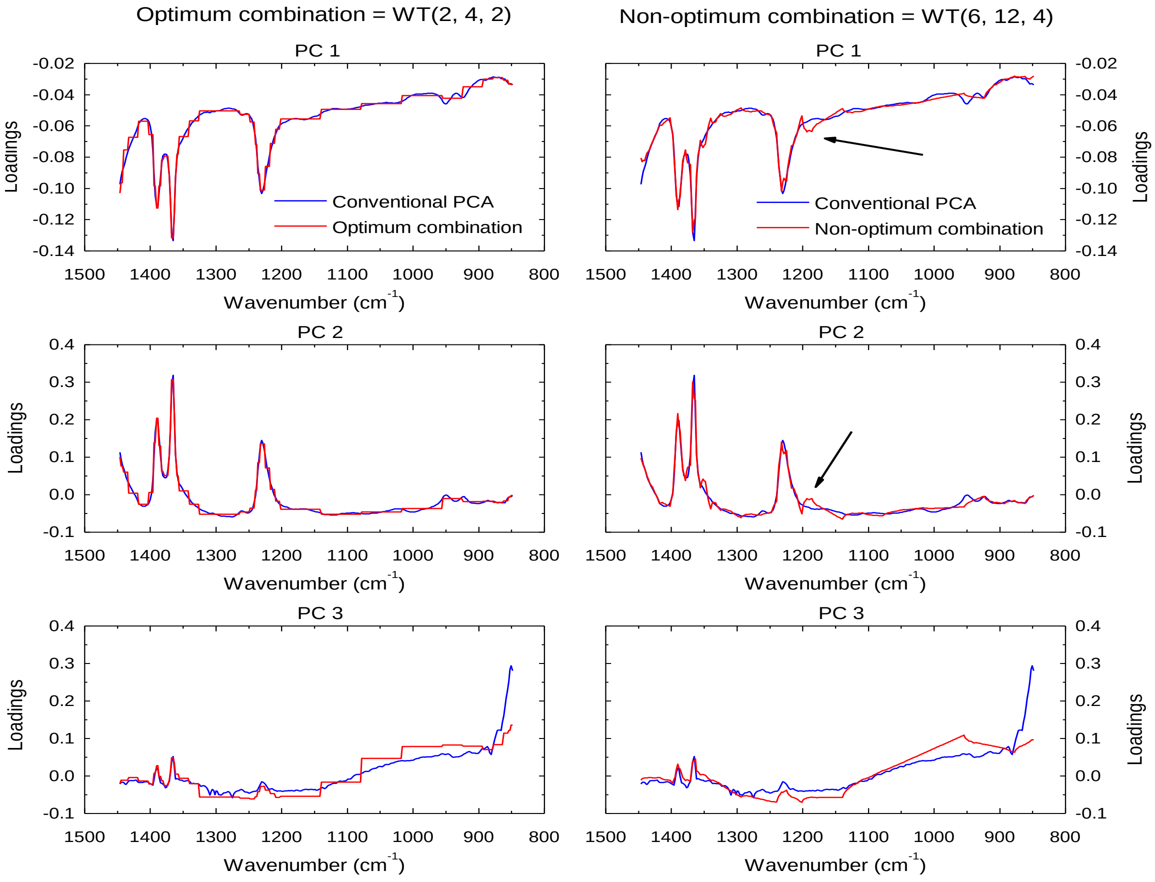


Figure 16: Plots comparing the first three principal components (PCs) for data set #2. The left column is a comparison between the PCs obtained from the conventional PCA method with no applied compression and the PCs obtained by using an optimal wavelet combination. A 50/50 accuracy/size ratio and a compression level of 80% retained information (54) were used for data set #2. The right column shows a similar comparison, but instead, using PCs obtained from a non-optimal wavelet combination. Arrows indicate stronger deviations from the true PCs when a non-optimal wavelet combination is utilized.

be representative of the entire, original data set. From these test subsets, the selection algorithm is capable of choosing optimum wavelet combinations that are used to transform and compress the original data. Once the optimum wavelet for a certain dimension is found it is applied to all the vectors in the corresponding dimension of the original data set. The algorithm progresses dimension-wise until the optimum wavelets for all dimensions are determined and applied.

In repeated test runs, it is possible for several wavelet combinations to compete. When this occurs, the shorter wavelets are used since they perform WTs (and iWTs) faster. However, all of these different combinations are still considered to be suitable. Overall computation times are decreased with this algorithm even though the initial preprocessing step requires additional computation time. These additional computations are later over-compensated since chemometric methods are applied to smaller, compressed data sets. Some of the studies produced acceleration factors up to ~22.

Since the algorithm chooses a wavelet combination based on a representative sub-set, only the accelerated methodology is potentially sub-optimal. Nonetheless, the results demonstrate that this is an effective technique. Two separate data sets from two different experimental setups are studied to demonstrate that the algorithm is applicable to different data types. Results show that this optimum wavelet method increases acceleration factors for both data sets. Residual images are generated to display the enhanced accuracy of the wavelet combinations chosen by the algorithm as opposed to the combinations never selected by the algorithm. Also, a comparison of principal components (PCs) is performed to demonstrate that the algorithm selects wavelet combinations that introduce fewer deviations into the chemometric models.

Chapter 4

Accelerating Kernel Principal Component Analysis (KPCA): Applications to Spectroscopic Imaging

Common linear algorithms such as principal component analysis (PCA) (Chapter 2.3) are frequently used to model chemical systems. These linear methods have been successful for many applications but are often ill-suited for modeling nonlinear processes. For complex data, where nonlinear behavior is often abundant, a method named 'kernel principal component analysis' (KPCA) (Chapter 2.4) [39], [40] has been developed and applied in a number of fields. Examples include nonlinear process monitoring for failure detection in waste water treatment plants [41] - [43], data denoising [44], recognition of handwritten digits [45], and classification of genetic data [46]. Despite the growing popularity of KPCA, its drawback is a high demand for computational resources; namely, memory and processing speed. As a consequence, calculations on common personal computers become unfeasible for many KPCA applications.

In this chapter, an algorithm is presented that performs two-dimensional (2D) wavelet compression of spectroscopic imaging [6] - [8] data during KPCA calculations. Compared to the aforementioned compression algorithms, the main technical challenge of this proposed method is that handling multi-gigabyte data sets has to be avoided during all points of the computations. This requires a novel approach which combines data handling, compression, and data analysis. In addition, a new way of mean-centering has to be developed that is compatible with this compression procedure (see Chapter 4.1.2).

This research demonstrates how the introduction of data compression enables the routine application of KPCA to large data sets (Chapter 4.1), as acquired, for example, in spectroscopic imaging. Chapter 4.2 briefly discusses two experimental setups that have been used for acquiring large real-world data sets. In Chapter 4.3, results are presented regarding the figures of merit ‘reduction in memory requirements’, ‘quality of compression-based models’, and ‘gain in computation speed’. The KPCA algorithm itself is outlined in Chapter 2.4. There, it is also shown how the KPCA algorithm compares and contrasts to PCA.

The computational methods discussed in the following chapter have been developed and implemented into C++ source code. All of the results presented in Chapter 4.3 were obtained using this code. All calculations were performed under Linux on a 64-bit Xeon machine with 32 GB of RAM utilizing an Intel® C++ compiler (version 9.1.049).

4.1 Kernel Principal Component Analysis (KPCA) Compression Method

4.1.1 Incorporating Data Compression into Calibration

Regardless of whether the large or the small covariance matrix is used (see Chapter 2.4), PCA can only return a number of principal components that is either equal to the number of variables (wavelength positions) or to the number of samples (spectra), whichever is smaller. Thus, it is computationally more efficient to always use the *smaller* covariance matrix when applying PCA. However, the “large” covariance matrix (see (39) in Chapter 2.4) must be used when applying KPCA. The reason for this is that KPCA can extract a number of principal components that exceeds the number of variables if the number of samples is greater [39], [40]. This is a significant advantage KPCA has over PCA. This potentially larger number of principal components KPCA is capable of extracting could possibly result in a more representative model for complex data sets.

One of the main steps of KPCA calibration is to solve an eigenproblem for a covariance matrix, the so-called Gram matrix, $\mathbf{K}$ (47), (48). $\mathbf{K}$ is built using measurement spectra that have previously been mapped from the ‘input space’ X into the ‘feature space’ H (see (40), (41) and related discussion). As outlined in Chapter 2.4, $\mathbf{K}$ has dimensions $M \times M$ with M being the number of (here) spectra.

Since spectroscopic imagers acquire thousands of spectra in parallel, memory requirements and the number of floating point operations needed to diagonalize $\mathbf{K}$ become unrealistic on a personal workstation. For example, one spectroscopic imager described in Chapter 4.2.2 features a focal plane array detector of 320 x 256 pixels. This setup produces data cubes (see Figure 17 top) that contain 320 x 256 = 81,920 spectra. The resulting Gram matrix $\mathbf{K}$ is of dimensions 81,920 x 81,920, or ~50 GB assuming double precision (64 bit per element) is used. New approaches are required that enable diagonalization of such large matrices on personal computers within reasonable time. Already available wavelet-based compression methods [1], [36], [37], [55] are not completely suitable for this study since they load the full data set first then compress it while holding all information in memory. Here, this must be avoided at all times due to the large data set sizes (i.e. Gram matrices (48)). The novel approach presented in this study is based on wavelet-compressing the Gram matrix $\mathbf{K}$ (48) 'on the fly' while it is generated (47). All wavelet types could be utilized; however, here, wavelets of the Daubechies family [19], [52] (see Chapter 2.5) are implemented. They enable a perfect reconstruction of the data in the absence of compression and thus ensure that all information is transferred into the wavelet domain. A short overview of the novel wavelet-based compression algorithm is given first, followed by a detailed discussion (also see Figure 17 and Figure 18).

To make KPCA calculations possible on common workstations, 2D wavelet compression of the Gram matrix $\mathbf{K}$ (48) is performed. The compression procedure begins by first compressing the X-dimension followed by compression of the Y-dimension of $\mathbf{K}$. In order to ensure a meaningful wavelet compression in the X-dimension [1], [36], [37], [55], the *same* wavelet coefficients in the X-direction have to be removed from *all* rows of $\mathbf{K}$. If this is not ensured, the subsequent WTs in the Y-dimension would incorporate wavelet coefficients belonging to different positions of the X-dimension WTs. This would cause the final result to be meaningless. In order to determine which irrelevant wavelet coefficients can be removed from all rows, the algorithm has to wavelet transform each row twice. This two step procedure is implemented in order to keep only one vector in memory at a time; this way, very large (Gram) matrices can be handled. During the first step, the algorithm determines which wavelet coefficients are relevant for all rows and which are irrelevant. In the second step, $\mathbf{K}$ is wavelet compressed row-by-row by removing the wavelet coefficients that were

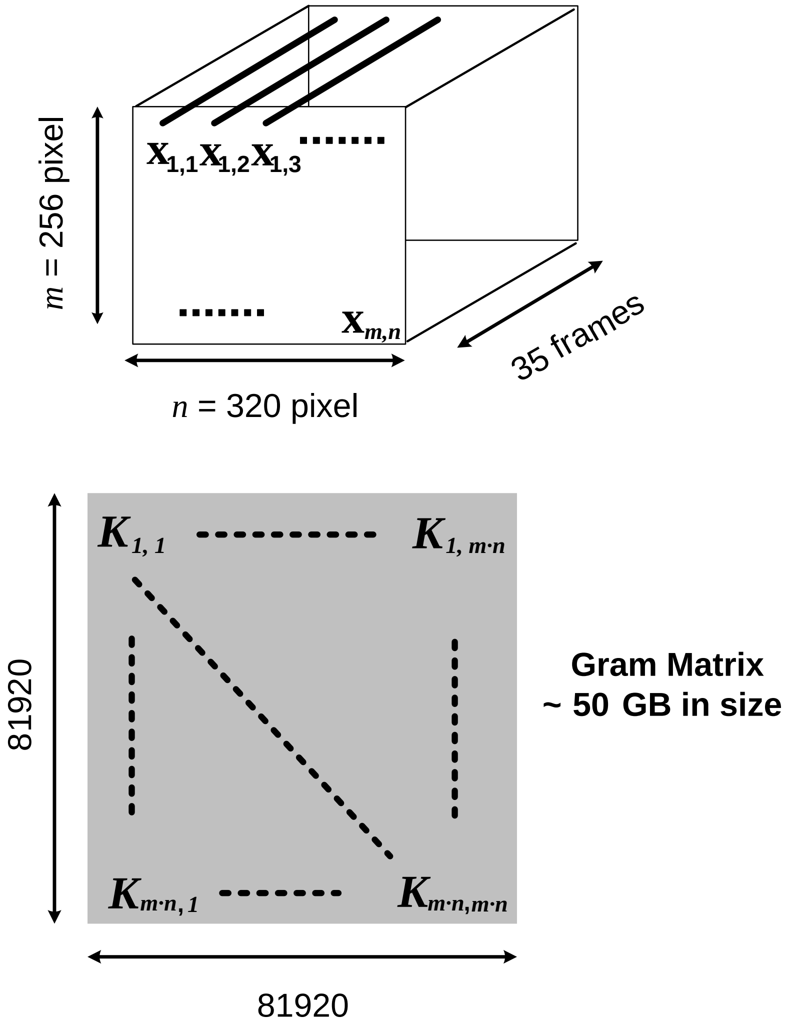
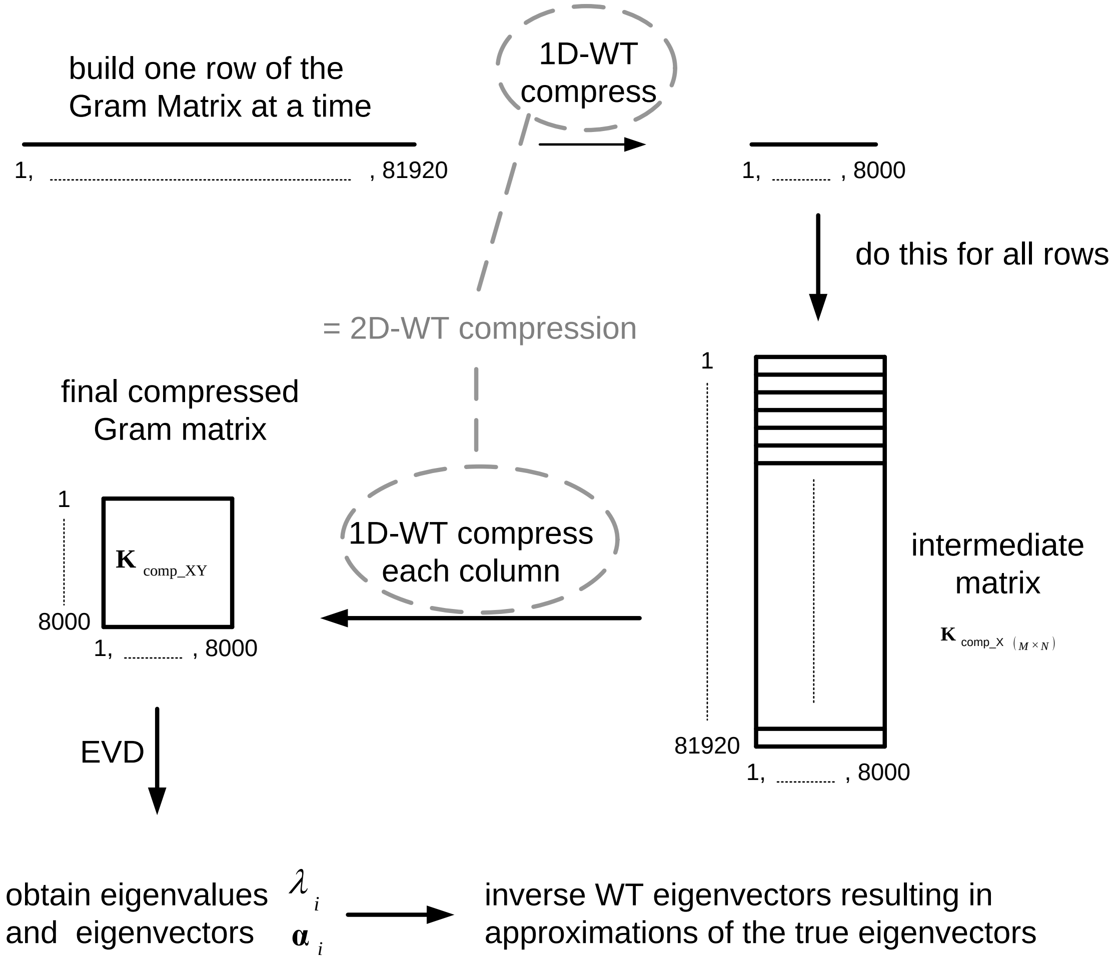


