Exploring the latent space between brain and behaviour using eigen-decomposition methods

João André de Matos Monteiro

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of

University College London.

Department of Computer Science

University College London

September 29, 2017

I, João André de Matos Monteiro, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

Machine learning methods have been successfully used to analyse neuroimaging data for a variety of applications, including the classification of subjects with different brain disorders. However, most studies still rely on the labelling of the subjects, constraining the study of several brain diseases within a paradigm of pre-defined clinical labels, which have shown to be unreliable in some cases. The lack of understanding regarding the association between brain and behaviour presents itself as an interesting challenge for more exploratory machine learning approaches, which could potentially help in the study of diseases whose clinical labels have shown limitations. The aim of this project is to explore the possibility of using eigen-decomposition approaches to find multivariate associative effects between brain structure and behaviour in an exploratory way.

This thesis addresses a number of issues associated with eigen-decomposition methods, in order to enable their application to investigate brain/behaviour relationships in a reliable way. The first contribution was showing the advantages of an alternative matrix deflation approach to be used with Sparse Partial Least Squares (SPLS). The modified SPLS method was later used to model the associations between clinical/demographic data and brain structure, without relying on *a priori* assumptions on the sparsity of each data source. A novel multiple hold-out SPLS framework was then proposed, which allowed for the detection of robust multivariate associative effects between brain structure and individual questionnaire items.

The linearity assumption of most machine learning methods used in neuroimaging might be a limitation, since these methods will not have enough flexibility to detect non-linear associations. In order to address this issue, a novel Sparse Canonical Correlation Analysis (SCCA) method was proposed, which allows one to use sparsity constraints in one data source (e.g. neuroimaging data), with non-linear transformations of the data in the other source (e.g. clinical data).

Acknowledgements

Reaching the end of a PhD project is a very long and arduous process, which is only possible with the help of numerous people. In my case, they were far too many to thank them all.

I would like to start by acknowledging my supervisors, Prof. John Shawe-Taylor and Prof. Janaina Mourão-Miranda, whose support and advice were absolutely essential. Without them, this thesis would have never been written.

I would also like to thank my collaborators and colleagues, which helped me throughout the various stages of my PhD: Prof. John Ashburner, Dr. Jessica Schrouff, Dr. Michele Donini, Dr. Dimitrios Athanasakis, Dr. Liana Portugal, Viivi Uurtio, Dr. Jaz Kandola, Dr. Delmiro Fernandez-Reyes, Prof. Juho Rousu, Dr. Gita Prabhu, Dr. Michael Moutoussis, Dr. Gabriel Ziegler, and Martin Axelsen. With a special thank you to Dr. Anil Rao and Dr. Maria João Rosa, who not only were very patient to sit down and answer my questions, but also gave me valuable feedback on the first complete draft of this thesis.

Although not included in this thesis, my work was also comprised of some contributions to the PRoNTo toolbox[1]. Therefore, I would like to thank the PRoNTo development team for the very insightful discussions: Prof. John Ashburner, Dr. Carlton Chu, Dr. Andre Marquand, Dr. Janaina Mourão-Miranda, Dr. Christophe Phillips, Dr. Jonas Richiardi, Dr. Maria João Rosa, and Dr. Jessica Schrouff.

Working with machine learning methods involves spending a lot of time dealing with all sorts of IT problems, which could not have been solved without the help of the UCL computer science department technical support group. I would like to thank Neil Daeche for handling most of my IT-related issues, and Tristan Clark for keeping the cluster running (probably the most important tool in my project).

[1] `http://www.mlnl.cs.ucl.ac.uk/pronto/`

I would like to acknowledge the people responsible for collecting and organising the data used in this thesis: the OASIS dataset, and the ADNI dataset. Furthermore, I would like to thank the NSPN consortium for providing their data, even though the results did not make the final version of this thesis.