Figure 17: From a spectroscopic image cube (top; refer to Chapter 4.2.2) a Gram matrix $\mathbf{K}$ (47), (48) is derived (bottom, refer to Chapter 2.4). Since Gram matrices are often too large to be handled routinely, an algorithm has been developed that performs data compression and avoids holding such large matrices in memory (Figure 18). From highly compressed Gram matrices an approximation of the KPCA model is computed (see text).



build one row of the Gram Matrix at a time
1, ... , 81920
1D-WT compress
1, ... , 8000
do this for all rows
= 2D-WT compression
final compressed Gram matrix
1
8000
$\mathbf{K}_{\text{comp\_XY}}$
1, ... , 8000
1D-WT compress each column
1
81920
1, ... , 8000
intermediate matrix
$\mathbf{K}_{\text{comp\_X}}\ (M \times N)$
EVD
obtain eigenvalues λ_i and eigenvectors $\boldsymbol{\alpha}_i$
inverse WT eigenvectors resulting in approximations of the true eigenvectors

Figure 18: The main steps of the compression algorithm are: (i) After determining which wavelet coefficients are to be preserved in the X-dimension, one row of the Gram matrix is built (47) at a time. This row is then compressed using a 1D WT and stored in an intermediate matrix $\mathbf{K}_{\text{comp\_X}}$. This is performed for all remaining rows, with each row being compressed the same way. Although $\mathbf{K}_{\text{comp\_X}}$ may be large in the Y-dimension memory requirements have been reduced to a practical level. (ii) Next, each column of $\mathbf{K}_{\text{comp\_X}}$ is compressed using a 1D WT in the Y-dimension. This results in the final compressed Gram matrix $\mathbf{K}_{\text{comp\_XY}}$. This compressed Gram matrix is diagonalized by an EVD (see (48) in Chapter 2.4.1 and (71) in Chapter 4.1.1) to obtain its eigenvalues λ_i and eigenvectors $\boldsymbol{\alpha}_i$ (48). (iv) In order to derive eigenvectors in the uncompressed feature space, two steps are required. First, a de-compression step is applied by inserting zeros at positions where wavelet coefficients were previously removed. Then, a 1D inverse wavelet transform is applied to all eigenvectors $\boldsymbol{\alpha}_i$ individually. The resulting vectors are approximations of the true but often inaccessible eigenvectors. The eigenvalues λ_i in the wavelet domain of the feature space H (40) and the feature space are equal.

found to be irrelevant in the previous step. This two step procedure has been implemented in order to keep only one vector at a time in memory; this way, very large matrices can be handled. The resulting compressed rows are stored in an "intermediate matrix" $\mathbf{K}_{\text{comp\_X}}$ (Figure 18) which is much smaller than $\mathbf{K}$ and thus can be held in memory.

The same procedure is applied to the Y-dimension. The first step wavelet transforms the columns of the intermediate matrix one-by-one and determines which wavelet coefficients can be removed from *all* columns. The second step goes back to the intermediate matrix $\mathbf{K}_{\text{comp\_X}}$ (Figure 18), wavelet transforms its columns again and performs the compression by removing the irrelevant wavelet coefficients. This 2D wavelet compression results in the final compressed Gram matrix (Figure 18, bottom left). The compressed Gram matrix $\mathbf{K}_{\text{comp\_XY}}$ is then diagonalized to derive its eigenvectors $\boldsymbol{\alpha}_i$ and eigenvalues λ_i required for building a KPCA model (44), (48), (51), (52).

Finally, a decompression step utilizing an inverse WT is applied to the eigenvectors obtained in Step iii of Figure 18. This translates the KPCA model from the compressed feature space into the uncompressed feature space (40), (41). Since wavelet coefficients were removed during this procedure, the obtained KPCA model is an approximation of the true uncompressed model. In the remainder of this section, the compression algorithm is discussed in more detail.

In order to determine which wavelet coefficients in the X-dimension can be removed from *all* rows of $\mathbf{K}$, a figure of merit $\mathbf{K}\_\text{row}$ is defined. For this purpose, once a row of $\mathbf{K}$ is built, it is wavelet transformed then added to $\mathbf{K}\_\text{row}$. Subsequently, the next row of $\mathbf{K}$ is transformed and added to $\mathbf{K}\_\text{row}$. This process is continued following (64):

$$\mathbf{K}\_\text{row} = \sum_{i=1}^{M} \left| \text{WT}(\mathbf{K}_i) \right| \quad \text{with} \quad i = i^{th} \text{ row of } \mathbf{K}$$

(64)

Absolute values are added in (64) to avoid large positive and large negative wavelet coefficients, from different rows of $\mathbf{K}$, from canceling each other out. This would incorrectly specify an overall irrelevant wavelet coefficient. Once a row of $\mathbf{K}$ is computed, wavelet transformed, and added to $\mathbf{K}\_\mathrm{row}$, it is removed from the computer's memory. This procedure only requires two vectors of length M (or its next power of two) to be held in memory, i.e. $\mathbf{K}\_\mathrm{row}$ and the currently loaded row of $\mathbf{K}$. This ensures that the minimum amount of computer memory is being used during the compression process, which is essential for enabling routine application of KPCA to large data sets.

If a certain wavelet coefficient is considered to be relevant for a considerable number of rows, its corresponding element in $\mathbf{K}\_\mathrm{row}$ (64) will be large. Again, large (positive and negative) wavelet coefficients are more relevant than small coefficients (see related discussion in Chapter 2.5). If a wavelet coefficient is irrelevant, $\mathbf{K}\_\mathrm{row}$'s corresponding element will be small. Thus, the size of an element of $\mathbf{K}\_\mathrm{row}$ is an indication of the overall importance of a certain wavelet coefficient in the X-dimension.

For measuring the relative importance of a certain element in $\mathbf{K}\_\mathrm{row}$, each element is compared to the sum of $\mathbf{K}\_\mathrm{row}$'s elements. This sum has been named 'total information content (TIC) of X' (X is used in this definition since we are analyzing the rows of $\mathbf{K}$):

$$\mathrm{TIC\ \ of\ \ X} = \sum_{i=1}^{M} \mathrm{K\_row}_i$$

If an element of $\mathbf{K}\_\mathrm{row}$ is small, its contribution to the 'TIC of X' is negligible and thus irrelevant. If, however, an element of $\mathbf{K}\_\mathrm{row}$ contributes considerably to the 'TIC of X', the corresponding wavelet coefficient is relevant and must be retained. In order to adjust the level of compression, the user determines what percentage ($\mathrm{X\%}$) of the 'TIC of X' will be retained during the compression of the rows of $\mathbf{K}$:

$$\%\,\mathrm{retained\,information\ X} = \mathrm{X\%} \cdot \mathrm{TIC\ of\ X} \tag{65}$$

The smaller $\mathrm{X\%}$ is the more compression that is achieved; however, more approximation is introduced into the KPCA model at higher compression levels. $\mathrm{X\%}$ is chosen to be within the range of 50 - 90% to assess the effects various levels of compression have on the final KPCA results.

For determining which wavelet coefficients in $\mathbf{K}\_\mathrm{row}$ (64) are relevant and which are irrelevant, the following procedure is implemented. First, a copy of $\mathbf{K}\_\mathrm{row}$ is made and its elements are sorted in decreasing order. Once the user has determined $\%\text{retained information X}$ (65), a threshold TH_{X} can be established. TH_{X} represents the minimum relevant wavelet coefficient in $\mathbf{K}\_\text{row}$. To calculate TH_{X} , the elements of the sorted copy of $\mathbf{K}\_\text{row}$ are summed. The last added element that makes this sum larger than $\%\text{retained information X}$ is considered to be the smallest relevant element. This value is used as the threshold TH_{X}. Once TH_{X} is known, the elements of $\mathbf{K}\_\text{row}$ are then compared to TH_{X}. Elements in $\mathbf{K}\_\text{row}$ that are smaller than TH_{X} indicate the positions of the irrelevant wavelet coefficients that will be removed from each row of $\mathbf{K}$ during compression. The positions of these irrelevant coefficients are saved in a separate vector named $\text{Remove\_X}_{(M\times 1)}$ (66). The elements of $\text{Remove\_X}_{(M\times 1)}$ are defined as:

$$\text{Remove\_X}_{i=1,\cdots,M} = \begin{cases} 0 & \text{if } i^{\text{th}} \text{ element of } \mathbf{K}\_\text{row} < \text{TH}_{\text{X}} \\ 1 & \text{if } i^{\text{th}} \text{ element of } \mathbf{K}\_\text{row} \geq \text{TH}_{\text{X}} \end{cases}$$

(66)

$\text{Remove\_X}_{(M\times 1)}$ is used for the subsequent compression of $\mathbf{K}$'s M rows, thus deriving $\mathbf{K}_{\text{comp\_X}}$ (Figure 18). Obviously, the number N of wavelet coefficients preserved by the compression of $\mathbf{K}$'s X-dimension equals the sum of $\text{Remove\_X}_{(M\times 1)}$'s elements:

$$N = \sum_{i=1}^{M} \text{Remove \_X}_{i}$$

(67)

Now, all information required for the X-compression of $\mathbf{K}$ is available and can be performed. For this purpose, all rows are consecutively rebuilt (47) and wavelet transformed. This is done because previously only one wavelet transformed row was held in memory in order to limit memory requirements. Next, a row of $\mathbf{K}$ is compressed by copying the wavelet coefficients which have been marked by a ‘one’ in Remove_X (66) into the appropriate positions of the “intermediate” matrix $\mathbf{K}_{\text{comp\_X}(M\times N)}$ (where $N < M$, see Figure 18); all other elements are discarded.

So far, a detailed description of the compression of $\mathbf{K}$ in the X-dimension has been given. This procedure involves building and compressing each row of $\mathbf{K}$ one at a time. This approach keeps memory requirements to a minimum because the Gram matrix $\mathbf{K}$ is never handled in its entirety, thus the compression process remains computationally efficient. Now that all rows of $\mathbf{K}$ are compressed, giving $\mathbf{K}_{\text{comp\_X}}(M \times N)$, the columns of $\mathbf{K}_{\text{comp\_X}}(M \times N)$ will be compressed; hence 2D wavelet compression (see Figure 18).

The next step performs compression of the columns (Y-dimension) of $\mathbf{K}_{\text{comp\_X}}$ in a way similar to the compression of $\mathbf{K}$'s X-dimension. This will result in a final compressed Gram matrix $\mathbf{K}_{\text{comp\_XY}}$ (Figure 18). The algorithm copies the columns of $\mathbf{K}_{\text{comp\_X}}$ one at a time, wavelet transforms them, and adds their absolute values to a vector named $\mathbf{K}\_\text{col}(M \times 1)$ (compare (64) and related discussion):

$$\mathbf{K}\_\text{col}(M \times 1) = \sum_{i=1}^{N} \left| \text{WT}\left(\mathbf{K}_{\text{comp\_X}_i}\right) \right| \quad \text{with} \quad i = i^{th} \text{ column of } \mathbf{K}_{\text{comp\_X}} \tag{68}$$

After a column of $\mathbf{K}_{\text{comp\_X}}$ is copied, it is wavelet transformed, added to $\mathbf{K}\_\text{col}$ and then immediately deleted from memory. The next column $\mathbf{K}_{\text{comp\_X}}$ is then analyzed following the same procedure. Note, only the copies of the columns of $\mathbf{K}_{\text{comp\_X}}$ are deleted; not the original columns themselves. This procedure only requires enough memory to store both $\mathbf{K}\_\text{col}(M \times 1)$ (68) and one wavelet transformed column (copy) of $\mathbf{K}_{\text{comp\_X}}$. Again, the amount of required computer memory needs to be kept to a minimum to make the calculations feasible.

Once this procedure is finalized, a threshold TH_Y is determined following a similar approach leading to (65) and (66). Likewise, $\%\text{retained information Y}$ is defined as:

$$\%\text{retained information Y} = \text{Y\%} \cdot \text{total information content of Y} \tag{69}$$

with $\text{Y\%}$ being the user-selected amount of information to be retained during compression of the Y-dimension of $\mathbf{K}_{\text{comp\_X}}$ (compare (65)). $\text{Y\%}$ is typically chosen to be within the range of 50 - 90%. A vector named $\text{Remove\_Y}(M \times 1)$ similar to (66) is defined to store the positions of the relevant and irrelevant wavelet coefficients in $\mathbf{K}\text{\_col}_{(M\times 1)}$. The positions of the irrelevant coefficients (indicated by ‘0s’, see (66)) in $\mathbf{K}\text{\_col}_{(M\times 1)}$ correspond to entire columns in $\mathbf{K}_{\text{comp\_X}}$ that will be removed during compression. The sum of Remove_Y ’s elements:

$$L = \sum_{i=1}^{M} \text{Remove \_Y}_i$$

(70)

equals the size of the Y-dimension of the final compressed Gram matrix $\mathbf{K}_{\text{comp\_XY}(L\times N)}$. It is common that $L \neq N$. This is explained by the fact that the second compression step is performed on an already partially compressed matrix $\mathbf{K}_{\text{comp\_X}}$. Remove_Y is now used to compress the columns of $\mathbf{K}_{\text{comp\_X}}$. Each column of $\mathbf{K}_{\text{comp\_X}}$ is now copied and wavelet transformed again one-by-one. These same columns were previously deleted from memory after being used in (68) to determine which wavelet coefficients can be removed. Once a column is transformed, every element is compared to the corresponding element of Remove_Y . Only when $\text{Remove\_Y}_{i=1,\cdots,M} = 1$ does the corresponding wavelet coefficient in the column of $\mathbf{K}_{\text{comp\_X}}$ get copied, at the appropriate positions, into the final compressed Gram matrix $\mathbf{K}_{\text{comp\_XY}(L\times N)}$.