This project was funded a PhD scholarship awarded by the *Fundação para a Ciência e a Tecnologia* (SFRH/BD/88345/2012). I would also like to thank the Wellcome Trust and the Guarantors of Brain, for the extra funding provided to attend several conferences during my project.

On non-work related acknowledgements, I would like to thank all my friends in London, which helped me throughout the years of my PhD. Living in London is definitely not a walk in the park, but it is definitely easier among friends, thus, I would like to specially thank my flatmates Débora Salvado and Florin Rothwell.

Last, but definitely not least, I would like to thank my family for all their continued love and support.

Contents

Impact Statement

With the increase in data collection that has been observed in recent years, several datasets are no longer comprised of a single data type coming from a single source, but of different types of data collected from multiple sources. In some situations, one may be interested in modeling the underlying relationships in a dataset containing information coming from two different sources (i.e. views), e.g. neuroimaging data and clinical/demographic data, in order to gain insights into the unobserved latent process which generated the data. This type of modeling has been undertaken in many applications, including: language [Hardoon and Shawe-Taylor, 2011], genetics [Witten et al., 2009, Parkhomenko et al., 2009], neuroimaging [Hardoon et al., 2007], and facial expression recognition [Zheng et al., 2006].

Although, this type of model has applications which span many different fields, the potential medical applications may have a paradigm-shifting impact, by helping redefine the way in which diagnosis is currently carried out, particularly in the psychiatric field. Current diagnostic labels in psychiatry are not very reliable, indeed, they have failed to predict treatment response, which suggests that the labels may not accurately reflect the underlying disease process [Insel et al., 2010]. In order to gain insights into these processes, one has to look at brain diseases from different angles simultaneously, which can be achieved by using eigen-decomposition methods. By making use of more exploratory modeling approaches, one may be able to combine large amounts of heterogeneous data to find patterns which allow for its stratification.

This thesis has provided several contributions to the neuroimaging field, by providing novel ways to model the relationships between brain and behaviour. The first contribution (Chapter 4) showed the advantages of using an alternative matrix deflation approach with sparse eigen-decomposition methods. This approach was then used with a sparse eigen-decomposition method to model the association

between clinical/demographic features and brain structure without relying on *a priori* assumptions regarding the sparsity of each view (Chapter 5). A multiple hold-out framework was then proposed (Chapter 6), which allowed for the detection of robust multivariate associative effects between brain structure and individual questionnaire items. Chapter 7 proposed an adaptation of the Alternating Least Squares (ALS) algorithm, which is a commonly used approach to solve eigen-decomposition problems, this adaptation allowed the ALS to converge more often, while providing comparable results. The ALS was finally adapted to solve a novel eigen-decomposition method (Chapter 8), which allowed one to enforce sparsity in views where the dimensionality is high, while simultaneously exploring non-linear relationships in views where the dimensionality is lower.

The benefits provided by a better understanding of brain disorders goes beyond the realm of academia, it is an essential step to refine current diagnostic tools. This is indeed a very important issue, but a very challenging problem as well, which is why it is unlikely that it will be solved by the work of a single research group. Nevertheless, this thesis has made some contributions which will hopefully enable other researchers to better understand the relationships between brain and behaviour, helping to pave the way for future work which may lead to the improvement of psychiatric diagnosis.

Publications

Software

- Pattern Recognition for Neuroimaging Toolbox (PRoNTo): `http://www.mlnl.cs.ucl.ac.uk/pronto/`

- Sparse Partial Least Squares (SPLS): `https://github.com/jmmonteiro/spls`

Papers

Submitted / under revision

- J. Schrouff, J. M. Monteiro, L. Portugal, M. J. Rosa, C. Phillips, and J. Mourão-Miranda. Embedding anatomical or functional knowledge in whole-brain multiple kernel learning models. *Neuroinformatics*, (accepted)[2]

- V. Uurtio, J. M. Monteiro, J. Kandola, J. Shawe-Taylor, D. Fernandez-Reyes, and J. Rousu. A tutorial on canonical correlation methods. *ACM Computing Surveys*, (accepted)