$\mathbf{K}_{\text{comp\_XY}}$ then undergoes an eigenvalue decomposition (EVD) (71) to derive its eigenvalues λ_i and eigenvectors $\boldsymbol{\alpha}_i$ (see Chapter 2.4 and (48)). If $L = N$ then $\mathbf{K}_{\text{comp\_XY}}$ is square and both sets of eigenvectors contained in the matrices $\mathbf{V}$ and $\mathbf{V}^{\mathbf{T}}$ are identical. This is the standard EVD procedure. If, however, $L \neq N$ the EVD is based on a singular value decomposition (SVD) [52], (39).

$$\begin{aligned} \text{case } L = N: \quad & \mathbf{K}_{\text{comp\_XY}(N\times N)} \overset{\text{EVD}}{=} \mathbf{V}_{(N\times N)} \cdot \Sigma_{(N\times N)} \cdot \mathbf{V}^{\mathbf{T}}_{(N\times N)} \\ \text{case } L \neq N: \quad & \mathbf{K}_{\text{comp\_XY}(L\times N)} \overset{\text{SVD}}{=} \mathbf{U}_{(L\times N)} \cdot \Sigma_{(N\times N)} \cdot \mathbf{V}^{\mathbf{T}}_{(N\times N)} \end{aligned}$$

(71)

If $L > N$, the columns of $\mathbf{U}$ are longer than the rows of $\mathbf{V}^{\mathbf{T}}$. Therefore, these columns are chosen as the eigenvectors $\boldsymbol{\alpha}_i$ (48) because more information has been retained in

$\mathbf{U}$. If $L < N$, then more information is preserved in the rows of $\mathbf{V}^{\mathbf{T}}$ and, in this case, these rows are used as the eigenvectors. Using the ‘longer’ set of eigenvectors will reduce approximation error.

Regardless of which set of eigenvectors is used, the compressed eigenvectors are still in the wavelet domain. Therefore, the eigenvectors need to be 1) uncompressed and then 2) inverse wavelet transformed to bring them back into the original feature space. The eigenvectors are uncompressed by inserting zeros into the positions where wavelet coefficients were previously removed. The latter information can be retrieved from Remove_X (67) if $\mathbf{V}^{\mathbf{T}}$ was utilized or from Remove_Y if $\mathbf{U}$ was selected.

4.1.2 Incorporating Mean-Centering into Wavelet Compressed KPCA

One standard data pre-processing procedure is to mean-center [9] the data vectors contained in the calibration set. Mean-centering removes any common background that may be present in the data; this common feature does not contain any chemical information. For KPCA, the conventional mean centering approach would require computing:

$$\tilde{\Phi}(x_k) = \Phi(x_k) - \frac{1}{M} \cdot \sum_{i=1}^{M} \Phi(x_i) \quad \text{for all} \quad k = 1, \cdots, M$$

(72)

However, due to the lengths (42) of these vectors this procedure can often not be applied. To circumvent this, a mathematically equivalent technique has been developed [39], [40], that directly operates on the Gram matrix $\mathbf{K}_{(M \times M)}$ (48) and derives a mean-centered version $\tilde{\mathbf{K}}$ (73):

$$\tilde{\mathbf{K}}_{(M \times M)} = \mathbf{K} - \mathbf{1}_M \cdot \mathbf{K} - \mathbf{K} \cdot \mathbf{1}_M + \mathbf{1}_M \cdot \mathbf{K} \cdot \mathbf{1}_M$$

(73)

with:

$$\mathbf{1}_M = \begin{pmatrix} 1/M & \cdots & 1/M \\ \vdots & \ddots & \vdots \\ 1/M & \cdots & 1/M \end{pmatrix}_{\left(M\times M\right)}$$

(74)

For large data sets (i.e. large M ; see Chapters 4.2.1 and 4.2.2), implementing (73) can require multiple gigabytes of memory. Therefore, a new algorithm is derived here that enables the element-wise mean-centering of $\mathbf{K}$ and thus avoids handling multiple-gigabyte matrices.

- First term on the right-hand side of (73): One element of the Gram matrix $\mathbf{K}$ is derived by means of (47) and (50).
- Second term: All rows of $\mathbf{1}_M \cdot \mathbf{K}$ are equal because the columns of $\mathbf{K}$ are multiplied with rows of $\mathbf{1}_M$ (74) which contain identical elements. Thus, calculating one row of $\mathbf{1}_M \cdot \mathbf{K}$ is sufficient for element-wise mean centering. This vector of length M is denoted MeanCenter_vector. Its elements are determined by performing dot products of the first (or any) row of $\mathbf{1}_M$ (74) and the corresponding columns of $\mathbf{K}$:

$$\begin{aligned} \text{MeanCenter\_vector}_k &= \sum_{j=1}^{M} \frac{1}{M} \cdot K_{j,k} = \frac{1}{M} \cdot \sum_{j=1}^{M} \left\langle \Phi\left(\mathbf{x}_j\right), \Phi\left(\mathbf{x}_k\right) \right\rangle \quad \text{with:} \quad k = 1, \ldots, M \\ &= \frac{1}{M} \cdot \sum_{j=1}^{M} \left\langle \mathbf{x}_j, \mathbf{x}_k \right\rangle^d \qquad \text{for polynomial kernels; equ. (50)} \end{aligned}$$

(75)

- Third term: $\mathbf{K} \cdot \mathbf{1}_M$ is the transpose of $\mathbf{1}_M \cdot \mathbf{K}$ because $\mathbf{1}_M$ (73) has the same value at each position (74) and $\mathbf{K}$ is symmetric. While $\mathbf{1}_M \cdot \mathbf{K}$ consists of identical rows, $\mathbf{K} \cdot \mathbf{1}_M$ consists of identical columns and these rows and columns are identical vectors. Thus, MeanCenter_vector (75) can also be used to incorporate the contribution of $\mathbf{K} \cdot \mathbf{1}_M$ to the mean centering process.
- All elements of the fourth term $\mathbf{1}_M \cdot \mathbf{K} \cdot \mathbf{1}_M$ are identical. This element is determined by computing the dot product between $\text{MeanCenter\_vector} = \mathbf{1}_M \cdot \mathbf{K}$ (75) and a column of $\mathbf{1}_M$ (74). Since all columns of $\mathbf{1}_M$ are identical, only one value, referred

to as MeanCenter_number, needs to be determined in order to gain the complete information contained in $\mathbf{1}_M \cdot \mathbf{K} \cdot \mathbf{1}_M$:

$$\begin{aligned}\text{MeanCenter\_number} &= \sum_{k=1}^{M} \text{MeanCenter\_vector}_k \cdot \frac{1}{M} \\ &= \frac{1}{M^2} \cdot \sum_{k=1}^{M} \sum_{j=1}^{M} \left\langle \Phi(\mathbf{x}_j), \Phi(\mathbf{x}_k) \right\rangle \end{aligned} \tag{76}$$

In conclusion (47), (50), (73) - (76):

$$\tilde{K}_{i=1\cdots M\ ,\ j=1\cdots M} = K_{i,j} - \text{MeanCenter\_vector}_j - \text{MeanCenter\_vector}_i + \text{MeanCenter\_number} \tag{77}$$

As explained in bullet number two, the contribution of $\mathbf{1}_M \cdot \mathbf{K}$ does depend on the column index j but *not* on the row index i since all rows of $\mathbf{1}_M \cdot \mathbf{K}$ are identical. Similarly (see bullet number three), the contribution of $\mathbf{K} \cdot \mathbf{1}_M$ is *independent* of the column index j, since all columns of $\mathbf{K} \cdot \mathbf{1}_M$ are equal, and only depends on the row index i.

Utilizing (77) enables the generation of $\tilde{\mathbf{K}}$ row-wise. This row is then wavelet compressed and written into the intermediate matrix $\mathbf{K}_{\text{comp\_X}}$ (see Chapter 4.1.1 and Figure 18). From there, the compression based KPCA follows the procedure introduced in Chapter 4.1.1. Thus, mean centering has been incorporated into the wavelet compression.

After finalizing the calibration, Q unknown spectra in feature space $\Phi(\mathbf{x}_{\text{unknown }1,\cdots,Q})$ have to be evaluated (see Chapter 2.4.2). For that purpose, $\mathbf{t}$-vectors are calculated, which are equivalent to PCA's score vectors (see (51), (52) and related discussion). But prior to that, the $\Phi(\mathbf{x}_{\text{unknown }1,\cdots,Q})$ must be mean-centered, too. Again, the $\Phi(\mathbf{x}_{\text{unknown }1,\cdots,Q})$ are very long (42) and would require computation resources that are often unavailable. In order to avoid direct mean centering (72) of the vectors $\Phi(\mathbf{x}_{\text{unknown }1,\cdots,Q})$ another Gram matrix denoted $\mathbf{K}^{\text{unknown}}_{(Q\times M)}$ is introduced in equation (51) and (52). A similar approach [39], [40], as utilized in (73), derives a mean centered version $\tilde{\mathbf{K}}^{\text{unknown}}$ of $\mathbf{K}^{\text{unknown}}$:

$$\widetilde{\mathbf{K}}^{\text{unknown}} = \mathbf{K}^{\text{unknown}} - \mathbf{1}'_M \cdot \mathbf{K} - \mathbf{K}^{\text{unknown}} \cdot \mathbf{1}_M + \mathbf{1}'_M \cdot \mathbf{K} \cdot \mathbf{1}_M \qquad (78)$$

In (78), $\mathbf{1}'_M$ denotes a $Q \times M$ matrix whose entries are all equal to $1/M$ (compare (74)). All other matrices have been defined previously.

Again, some matrices in (78) can require several gigabytes of memory and thus handling the full equation must be avoided. This is achieved by mean centering $\mathbf{K}^{\text{unknown}}$ element-wise following the same strategy when mean centering $\mathbf{K}$ (73), (77).

- The first term on the right-hand side of (78) is the Gram matrix $\mathbf{K}^{\text{unknown}}$ (52). An element of $\mathbf{K}^{\text{unknown}}$ is computed by utilizing (51).
- The second, $\mathbf{1}'_M \cdot \mathbf{K}$, and the fourth term, $\mathbf{1}'_M \cdot \mathbf{K} \cdot \mathbf{1}_M$, contain the same values as the corresponding terms $\mathbf{1}_M \cdot \mathbf{K}$ and $\mathbf{1}_M \cdot \mathbf{K} \cdot \mathbf{1}_M$ in equation (73). The only difference is that they have a different number of rows (which are all identical) – they are of dimensions $Q \times M$. Therefore, MeanCenter_vector (75) and MeanCenter_number (76) can be used again.
- Since $\mathbf{K}^{\text{unknown}}$ (51), (52) is not equal to $\mathbf{K}$, the third term, $\mathbf{K}^{\text{unknown}} \cdot \mathbf{1}_M$, needs to be calculated. This follows a similar approach that led to (75). A vector denoted MeanCenter_unknown_vector of length Q is derived by:

$$\begin{aligned}\text{MeanCenter\_unknown\_vector}_k &= \sum_{j=1}^{M} \left\langle \Phi(\mathbf{x}_{\text{unknown } k}), \Phi(\mathbf{x}_j) \right\rangle \cdot \frac{1}{M} \qquad \text{with:} \quad k = 1, \ldots, Q \\ &= \frac{1}{M} \cdot \sum_{j=1}^{M} \left\langle \mathbf{x}_{\text{unknown } k}, \mathbf{x}_j \right\rangle^d \\ &\qquad\qquad \text{for polynomial kernels; equ. (50)}\end{aligned} \qquad (79)$$

Finally, using MeanCenter_vector (75), MeanCenter_number (76), and MeanCenter_unknown_vector (79), the elements of $\mathbf{K}^{\text{unknown}}_{(Q \times M)}$ are mean centered individually to give $\widetilde{\mathbf{K}}^{\text{unknown}}_{(Q \times M)}$:

$$\widetilde{K}^{\text{unknown}}_{i,j} = K^{\text{unknown}}_{i,j} - \text{MeanCenter\_vector}_j - \text{MeanCenter\_unknown\_vector}_i + \text{MeanCenter\_number} \qquad (80)$$

In conclusion, the use of equations (77) and (80) instead of (73) and (78) provides a way to mean center the data in the feature space without requiring unfeasible amounts of computer memory. $\tilde{\mathbf{K}}^{\text{unknown}}_{(Q \times M)}$ is then evaluated by means of (52).

4.2 Experimental

To test the performance and robustness of the compression algorithm developed in this thesis, two spectroscopic imaging data sets are utilized which were acquired with two different experimental setups. Both data sets were mean-centered prior to all calculations following the procedure outlined in Chapter 4.1.2. Daubechies-8 wavelets [19], [52] are utilized for the X and Y-compression of the Gram matrix $\mathbf{K}$ (48).

4.2.1 “Bacteria Data”

Data set #1 is a spectroscopic image cube of *E*. *coli K12* bacteria obtained by FTIR microscopy (see Figure 2 but note that *E*. *coli B* is used for this figure). The cube’s Z-dimension holds the spectral information acquired from different X and Y-positions of the sample. A Bruker Optics Vertex 70 spectrometer combined with a Hyperion 1000 IR/VIS microscope featuring a 64 x 64 pixel MCT focal plane array and a 15x objective is utilized in this experiment. Spectra covered the wavenumber range (1922 - 960 cm^{-1}) containing 500 data points per spectrum. Thus, the Gram matrix produced by this instrument has dimensions 4096 x 4096 (~128 MB). Although this is a rather small data set, uncompressed KPCA computations took about one day.

4.2.2 “Remote Sensing Data”

Data set #2, having dimensions X = 320, Y = 256, Z = 35, was acquired via remote sensing; details on the experimental setup are discussed in references [36] and [59] and in Chapter 3.2. This data set consists of 81,920 spectra each containing 35 wavelength positions covering the 3 - 5 µm atmospheric window. In the Results section below, KPCA models derived from both the compressed and the conventional, uncompressed approach will be compared. Based on these comparisons the quality of the compression based KPCA method will be assessed. The uncompressed Gram matrix contains ~50 GB of data and is therefore too large for applying conventional KPCA using the aforementioned workstation. Hence, for proof-of-principle calculations, the Gram matrix was restricted to dimensions 12,100 x 12,100 (~1.1 GB). However, the workstation can perform the calculations on the full data set when the compression based algorithm is applied.

4.2.3 Compression Algorithm Parameter Settings

The main focus will be on the results pertaining to data set #1 (see Chapter 4.2.1). Comparable results were obtained for data set #2 (see Chapter 4.2.2). Several parameters (i.e. kernel type, kernel variables, wavelet type, compression level, etc.) can be adjusted for each application. For proof-of-principle, investigations have been limited to polynomial kernels (40), (41) of exponent order three. However, similar results were observed in studies utilizing a Gaussian radial basis kernel applied to data set #1.

To both data sets a conventional, uncompressed KPCA is applied. These results are then compared to those obtained by the compression-based algorithm which makes use of a Daub8 wavelet. The following levels of compression have been investigated:

Compression level 1: 90% retained information (both X and Y dimension, see (65) and (69))

Compression level 2: 80% retained information (both X and Y dimension)

Compression level 3: 70% retained information (both X and Y dimension)

Compression level 4: 60% retained information (both X and Y dimension)

Compression level 5: 50% retained information (both X and Y dimension)

(81)

4.3 Results and Discussion

To evaluate the capabilities of the compression-based algorithm, comparisons are made to the uncompressed approach based on the following figures of merit:

‘reduction in memory requirements’, ‘quality of compression-based models’, and ‘gain in computation speed’. The following sections discuss each figure of merit individually.

4.3.1 Reducing Memory Requirements

The memory requirements for the compressed Gram matrices (81) are compared to the amount of memory needed to hold the uncompressed Gram matrix. Since each element of a Gram matrix is represented by an 8-bytes floating point number, the total amount of memory needed to store a Gram matrix is determined by:

$$\text{memory requirement of } \mathbf{K} = \text{X dimension of } \mathbf{K} \cdot \text{Y dimension of } \mathbf{K} \cdot 8 \text{ bytes} \tag{82}$$

Figure 19 depicts the reduction in memory requirements for data set #1 after application of each of the five increasing compression levels (81). The true Gram matrix for data set #1 requires 128 MB of memory. Compression level 1 results in a Gram matrix that is approximately 10% (~13 MB) the size of the original Gram matrix. Thus, a considerable decrease in memory requirements can be achieved by applying only a low compression. This trend continues with increasing compression level resulting in a final compressed Gram matrix size of ~0.3 MB (compression level 5).

4.3.2 Analysis of the Quality of the Compression Results

Two figures of merit are required to assess the quality and accuracy of the compression-based results: (i) t-values (51), (52) obtained from the compression-based KPCA method are compared to their uncompressed equivalents. These *t*-values are equivalent to the ‘scores’ in PCA (see (37)). For this purpose, “relative errors” (or deviations) between the two sets of results are calculated. (ii) “Correlation coefficients” are calculated between the uncompressed and compressed eigenvectors $\boldsymbol{\alpha}_i$ (48) of the Gram matrix to measure the similarity of both sets of vectors.

Relative errors: Since different compression methods (81) derive different eigenvectors $\boldsymbol{\alpha}_i$ and eigenvalues λ_i (48), each compression method computes a different $\mathbf{t}_{(R\times 1)}$ (51) or $\mathbf{T}_{(Q\times R)}$ (52). In order to assess the quality of a compression

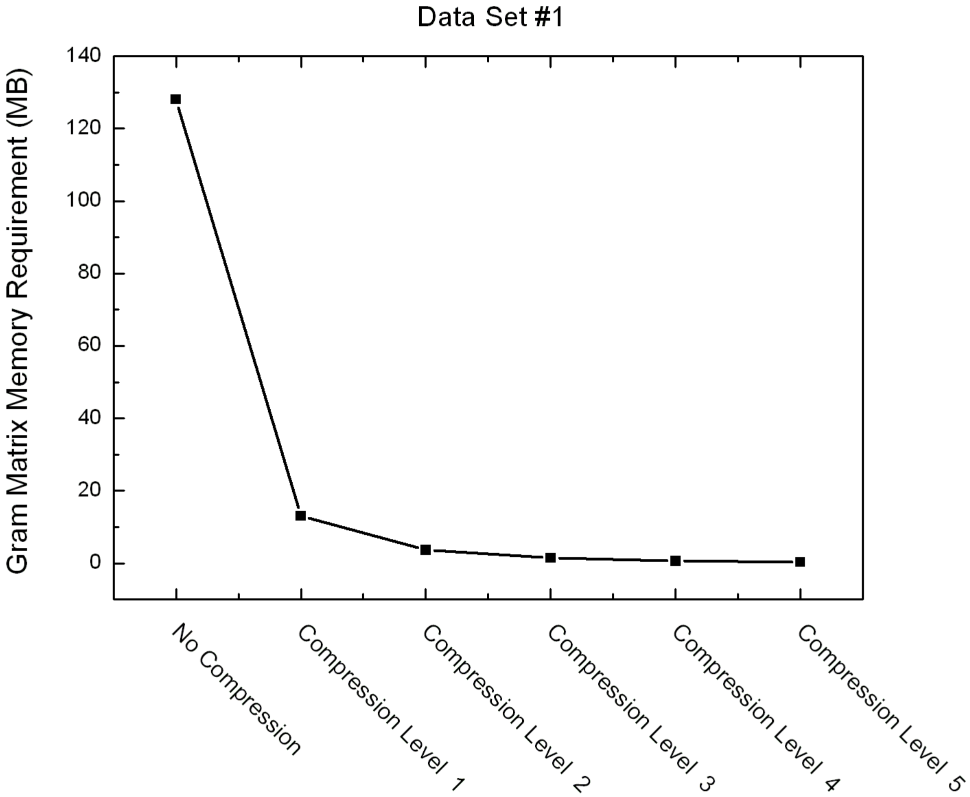


Figure 19: Memory requirements for storing the Gram matrix of data set #1 in the absence of compression and at five increasing compression levels (81).

method a figure of merit has to be defined that measures how close the approximation-based results are to the uncompressed results. For this purpose, $\mathbf{T}_{\left(Q\times R\right)}^{\text{uncompressed}}$ and $\mathbf{T}_{\left(Q\times R\right)}^{\text{compressed}}$ are computed from the uncompressed and compression-based KPCA methods, respectively. Also, only for comparison purposes, the 4096 spectra contained in data set #1 will be used, again, as the ‘unknown’ test vectors for calculating both $\mathbf{T}_{\left(Q\times R\right)}^{\text{uncompressed}}$ and $\mathbf{T}_{\left(Q\times R\right)}^{\text{compressed}}$ (see (51) and (52) and related discussion). Relative errors (or deviations) are calculated by:

$$\text{relative error}_{j,k} = \left| \frac{T_{j,k}^{\text{uncompressed}} - T_{j,k}^{\text{compressed}}}{T_{j,k}^{\text{uncompressed}}} \right|$$

$$\text{for :}$$

$$j = 1 \ldots 4{,}096 \quad \text{\# of unknown test vectors} = \text{\# of calibration vectors (for this study only)}$$

$$k = 1 \ldots R \qquad R = \text{\# of relevant eigenvectors (or \# of score elements in each vector } \mathbf{t}_{\left(R\times 1\right)}\text{)}$$

(83)

One method for analyzing a large number of relative errors is to plot them in a histogram with the x-axis being relative error. For this reason, absolute values are calculated using (83). It can be anticipated that with increasing compression (81) the relative errors (83) will also increase. That is, in the histograms the center of the distribution will shift to high error values. For each of the five compression levels, Figure 20 displays the error distributions for all $j = 1 \ldots 4{,}096$ $T_{k=1}$-values obtained from data set #1. For compression levels 1 and 2 it is obvious that the maximum population exists at rather low error values. At compression level 3, the maximum population begins to shift to higher error values. This trend continues throughout compression levels 4 and 5 indicating that more and more approximation is being introduced. Figure 21 displays the error distribution for all $j = 1 \ldots 4{,}096$ $T_{k=3}$-values obtained after applying all five compression levels. Based on the strong shift to higher relative error values at compression levels 4 and 5 it can be concluded that too much approximation is introduced after compression level 3. However, increasing compression from level 3 to 4 does not offer much in terms of reduction in memory requirements for storing the Gram matrix (Figure 19). At this point, only a minute reduction in required memory (~1.4 MB to ~0.6 MB) is achieved for data set #1. Thus, a small improvement memory-wise can result in a considerable decrease in model quality. Since there is an obvious limitation of

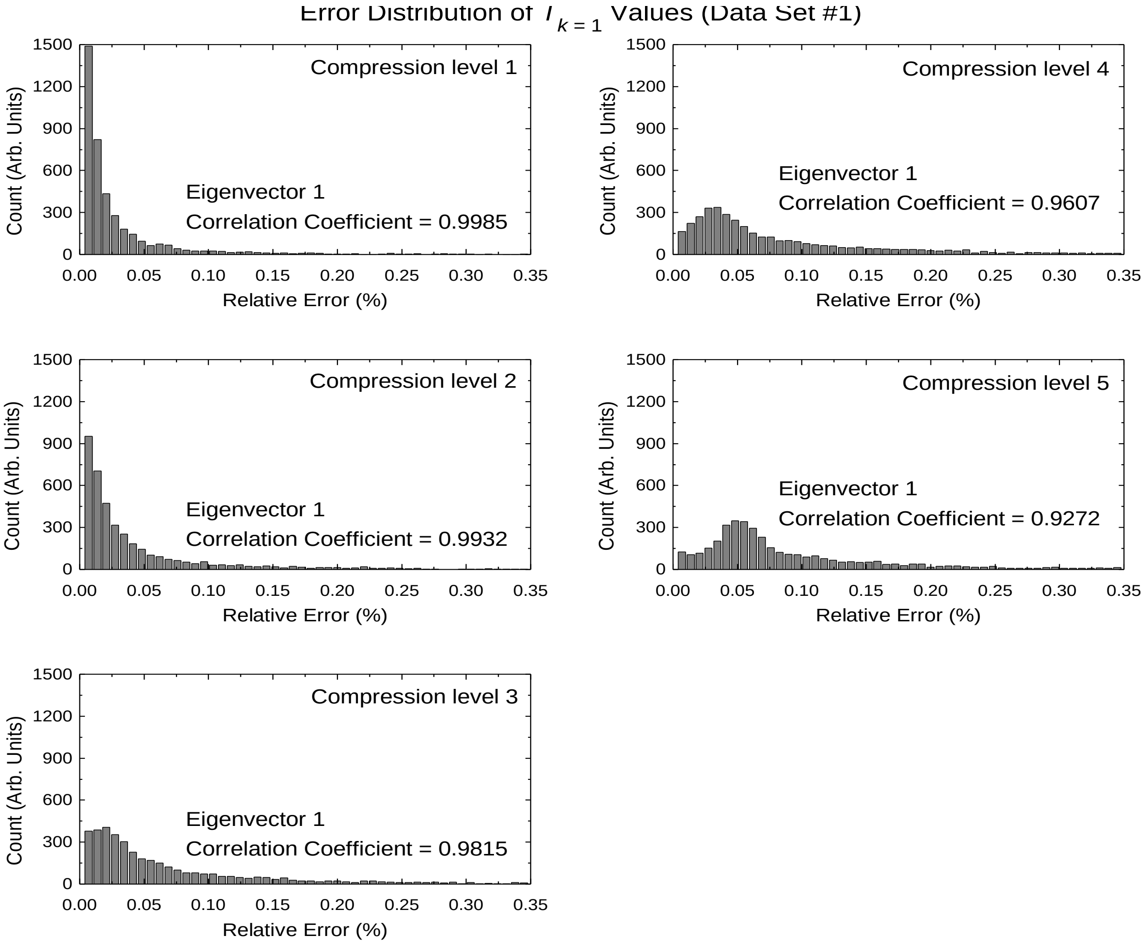


Figure 20: Error distributions of the $j = 1 \ldots 4{,}096$ $T_{k=1}$-values (51) for data set #1 (Chapter 4.2.1) at five increasing compression levels (81). Also, correlation coefficients (Chapter 4.3.2) between the uncompressed and compressed eigenvectors $\boldsymbol{\alpha}_{i=1}$ (48) of the Gram matrix are given for each compression level. These correlation coefficients are a measure of the closeness of the compression-based eigenvectors to the true, uncompressed eigenvectors.

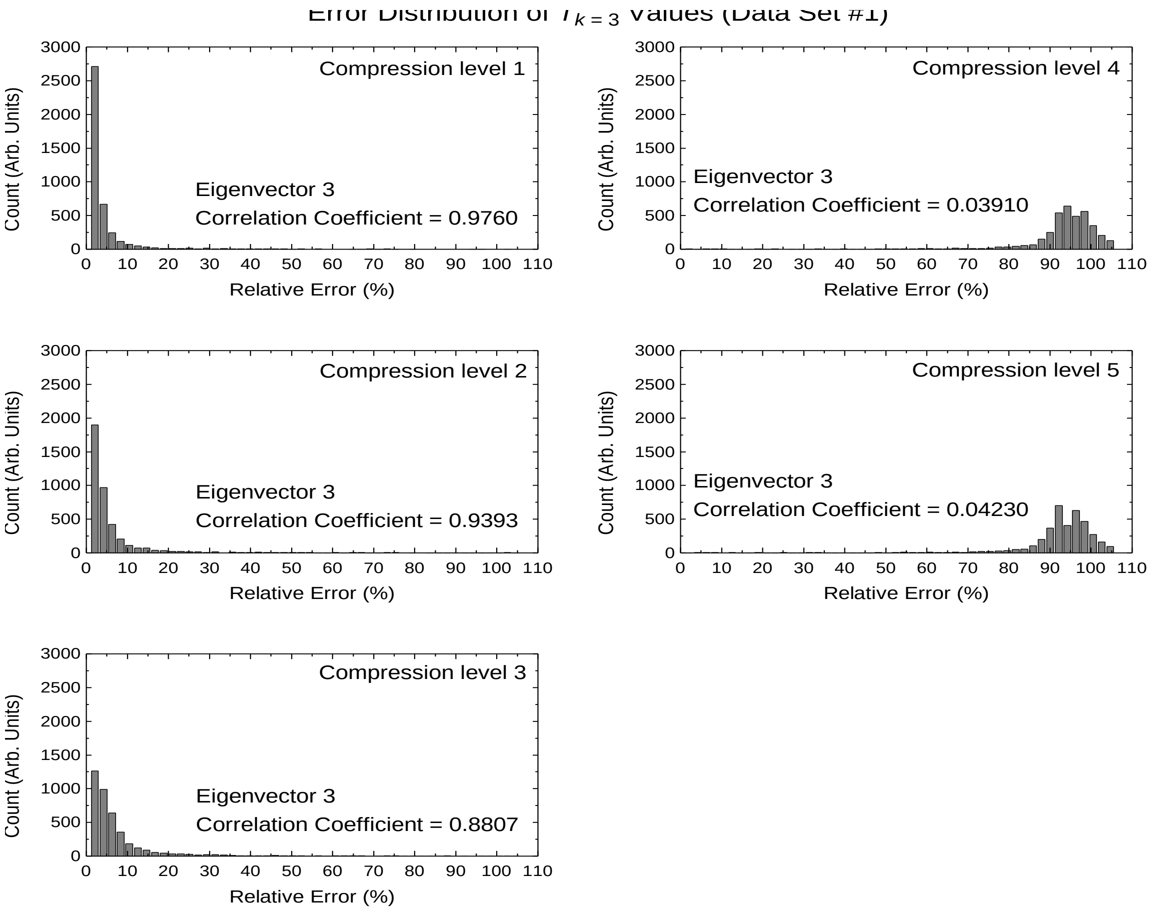


Figure 21: Error distribution of the $j = 1\ldots4{,}096$ $T_{k=3}$-values (51) for data set #1 (Chapter 4.2.1) at five increasing compression levels (81). Also shown are the correlation coefficients (Chapter 4.3.2) between the uncompressed and compressed eigenvectors $\boldsymbol{\alpha}_{i=3}$ (48) of the Gram matrix. These correlation coefficients are a measure of the closeness of the compression-based eigenvectors to the true, uncompressed eigenvectors.

how much compression can be applied and still result in useful models, compression must be adapted to a specific application.

Correlation: Correlation coefficients are used as an indication of how similar eigenvectors $\boldsymbol{\alpha}_i$ (48) of the compressed Gram matrix are to their uncompressed counterparts. A decrease in correlation coefficient is expected as the level of compression increases. In Figure 20 and Figure 21, the correlation coefficients obtained for $\boldsymbol{\alpha}_{i=1}$ and $\boldsymbol{\alpha}_{i=3}$ of data set #1 (see Chapter 4.2.1) are given in the respective graphs. The correlation coefficient for $\boldsymbol{\alpha}_{i=1}$ (Figure 20) at compression level 1 is 0.9985 which indicates that the compressed eigenvector is a very good approximation of the true (but often inaccessible) eigenvector. Good correlation coefficients (0.9932 and 0.9815, respectively) are also obtained for the next higher compression levels. Compression levels 4 and 5 still resulted in good correlation coefficients although a slight decline in value is apparent. A somewhat different trend in correlation coefficients is observed for the third eigenvector $\boldsymbol{\alpha}_{i=3}$ (Figure 21). Only for compression levels 1 and 2 are good correlation values obtained. For higher compression levels the correlation between the compressed and uncompressed eigenvector quickly declines. Compression level 3 results in a correlation coefficient of only 0.8807 which can be considered a borderline case. Nonetheless, this level of compression still features a high population at the lower relative error values (83). For compression levels 4 and 5 the quality of the *t*-values quickly drops. In conclusion, there certainly exists a limitation of compression that should be applied. In the studied cases, this limitation was found to exist at compression levels that allow for a considerable amount of data reduction (Figure 19) while still producing reliable results (Figure 20 and Figure 21). Also, it was found that limiting the compression to level 3 was sufficient for adequately reducing the amount of required memory to enable KPCA to become feasible for large data sets.

4.3.3 Decrease in Computation Time

Aside from making KPCA possible on common personal computers, significant decreases in computation time are achieved as well. For demonstration and assessment purposes an acceleration factor is defined as:

$$\text{acceleration factor} = \frac{\text{comp. time(conventional KPCA of full Gram matrix)}}{\text{comp. time(WT compression of true Gram matrix} + \text{EVD of compressed Gram matrix} + \text{inverse WTs)}} \quad (84)$$

Acceleration factors are ratios between the computation time needed for performing a conventional KPCA and that required for the compression-based KPCA approach. Figure 22 displays the acceleration factors achieved at each of the five compression levels for both data sets. Both figures show a significant increase in acceleration as compression is increased. For data set #1 a maximum acceleration factor of ~22 is obtained; a maximum acceleration factor of ~200 is calculated for data set #2. The larger acceleration factor for data set #2 is attributed to the larger compression that is achieved for this data set. However, both of these values correspond to compression level 5, which is considered to be too much compression for these data sets. For compression level 3, which was determined to be the optimal amount of compression (see Chapter 4.3.2), acceleration factors of ~10 and ~90 are calculated for data sets #1 and #2, respectively.

4.4 Conclusions

Many chemometric methods are only capable of evaluating linear relationships within data. This is insufficient for many applications and thus kernel principal component analysis (KPCA) has been introduced as a means to model nonlinear data [39], [40]. However, one major drawback of KPCA is that it involves the eigenvalue decomposition of a covariance matrix, or 'Gram matrix', which has dimensions # of samples by # of samples. This 'large' covariance matrix is decomposed in order to derive the maximum number of principal components produced by KPCA [39], [40]. This amount of principal components can exceed the number of variables (wavelength positions) if the number of samples (spectra) is larger. This is not possible in PCA. This larger number of principal components achieved with KPCA could lead to a more enhanced understanding of complex data. Applying KPCA to large data sets, as encountered in spectroscopic imaging, often requires computational resources, specifically memory and computation speed, that are not commonly available.

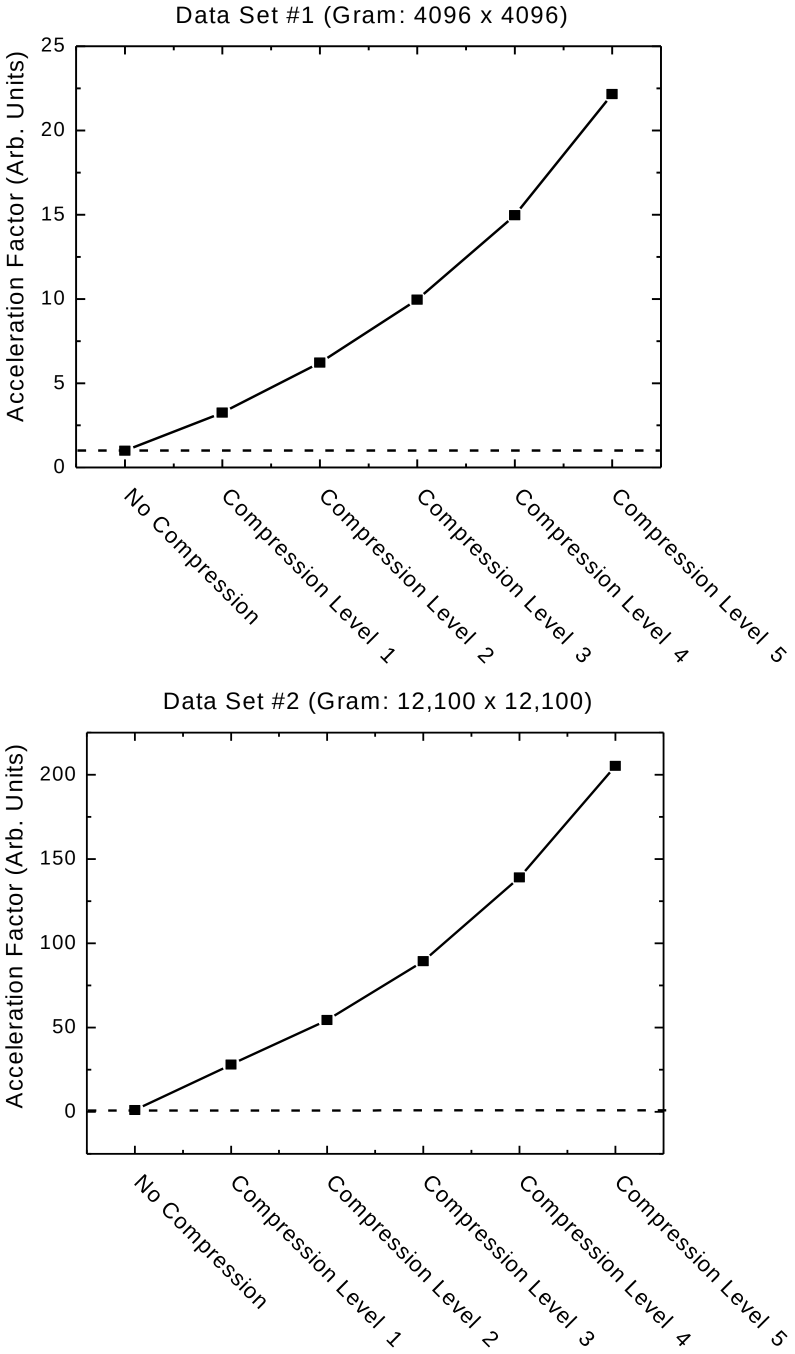


Figure 22: Acceleration factors (84) for data set #1 (top) and data set #2 (bottom) at each of the five compression levels (81). The dashed line in each plot represents acceleration factor = 1, which indicates no acceleration.

In this study, an algorithm is developed, implemented, and assessed which applies a two dimensional (2D) wavelet compression to the Gram matrix. Direct compression of the full Gram matrix cannot be accomplished since this would still require holding the entire, multiple-gigabyte matrix in memory. This novel approach is based on building and compressing one row of the Gram matrix at a time and thus only requires amounts of memory that are realistic for workstations. Also, a novel element-wise mean-centering procedure is developed and is incorporated into the compression. Now, KPCA can be routinely applied to large (spectroscopic imaging) data sets.

Two data sets acquired with two different experimental setups are utilized to demonstrate the compression capabilities and robustness of the proposed algorithm. The final sizes of the compressed Gram matrices can be handled on common desktop computers. The quality of the compressed results is analyzed by plotting relative error histograms between corresponding t-values (51), (52) obtained from both the uncompressed and compressed KPCA methods. The compressed results are further assessed by determining correlation coefficients between the true uncompressed eigenvectors $\boldsymbol{\alpha}_i$ (48) and the compressed eigenvectors. Different compression levels are studied in order to determine the amount of compression that can be applied and still yield reliable results. The resulting compressed KPCA models are in close agreement with the uncompressed case. The memory requirements for storing a multi-gigabyte Gram matrix are reduced by several orders of magnitude. Also, considerable increases in acceleration factors are observed for both data sets.

Chapter 5

Summary and Conclusions

Given its ability to acquire high spatial and spectral resolution data, spectroscopic imaging provides a detailed understanding of heterogeneous samples. These data are commonly arranged in the form of a three-dimensional (3D) data cube (Figure 1) which is then analyzed by means of chemometrics to extract the desired chemical information (Figure 2). Chemometrics can determine which analytes are present within a sample, their spatial distribution, and how much of the analytes are present. Because of the growing number of pixels in modern day imaging detectors, data sets acquired from high-resolution spectrometers can easily reach volumes of several gigabytes. Performing chemometric calculations on such large amounts of data creates unrealistic demands on computational resources; specifically, computer memory and processing speed. As measurement techniques advance and the size of multi-dimensional detectors continues to increase, the amount of acquired experimental data will often exceed computational resources. Also, new data analysis (chemometric) methods are persistently being developed. As with KPCA (Chapters 2.4 and 4), new chemometric methods may require individualized compression routines. Thus, there will always be a need for innovative compression-based chemometric algorithms to accelerate calculation times and reduce data storage space.

This thesis describes, in detail, two novel compression-based chemometric techniques. Both methods utilize multi-dimensional wavelet transforms to accomplish accurate data compression, thus increasing computation speed. The first method automatically selects the optimum wavelet combination for any multi-dimensional data set. Principal component analysis (PCA) is applied during this study to asses the capabilities of the algorithm. The second compression technique is developed specifically for kernel principal component analysis (KPCA), the nonlinear extension of PCA. The unique steps for executing KPCA require a compression method that is applied during the KPCA calculations (or ‘on the fly’). Both compression procedures are thoroughly tested using multiple 3D data sets obtained from different experimental setups. Several degrees of compression are also tested to determine how much compression can be applied and still yield reliable chemometric results. Significant decreases in both computation times and memory requirements are observed for both compression algorithms while simultaneously maintaining accurate data representation.

List of References

List of References

[1] Luttrell R.D., Gilbert M.K., Vogt F. Composing hybrid wavelets for optimum and near-optimum representation and accelerated evaluation of *N*-way data sets. *J. Chemom.* 2007; **21**: 65

[2] Luttrell R.D., Vogt F. Accelerating Kernel Principal Component Analysis (KPCA) by Utilizing 2-Dimensional Wavelet Compression: Applications to Spectroscopic Imaging. *J. Chemom.* 2008; in press

[3] Gilbert M., Luttrell R., Stout D., Vogt F. Introducing Chemometrics to the Analytical Curriculum – Combine Theory and Lab Experience, Journal of Chemical Education, 2008, **85**: 135 (+ 45 pages of supplemental material)

[4] Strang G. *Linear Algebra and its applications*, 3rd ed.; Harcourt Brace Jovanovich College Publishing: Fort Worth, 1988

[5] Kolman B., Hill D.R. *Elementary Linear Algebra*, 7th ed.; Prentice-Hall, Inc. Upper Saddle River, New Jersey, 2000

[6] Lewis E.N., Treado P.J., Reeder R.C., Story G.M., Dowery A.E., Marcott C., Levin I.W. Fourier transform spectroscopic imaging using an infrared focal-plane array detector. *Analytical Chemistry.* 1995; **67**: 3377

[7] de Juan A., Tauler R., Dyson R., Marcolli C., Rault M., Maeder M. Spectroscopic imaging and chemometrics: a powerful combination for global and local sample analysis. *Trends in Analytical Chemistry.* 2004; **23**: 70

[8] Gilbert M.K., Vogt F. Augmenting spectroscopic imaging for analyses of samples with complex surface topographies. *Analytical Chemistry.* 2007; **79**: 5424-5428

[9] Martens H., Næs T. *Multivariate Calibration*, 2nd ed., John Wiley & Sons: New York, 1991

[10] Massart D. L., Vandegonste B. G. M., Buydens L. M. C., DeJong S., Lewi P. J., Smeyers J. *Handbook of Chemometrics and Qualimetrics*, Elsevier: Amsterdam, 1997

[11] Jolliffe I.T. *Principal Component Analysis*, 2nd ed.; Springer: New York, 2002

[12] Malinowski E. *Factor Analysis in Chemistry*, 3rd ed.; John Wiley & Sons: New York, 2002

[13] Vogt F., Booksh K. Chemometric Methods for Data Analysis. In *Kirk-Othmer Encyclopedia of Chemical Technology;* 4th online edition, Seidel, A., Ed.; John Wiley & Sons: New York, 2004

[14] Bro R. Multivariate calibration – What is in chemometrics for the analytical chemists? *Anal. Chim. Acta*, 2003, **500**: 185

[15] Lavine B., Workman J. Chemometrics *Anal. Chem.*, 2008, early view

[16] Geladi P., Grahn H. *Multivariate image analysis*, John Wiley & Sons, Chichester, 1996

[17] Lockheed Martin, Santa Barbara Focalplane, Goleta, CA 93117, also see: http://www.sbfp.com/focalplanearrays.html

[18] Mallat S. A theory of multiresolution signal decomposition: the wavelet representation. *IEEE Trans. PAMI* 1989; **11**: 674

[19] Daubechies I. *Ten Lectures on Wavelets*, CBMS-NSF regional conference series in applied mathematics (61): Philadelphia, PA, 1992

[20] Walczak B., Massart D. Wavelets - something for analytical chemistry? *trends anal. chem.*, 1997; **16**: 451

[21] Strang G., Nguyen T. *Wavelets and Filter Banks*, Wellesley-Cambridge Press: Wellesley, MA, 1997

[22] Depczynski U., Jetter K., Molt K., Niemoller A. The fast wavelet transform on compact intervals as a tool in chemometrics. I. Mathematical background. *Chemom. Intell. Lab. Syst.*, 1997; **39**: 19

[23] Depczynski U., Jetter K., Molt K., Niemoller A. The fast wavelet transform on compact intervals as a tool in chemometrics II. Boundary effects, denoising and compression. *Chemom. Intell. Lab. Syst.*, 1999; **49**: 151

[24] Mallat S. *A wavelet tour of signal processing*, 2nd ed., Academic Press: San Diego, CA, 1999

[25] Jetter K., Depczynski U., Molt K, Niemoller A. Principles and applications of wavelet transformation to chemometrics. *Anal. Chim. Acta*, 2000; **420**: 169

[26] Chau F., Liang Y., Gao J., Shao X. *Chemometrics – From Basics to Wavelet Transform*, John Wiley & Sons: Hoboken, NJ, 2004

[27] Alsberg B., Woodward A., Winson M., Rowland J., Kell D. Wavelet denoising of infrared spectra. A*nalyst* 1997; **122**: 645

[28] Barclay V., Bonner R. Application of Wavelet Transforms to Experimental Spectra: Smoothing, Denoising, and Data Set Compression. *Anal. Chem*. 1997; **69**: 78

[29] Ehrentreich F. Wavelet transform applications in analytical chemistry. *Anal. Bioanal. Chem.* 2002; **372**: 115

[30] Fang H., Huang D. Noise reduction in LIDAR signal based on discrete wavelet transform. *Optics Comm.* 2004; **233**: 67

[31] PKWARE, Inc., Data Security Software, Milwaukee, WI 53203, also see: http://www.pkware.com/

[32] Vogt F., Tacke M. Fast principal component analysis (PCA) of large data sets. *Chem. Intell. Lab. Syst*. 2001; **59**: 1

[33] Vogt F., Mizaikoff B., Tacke M. Numerical methods for accelerating the PCA of large data sets applied to hyperspectral imaging. *SPIE* 2001; **4576**: 215

[34] Vogt F., Tacke M. Fast Principal Component Analysis of Large Data Sets Based on Information Extraction. *J. Chemom.* 2002; **16**: 562

[35] Vogt F., Booksh K. Realization of Discrete (Inverse) Wavelet Transforms in Arbitrary Dimensions. *J. Chemom.* 2005; **19**: 575

[36] Vogt F., Banerji S., Booksh K. Utilizing 3-dimensional wavelet transforms for accelerated evaluation of hyperspectral image cubes. *J. Chemom.* 2004; **19**: 350

[37] Vogt F., Cramer J., Booksh K. Introducing Multi-dimensional 'Hybrid Wavelets' for Enhanced Evaluation of Hyperspectral Image Cubes and Multi-way Data Sets. *J. Chemom.* 2005; **19**: 510

[38] Golub G., Van Loan C. *Matrix Computations*, 3rd ed.; Johns Hopkins University Press: Baltimore, 1996

[39] Schölkopf B., Smola A., Müller K.-R. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 1998; **10**: 1299

[40] Schölkopf B. and Smola A.J. *Learning with Kernels*. The MIT Press: Cambridge, Massachusetts and London, England, 2002

[41] Lee J.-M., Yoo C.K., Choi S.W., Vanrolleghem P.A., Lee I.-B. Nonlinear process monitoring using kernel principal component analysis. *Chemical Engineering Science*, 2004; **59**: 223

[42] Choi S.W., Lee I.-B. Nonlinear dynamic process monitoring based on dynamic kernel PCA. *Chemical Engineering Science*, 2004; **59**: 5897

[43] Choi S.W., Lee C., Lee J.-M., Park J.H., Lee I.-B. Fault detection and identification of nonlinear processes based on kernel PCA. *Chemom. Intell. Lab. Syst.*, 2005; **75**: 55

[44] Jade A.M., Srikanth B., Jayaraman V.K., Kulkarni B.D., Jog J.P., Priya L. Feature extraction and denoising using kernel PCA. *Chemical Engineering Science*, 2003; **58**: 4441

[45] Müller K.-R., Mika S., Rätsch G., Tsuda K., Schölkopf B. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 2001; **12**: 181

[46] Liu Z., Chen D., Bensmail H. Gene expression data classification with kernel principal component analysis. *J. of Biomedicine and Biotechnology*, 2005; **2**: 155

[47] Sen A., Srivastava M. *Regression Analysis – Theory, Methods, Applications*, Springer: New York, 1990

[48] Draper N., Smith H. *Applied Regression Analysis*, 3rd ed.; J. Wiley: New York, 1998

[49] Weisberg, S. *Applied linear regression*, 3rd ed.; John Wiley & Sons: Hoboken, 2005

[50] Skoog D. A., Holler F. J., Nieman T. A. *Principles of Instrumental Analysis,* 5th ed.; Saunders College Publishing: Philadelphia, 1998

[51] Vogt F., Steiner H., Booksh K., Mizaikoff B. Chemometric correction of drift effects in optical spectra. *Appl. Spectrosc.* 2004, **58**: 683

[52] Press W., Flannery B., Teukolsky S., Vettering W. *Numerical Recipes in C++: The Art of Scientific Computing*, 2nd ed.; Cambridge University Press: Cambridge, 2002. also see: http://www.library.cornell.edu/nr/cbookcpdf.html

[53] Mandel, J. Use of the Singular Value Decomposition in Regression Analysis. *Amer. Statistician.* 1982, **36**: 15

[54] Vogt F., Mizaikoff B. Dynamic determination of the dimension of PCA calibration models using F-statistics. *J. Chemom.* 2003, **17**: 346

[55] Cramer J., Kim Y.-C., Vogt F., Booksh K. Accelerating the Analyses of 3-way and 4-way PARAFAC Models Utilizing Multi-dimensional Wavelet Compression. *J. Chemom.* 2005; **19**: 593

[56] Hinnrichs M. Infrared hyperspectral tunable filter imaging spectrometer for remote leak detection, chemical speciation, and stack/vent analysis applications. *Proc. SPIE* 2002; **4574**: 144152

[57] Hinnrichs M., Jensen J., McAnally, G. Handheld hyperspectral imager for standoff detection of chemical and biological aerosols. *Proc. SPIE* 2004; **5268**: 67

[58] Hinnrichs M., Piatek B. Hand-held hyperspectral imager for chemical/biological and environmental applications. *Proc. SPIE* 2004; **5270**: 10

[59] Vogt F. Trends in Remote Spectroscopic Sensing – Experimental Techniques and Chemometric Concepts. *Current Anal. Chem.* 2006; **2**: 107

[60] Vogt F., Dable B., Cramer J., Booksh K. Recent advancements in chemometrics for smart sensors. *Analyst* 2004; **129**: 492

[61] Savitzky A., Golay, M. Smoothing and differentiation of data by simplified least squares procedures. *Anal. Chem.* 1964, **36**: 1627

[62] Adebajo M.O., Frost R.L., Kloprogge J.T., Kokot S. Raman spectroscopic investigation of acetylation of raw cotton. *Spectrochim. Acta Mol. Spectros.* 2006, **64**: 448

[63] Edwards H.G.M., Farwell D.W. FT-Raman spectrum of cotton: a polymeric biomolecular analysis. *Spectrochim. Acta*, 1994, **50A**: 807

Appendices

Appendix 1

Enhancing the Prediction of Cotton Micronaire Values from Near-infrared (NIR) Absorbance Spectra

A.1.1 Experimental Overview

Micronaire is a physical property of cotton that represents the maturity of the cellulose fibers within a cotton sample. Simply stated, the more cellulose layers within a sample, the greater its maturity (i.e. the higher the micronaire value). The United States Department of Agriculture (USDA), Cotton Structure and Research Unit, is currently investigating the use of near-infrared (NIR) spectroscopy to predict cotton micronaire values. Through collaboration with the USDA we are developing new data analysis methods to enhance their current cotton micronaire prediction techniques. For this purpose, Principal Component Regression (PCR) [3], [9], [11] - [14] (Chapter 2.3) is applied. Since all of the cotton NIR absorbance spectra contain strong, non-reproducible fluctuations, even within the same sample, data processing techniques are developed and applied to correct for these random baseline drifts. The goal is to enhance the precision of predicted micronaire values. Deviations within ± 0.3 micronaire units for 70% of the evaluated data are considered to be ideal.

In this report, representative results are discussed that reflect what has been found to be a general trend. First, single absorption spectra are analyzed individually followed by comparative studies involving spectra averaging. This report concludes with a performance assessment of the utilized chemometric algorithm, PCR, and its various data pre-processing steps. Notation is as follows: we are given NIR spectra for 191 different cotton samples. Each individual sample consists of 5 repeatedly measured spectra giving a total of 955 spectra. 9 different cotton files make up the total 191 samples. These cotton files are labeled as *cot*1, *cot*2, *cot*3, *cot*4, *cot*5, *cot*6, *cot*7, *cot*8, and *m*38. Thus, each cotton file contained a varying number of samples.

A.1.2 Results without Baseline Correction

Strong background drifts are present in all of the cotton absorbance spectra. These drifts are random from spectrum to spectrum; therefore, mean-centering the data prior to chemometric evaluation would not enhance the results. Mean-centering is successful only when a constant background is present in all spectra being analyzed. Thus, mean-centering has not been applied to any of the data.

A conventional PCR was applied first. All 175 single absorbance spectra corresponding to cotton file *cot6* were used for calibration. The predicted results of selected remaining cotton files (*cot4* and *cot8*) not used for calibration are shown in Figure 23.

Figure 23 shows that there are significant deviations in the predicted micronaire values for both *cot4* and *cot8*. Only 42% of the predicted micronaire values for *cot4* fell into the acceptance range of “true micronaire value” ± 0.3. A better percentage of 69% was achieved for *cot8*. This enhanced percentage is evident by the narrower scatter of the predicted micronaire values for *cot8* (Figure 23 (b)). Nevertheless, the prediction results for both *cot4* and *cot8* are still below the acceptable limit. Similar trends in predicted micronaire values were obtained for all of the remaining cotton spectra not used for calibration. Overall, the prediction percentages for each cotton sample were very random with most of the prediction percentages falling below 70%. Hence, improvements to the prediction algorithm are essential.

The research objective is to apply different data pre-treatment steps in order to reduce the impact of the random baseline fluctuations. Polynomial fitting and Savitzky-Golay differentiation [52], [61] are methods that have been implemented and applied on single and averaged spectra. The prediction results obtained with these methods will be presented in the remainder. Background correction methods such as poly-PCR and pseudo-PCR [51] have successfully been applied to account for spectral drifts in situations when the calibration spectra have a stable baseline and drifts only occur later on while measuring unknown samples. This is not the case here and it was found that

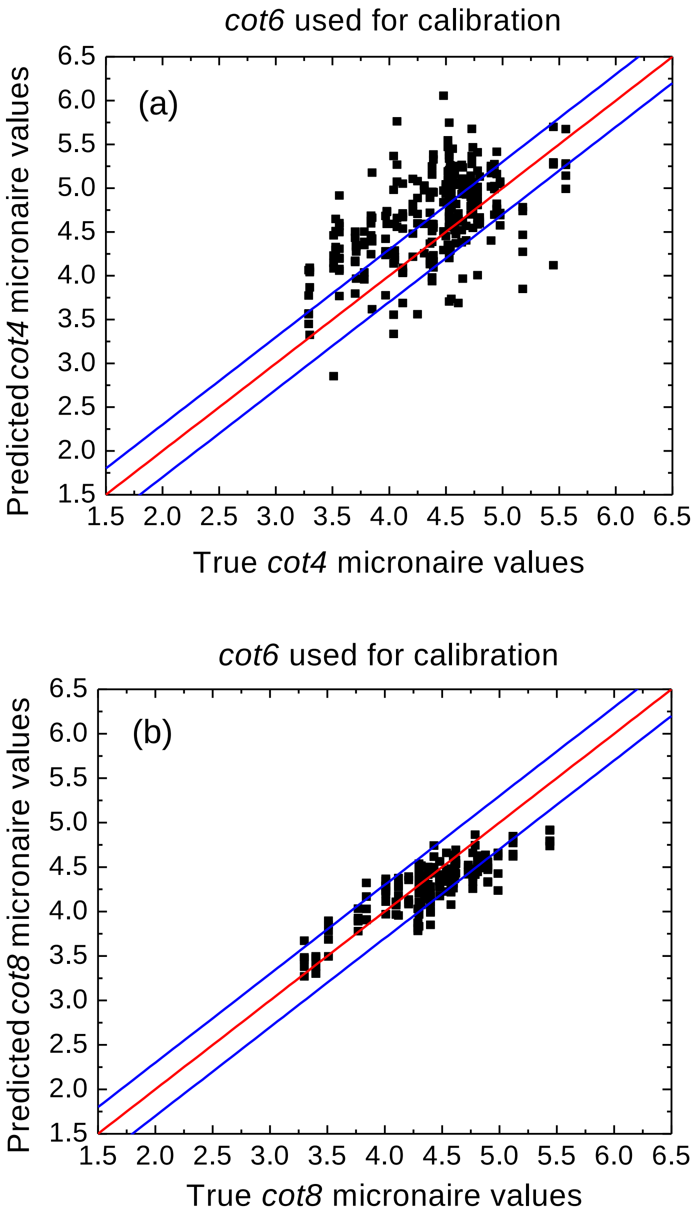


Figure 23: Predicted versus true micronaire values obtained from non-calibrated cotton spectra. (a) cot4 predicted values and (b) cot8 predicted values using cot6 for calibration. The red line at 45º indicates predicted = true micronaire value. The blue lines represent the ‘true micronaire value ± 0.3’ acceptance range.

applying poly- and pseudo-PCR to the cotton data did not enhance the prediction results. The reason for this is that drifting calibration spectra introduce such drifts into the PCR calibration model which in return features an increased number of PCs. These additional PCs enable the modeling of drifts in unknown spectra.

A.1.3 Results with Baseline Correction – Polynomial Fitting

The first method estimates the (drifted) baseline by fitting a fourth order polynomial to the individual absorbance spectrum. The resulting fit polynomial is then subtracted from the spectrum. Figure 24 shows the results of this procedure when applied to a *cot6* spectrum. After removal of this polynomial feature the resulting spectra show a flat baseline that is close to zero absorption; thus, the drift has been removed considerably (Figure 24). Since five spectra obtained from a *cot6* sample correspond to the same micronaire value it is anticipated that, after background correction, they will resemble each other closely. As shown in Figure 24, this has been achieved leaving only minor differences. Thus, with these random baseline fluctuations removed, an enhanced prediction of micronaire values is expected.

A data set consisting of all *cot6* spectra was used for calibration without averaging the five repetitions for each sample; the *cot4* spectra were used as the unknown test spectra for prediction. To all the spectra, fourth order polynomials were fitted and subtracted. This resulted in 46% of the predicted *cot4* micronaire values to fall inside the accepted range. This is only a slight improvement compared to the 42% predicted correctly without baseline correction (see A.1.2). A similar trend was found for the other data sets.

A.1.4 Results with Baseline Correction – Second Derivatives

As an alternative to baseline polynomial based drift correction the use of derivatives has been investigated. The first derivative removes constant offsets from the spectra, the second derivative cancels linear slopes and so on. For this purpose, the Savitzky-Golay method [52], [61] has been utilized. This technique was applied to *cot6* acting as calibration data and *cot4* used for prediction. Figure 25 displays three *cot6*

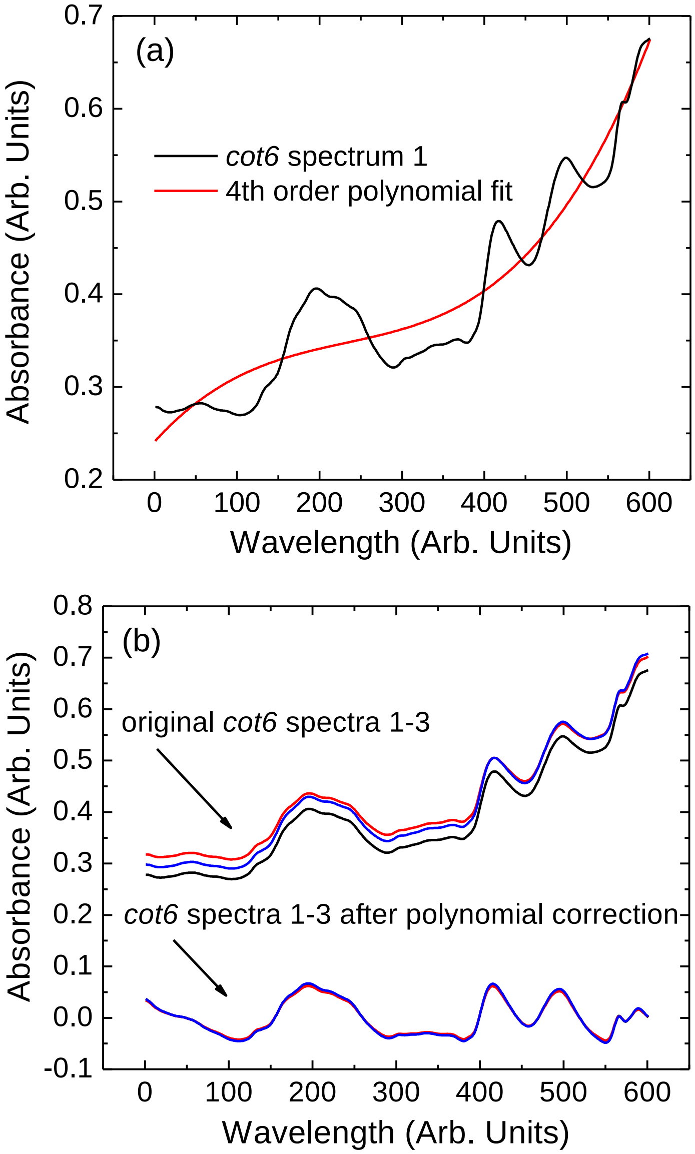


Figure 24: (a) An example of a fourth order polynomial fitted to the first spectrum of *cot6*. (b) First three original absorbance spectra of *cot6* before (top) and after (bottom) removal of a fourth order fit polynomial. After this baseline correction the spectra acquired from the same sample look much more similar. Since these spectra were all assigned the same micronaire number, this correction is expected to result in more precise micronaire prediction than without baseline correction.

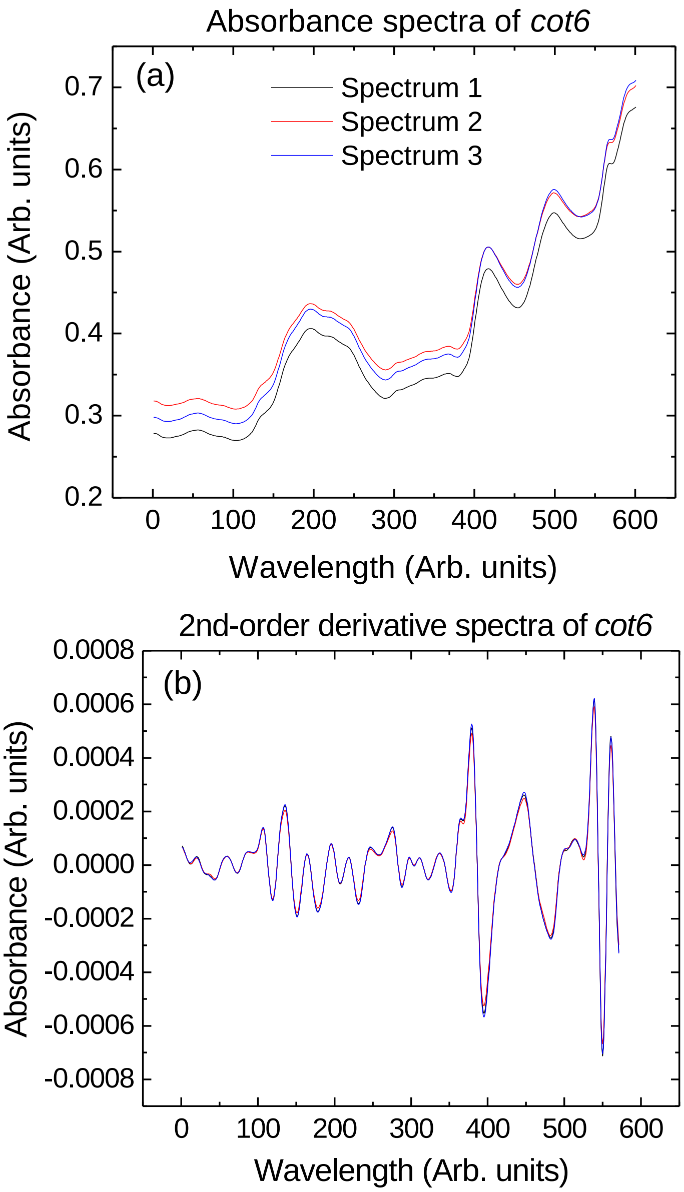


Figure 25: (a) The first three absorbance spectra from the *cot6* data set. (b) Corresponding 2nd-order derivative spectra. The three derivative spectra are similar since they all correspond to the same micronaire value.

absorbance spectra and the corresponding 2nd-order derivatives. Again, since these three *cot6* spectra correspond to the same micronaire value, it is expected that the spectra will be more similar to each other after background correction. As shown in Figure 25 (b) this is the case. After PCR evaluation of the derivative spectra 51% of the *cot4* micronaire values were predicted correctly within the acceptable threshold range. This is only about a 5% increase in prediction compared to the polynomial fitted data (46%, see A.1.3). These results indicate that there are more challenges to overcome than baseline drifts. This conclusion has been confirmed by the fact that after correcting for baseline drifts in the calibration *and* the unknown spectra, considerable improvements in the prediction results were not obtained.

A.1.5 Results Utilizing Spectra Averaging

Thus far, individual spectra have been analyzed. From each cotton sample, five spectra have been acquired and the average spectrum of these five spectra will be used in the remainder. The goal was to investigate whether averaging derives a more representative spectrum for a given cotton sample and its corresponding micronaire value.

Initial studies involving averaged spectra included using the averaged absorbance spectra of the ‘Texas Tech, TT’ (*cot5* - *cot8*) cotton samples as the calibration data and then predicting the micronaire values of selected cotton spectra. A standard PCR as well as both the polynomial fitting and Savitzky-Golay derivative methods were applied using the averaged ‘TT’ absorbance data for calibration. The results obtained using averaged *cot4* and *cot8* absorbance spectra for prediction, utilizing all of the different data pre-treatment methods, are given in Table 1. *cot4* spectra were chosen for evaluation because it was *not* included in the calibration set, thus giving insight of the strength of the prediction capabilities of this approach. The results for *cot4* prove that this method is not adequate for predicting micronaire values, to within the stated threshold range, for non-calibrated spectra. Although there is significant improvement in prediction after correcting for background drifts (Table 1, center and right columns), all of the results are considered to be unacceptable (<70%). *cot8 was included* in the calibration set and all of the prediction results for this data set fell within the

Table 1: Prediction results obtained via PCR using averaged Texas Tech ('TT') spectra for calibration and averaged *cot4* and *cot8* spectra for evaluation, respectively. (left column) results using averaged absorbance, (center column) polynomial fitted, and (right column) 2nd-order derivative spectra are given.

	Texas Tech ('TT') data (*cot5* – *cot8*) used for calibration		
	Calibration & Evaluation: ave. absorbance spectra	Calibration & Evaluation: ave. polynomial fitted spectra	Calibration & Evaluation: ave. 2nd-order derivative spectra
Cotton file used for evaluation	% Prediction	% Prediction	% Prediction
cot4 (not included in calibration)	9	55	51
cot8 (included in calibration)	74	97	97

acceptable prediction range (>70%) as shown in the bottom row of Table 1. There is also an improvement in prediction for the *cot8* data after correcting for the background drifts (Table 1, center and right columns).

It was observed (Table 1) that using a selected averaged set of spectra for calibration (i.e. 'TT' spectra) resulted in decent prediction results only for the spectra included in the calibration (i.e. *cot8*) and poor prediction results for spectra not included in the calibration (i.e. *cot4*). Therefore, a more drastic approach was taken next that used *all* average spectra for both calibration *and* prediction. These results are discussed next.

Using *all* of the averaged absorbance spectra for both calibration and evaluation and applying a standard PCR, only two cotton data sets (*cot2* and *cot8*) resulted in >70% prediction of micronaire values (left column of Table 2). Next, the averaged polynomial fitted absorbance spectra were used both for calibration and prediction and the results are listed in the center column of Table 2. There is a slight improvement in prediction compared to the left column, but only three sets of samples (*cot2*, *cot7* and *cot8*) gave >70% of acceptable prediction. Finally, the averaged 2nd-order derivative spectra were used for both calibration and prediction and these results are shown in the right column of Table 2. Although these results are better than the averaged polynomial fitted (Table 2, center column) and absorbance data (Table 2, left column), only 4 of the 9 cotton sample sets were predicted at 70% or greater. In summary, Table 2 shows that an overall increase in prediction can be achieved when using the averaged polynomial fitted data. The prediction results further improve when the averaged 2nd-order derivative spectra are used for PCR evaluation. An explanation for this trend could be the increase in total PCs that are gained from using averaged 2nd-order derivative and polynomial fitted spectra as opposed to using averaged absorbance spectra. After averaging these various data 5 principal components (PCs) were obtained for the absorbance data, 9 PCs for the polynomial fitted data, and 19 for the derivative data. In the case of non-averaged absorbance spectra, 5 PCs were obtained for conventional PCR, 11 PCs for the single polynomial fitted data and 42 PCs for the derivative data.

Using all of the averaged data for both calibration and evaluation should have resulted in much higher prediction percentages. This was clearly not the case (Table 2). Originally, it was assumed that the background drifts in the spectra were the main cause for the low prediction results. But, after implementing several background correction

Table 2: Prediction results obtained via PCR using averaged cotton absorbance (no baseline correction) (left column), polynomial fitted (center column), and 2nd-order derivative (right column) spectra. All of the averaged spectra were used for both calibration and prediction.

	Calibration & Evaluation: All ave. absorbance spectra	Calibration & Evaluation: All ave. polynomial fitted spectra	Calibration & Evaluation: All ave. 2nd-order derivative spectra
Cotton file	% Prediction	% Prediction	% Prediction
cot2	**86**	**71**	**86**
m38ng	44	50	61
cot3	33	33	33
cot4	64	69	**76**
cot5	50	50	50
cot6	61	67	69
cot7	56	**83**	**89**
cot8	**76**	**82**	**95**

methods significant improvements in prediction were not observed. This is a strong indication that something additional to the baseline problem is causing large deviations from the true micronaire values. A detailed investigation is required to answer why greater prediction results were not achieved (see A.1.6).

A.1.6 "A Closer Look"

The question here is whether or not the chemometric method of choice, PCR, is providing reliable results for these data and whether the requirements for a successful PCR are met. Therefore, it is necessary to analyze the quality of the PCs and scores that are calculated via PCR.

In order to asses the quality of the PCs, a set of PCs obtained for cotton data set will be used to reconstruct the spectra of another cotton data set. Because all of the cotton spectra contain the same major spectral features, which are modeled by the PCs, the PCs for one cotton file should be able to successfully model the spectral features of another cotton file. If this was not the case, the calibration model would not be able to predict future unknown samples. As an example, the PCs for *cot6* were used to reconstruct the spectra for *cot4* (see Figure 26). This is done by first calculating the PCs for *cot6* by PCR. Since any spectrum can be represented as a linear combination of PCs and scores (PCR), the scores for *cot4* are determined by projecting *cot4* spectra onto the PCs for *cot6*. Once the scores for *cot4* are known, they are then multiplied by the PCs of *cot6* giving the reconstructed spectra of *cot4*. As an example, one reconstructed spectrum of *cot4* is plotted in Figure 26 along with the corresponding original spectrum. Also shown is the residual spectrum (i.e. the difference between the original and reconstructed spectra). If the original and reconstructed spectra are very similar then the resulting residual spectra will approximately equal zero (see Figure 26 (bottom)). The good agreement between the original and reconstructed spectra indicates that the PCs acquired from one cotton data set is capable of describing the spectral features of another data set very well. Since the PCs do represent the cotton spectral features, the scores must now be assessed to see if they correctly represent changes in micronaire values.

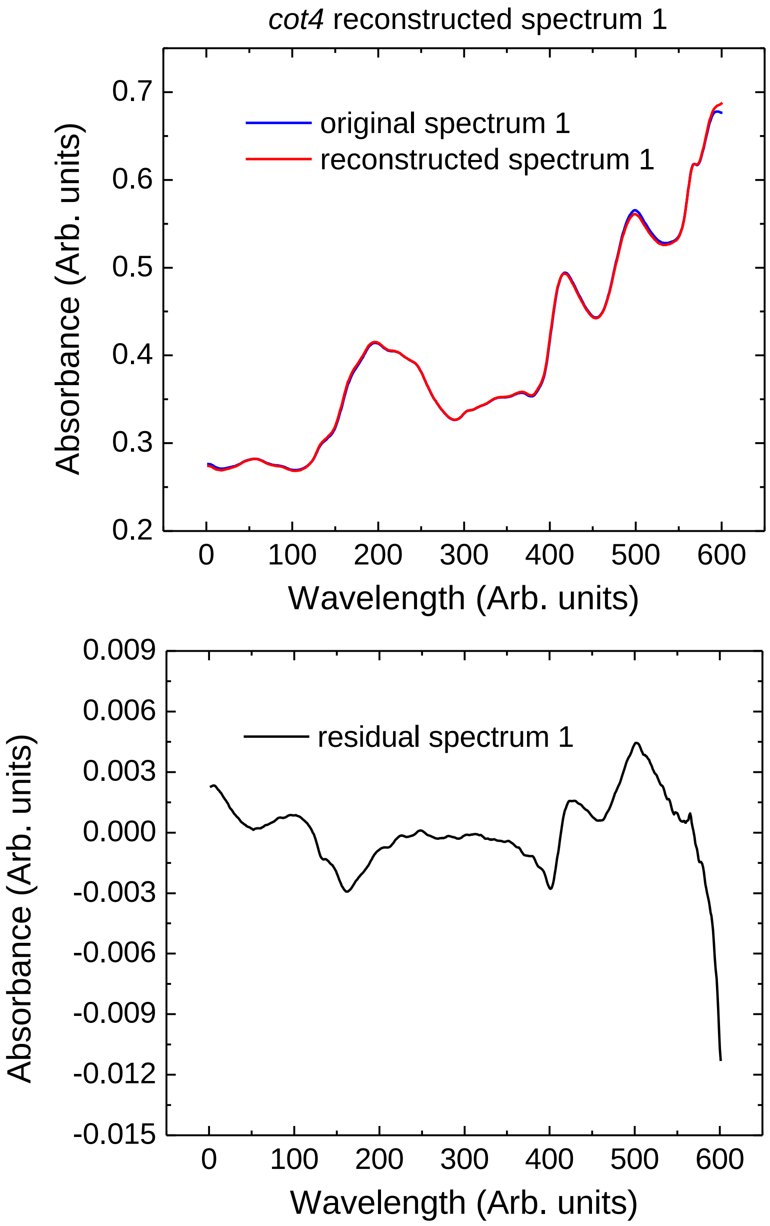


Figure 26: (top) Original vs. reconstructed spectrum 1 of *cot4*. The reconstructed spectrum was calculated using the PCs obtained for *cot6*. (bottom) The resulting residual spectrum. Since the residual values are two orders of magnitude smaller than the measurement values, it can be concluded that the PCs of one sample can model the spectral features of another sample.

Changes in micronaire should induce changes in the cotton spectra. As a result, changes in the scores should occur as well since the scores are directly related to the micronaire values via PCR (Chapter 2.3.2). Within one sample, however, the score values should not change. In other words, all five spectra obtained from the same sample (cotton boll) have the same micronaire number. Predicting these micronaire numbers from the score values requires that the scores are the same for all five spectra. If this is not the case, prediction will fail, thus resulting in a wide spread of predicted micronaire values for a single cotton sample. This is illustrated in Figure 27 (a). Further, since PCR is a linear algorithm, a linear trend between score values and micronaire should exist. If, for instance, upon PCR evaluation a set of spectra all corresponding to identical micronaire values produces a wide range of score values then there exists a miscorrelation between micronaire and scores. If a direct correlation between micronaire and scores is not given, it will hinder the PCR approach considerably.

To investigate the quality of the scores, various sets of score values are calculated for selected sets of cotton data and plotted against micronaire value (see Figure 27). Using *cot6* for calibration, the scores for *cot4* were calculated under four different conditions. (i) Single absorbance spectra were used for both calibration and prediction and the resulting score values are shown in Figure 27 (a). It is obvious that the scores are random and contain no apparent trend with micronaire. (ii) Using averaged, instead of single absorbance spectra, improves the results only slightly as shown in Figure 27 (b). (iii) Single 2nd-order derivative spectra improve the prediction results (Figure 27 (c)) by creating a more obvious linear trend between scores and micronaire; however, there is still a considerable spread of scores for any given micronaire value. (iv) The spread of score values is reduced when averaged 2nd-order derivative spectra (Figure 27 (d)) are used for the analysis, thus enhancing the prediction results even further (see Table 2, right column). As shown in Figure 27 (d), several different micronaire values correspond to the same score value. In spite of correcting for the background drifts present in all of the cotton spectra the scores still contain too much variation to accurately predict micronaire values for unknown spectra.

In conclusion, there are concerns about the correlation between the given micronaire values and their associated NIR spectra. The spectra may not be very representative of the micronaire values, thus causing imprecise micronaire predictions. Therefore, a closer look into the labeling of micronaire values to NIR spectra is

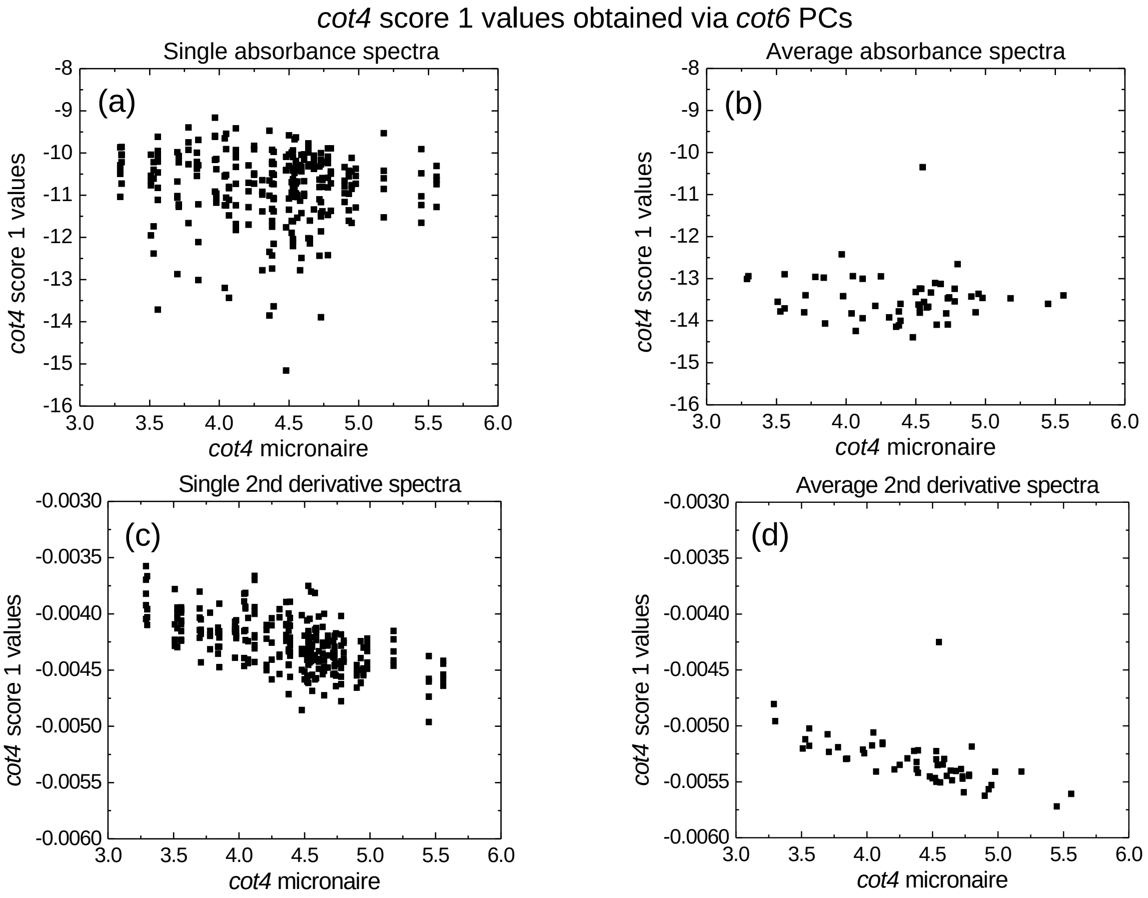


Figure 27: *cot4* score 1 values vs. micronaire. Results obtained via PCR using *cot6* PCs and (a) single absorbance, (b) averaged absorbance, (c) single 2nd-order derivative, and (d) averaged 2nd-order derivative cotton spectra.

suggested as this is a crucial requirement for the success for many chemometric methods.

A.1.7 Future Outlook

Given the concern about the quality of the scores obtained from the cotton data, we are interested in continuing our efforts in developing new chemometric methods to correct for score variations. One option is to replace linear PCR with its non-linear extension, Kernel-PCA/PCR (see Chapter 2.4). Deriving a non-linear relation between scores and micronaire values might be a feasible way to handle the rather broad spread of score values within the same sample.

If it is common that micronaire values can vary throughout a cotton sample (cotton boll), an imaging study may provide a better understanding of the distribution of micronaire within cotton. Also, cotton contains strong Raman features in the 3600 - 100 cm^{-1} spectral region (See Figure 28 and references [62], [63]). Hence, Raman imaging may lead to an enhanced correlation between micronaire and cotton sample. Figure 28 shows a Raman spectrum of a commercial cotton ball. The spectrum was acquired with a Bruker Optics Senterra Raman spectrometer equipped with an Olympus microscope containing 10, 20, 50, and 100x objectives, and a motorized three-dimensional translation stage which is programmable for sample mapping applications. The spectrum was recorded over a spectral region of 3200 – 70 cm^{-1} with a spectral resolution of 3 – 5 cm^{-1} using a 785 nm (100 mW) excitation laser and an integration time of 90 sec.

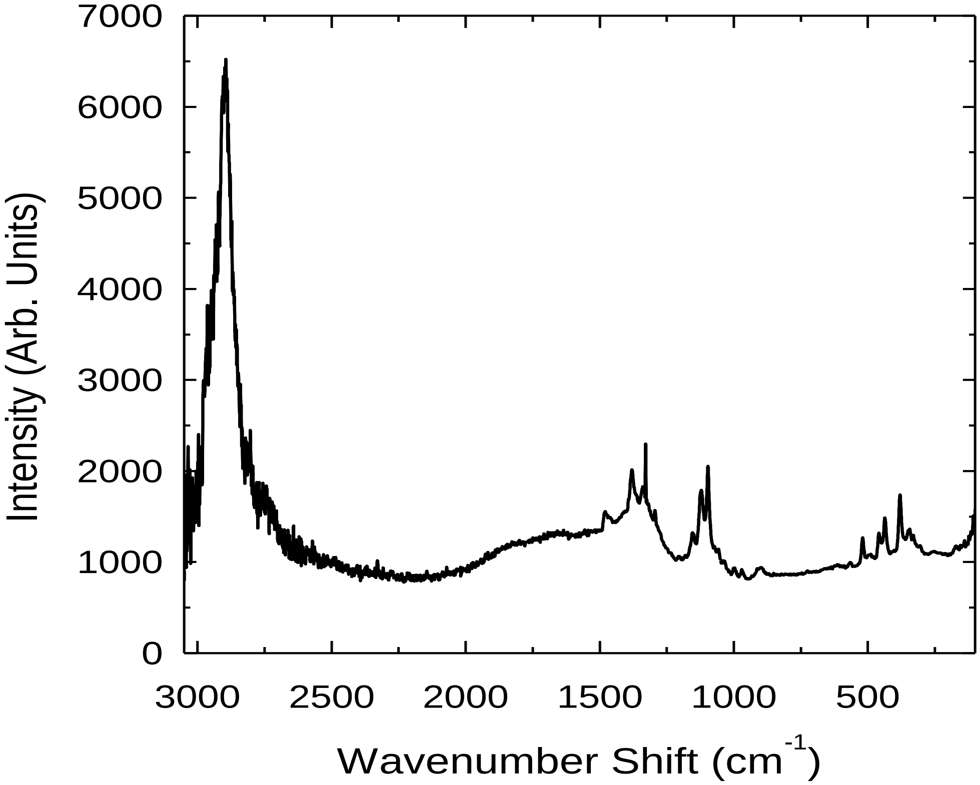


Figure 28: Raman spectrum of a commercial cotton ball. See text for experimental parameters.

Appendix 2

Summary of Graduate School Honors, Publications, and Presentations

Honors

1. The Burchfield Burridge Warner Graduate Fellowship in Chemistry, Department of Chemistry, University of Tennessee, Knoxville, TN, April **2007**

2. Graduate Research Merit Award, Department of Chemistry, University of Tennessee, Knoxville, TN, April **2005**

3. Chemical Physics Fellowship, Department of Chemistry, University of Tennessee, Knoxville, TN, January **2004**

Publications

1. R.D. Luttrell, F. Vogt
   Accelerating Kernel Principal Component Analysis (KPCA) by Utilizing 2-Dimensional Wavelet Compression: Applications to Spectroscopic Imaging
   Journal of Chemometrics, **2008**, in press

2. M.K. Gilbert, R.D. Luttrell, D. Stout, F. Vogt
   Introducing Chemometrics to the Analytical Curriculum – Combine Theory and Lab Experience
   Journal of Chemical Education, vol. 85, **2008**, 135-137

3. R.D. Luttrell, M.K. Gilbert, F. Vogt
   Composing Hybrid Wavelets for Optimum and Near-Optimum Representation and Accelerated Evaluation of N-way Data Set
   Journal of Chemometrics, vol. 21, **2007**, 65-75

4. A. Barbour, R.D. Luttrell, J. Choi, J.L. Musfeldt, D. Zipse, N.S. Dalal, D.W. Boukhvalov, V.V. Dobrovitski, M.I. Katsnelson, A.I. Lichtenstein, B.N. Harmon, P. Kögerler
   Understanding the Gap in Polyoxovanadate Molecule-Based Magnets
   Physical Review B, vol. 74, **2006**, 014411-1 – 014411-9

5. R.D. Luttrell, S. Brown, J. Cao, J.L. Musfeldt, R. Rosentsveig, R. Tenne
Dynamics of Bulk vs. Nanoscale WS_2: Local Strain and Charging Effects
Physical Review B, vol. 73, **2006**, 035410-1 – 035410-6

Presentations

1. R.D. Luttrell, F. Vogt
Enhancing the Prediction of Cotton Micronaire Values from NIR Absorbance Spectra
Progress Report at: United States Department of Agriculture (USDA), Southern Regional Research Center, New Orleans, LA, March **2008**

2. R.D. Luttrell, F. Vogt
Accelerating Kernel Principal Component Analysis (KPCA) by Utilizing 2-Dimensional Wavelet Compression: Applications to Spectroscopic Imaging
Poster at: Pittcon, New Orleans, LA, March **2008**

3. R.D. Luttrell, F. Vogt
Advancing Kernel Principal Component Analysis (KPCA) by Utilizing 2-Dimensional Wavelet Compression: A Means to Enhance Spectroscopic Imaging
Presentation at: FACSS, Memphis, TN, October **2007**

4. R.D. Luttrell, F. Vogt
Utilizing High-dimensional Wavelet Transforms for Accelerated Data Analyses in Chemical Imaging
Presentation at: Pittcon, Orlando, FL, March **2006**

5. R.D. Luttrell, S. Brown, J. Cao, J.L. Musfeldt, R. Rosentsveig, R. Tenne
Understanding the Dynamics of Bulk vs. Nanoscale WS_2: Local Strain and Charging Effects
Poster at: American Society of Mechanical Engineers Nanomechanics: Sensors & Actuators Conference, Knoxville, TN, May **2005**

6. R.D. Luttrell, S. Brown, J. Cao, J.L. Musfeldt, R. Rosentsveig, R. Tenne
Local Strain and Charge Localization Effects in WS_2 Nanoparticles
Presentation at: American Physical Society National Meeting, Los Angeles, CA, March **2005**

7. R.D. Luttrell, J. Cao, J.L. Musfeldt, R. Rosentsveig, R. Tenne
Understanding the Low-Energy Dynamics of WS_2 Inorganic Fullerene-Like (IF) Nanoparticles
Poster at: Gordon Research Conference: Solid State Chemistry, New London, NH, July **2004**

Vita

Robert Daryl Luttrell, Jr. was born in Norcross, GA on December 12, 1979. He was raised in Warner Robins, GA and went to grade school and middle school at Bonaire Elementary and Warner Robins Middle, respectively. He graduated from Warner Robins High School in 1998. From there, he attended Georgia College and State University in Milledgeville, GA and received a B.S. in chemistry in 2002. After working in the pharmaceutical industry as a research chemist for over a year, Robert began graduate school at the University of Tennessee, Knoxville in January, 2004. He will receive a doctorate in chemistry under the direction of research advisor Dr. Frank Vogt in summer 2008. Following graduation Robert will begin his career as professor of analytical chemistry at Salisbury University in Salisbury, MD.