- M. J. Rosa, M. Moutoussis, G. Ziegler, J. M. Monteiro, L. Portugal, F. S. Ferreira, E. T. Bullmore, P. Fonagy, I. M. Goodyer, P. B. Jones, the NSPN Consortium, R. Dolan, and J. Mourao-Miranda. Brain-behavior modes of covariation in healthy and clinically depressed young people. *Scientific Reports*, (in preparation)

- M. Donini, J. M. Monteiro, M. Pontil, T. Hahn, A. J. Fallgatter, J. Shawe-Taylor, and J. Mourão-Miranda. Combining heterogeneous data sources for prediction: re-weighting and selecting what is important. *NeuroImage*, (submitted)

[2] Joint first author

2017

- A. Rao, J. M. Monteiro, and J. Mourão-Miranda. Predictive modelling using neuroimaging data in the presence of confounds. *NeuroImage*, 2017

2016

- J. M. Monteiro, A. Rao, J. Shawe-Taylor, and J. Mourão-Miranda. A multiple hold-out framework for sparse partial least squares. *Journal of Neuroscience Methods*, 271:182–194, 2016

- M. Donini, J. M. Monteiro, M. Pontil, J. Shawe-Taylor, and J. Mourao-Miranda. A multimodal multiple kernel learning approach to Alzheimer's disease detection. In *Machine Learning for Signal Processing (MLSP), 2016 IEEE 26th International Workshop on*, pages 1–6. IEEE, 2016

- A. Rao, J. Monteiro, and J. Mourao-Miranda. Prediction of clinical scores from neuroimaging data with censored likelihood Gaussian processes. In *Pattern Recognition in Neuroimaging (PRNI), 2016 International Workshop on*, pages 1–4. IEEE, 2016

2015

- J. M. Monteiro, A. Rao, J. Ashburner, J. Shawe-Taylor, and J. Mourão Miranda. Multivariate effect ranking via adaptive sparse PLS. In *Pattern Recognition in NeuroImaging (PRNI), 2015 International Workshop on*, pages 25–28. IEEE, 2015

- A. Rao, J. M. Monteiro, J. Ashburner, L. Portugal, O. Fernandes, L. De Oliveira, M. Pereira, and J. Mourão-Miranda. A comparison of strategies for incorporating nuisance variables into predictive neuroimaging models. In *Pattern Recognition in NeuroImaging (PRNI), 2015 International Workshop on*, pages 61–64. IEEE, 2015

2014

- J. M. Monteiro, A. Rao, J. Ashburner, J. Shawe-Taylor, and J. Mourão-Miranda. Leveraging clinical data to enhance localization of brain atrophy. In *International Workshop on Machine Learning and Interpretation in Neuroimaging*, pages 60–68. Springer, 2014

List of Figures

List of Tables

Nomenclature

Greek Symbols

δ	Step size of SCCA-ALS (Chapter 8)
γ	Regularisation hyper-parameter
λ	Eigenvalue
$\boldsymbol{\omega}$	Projection of $\boldsymbol{Y}$ onto $\boldsymbol{v}$, i.e. $\boldsymbol{\omega} = \boldsymbol{Y}\boldsymbol{v}$
$\boldsymbol{\xi}$	Projection of $\boldsymbol{X}$ onto $\boldsymbol{u}$, i.e. $\boldsymbol{\xi} = \boldsymbol{X}\boldsymbol{u}$

Matrices and vectors

$\boldsymbol{C}_{xx}$	Covariance matrix of $\boldsymbol{X}$
$\boldsymbol{C}_{xy}$	Covariance matrix of $\boldsymbol{X}$ and $\boldsymbol{Y}$
$\boldsymbol{C}_{yy}$	Covariance matrix of $\boldsymbol{Y}$
$\boldsymbol{K}_x$	Kernel of $\boldsymbol{X}$
$\boldsymbol{K}_y$	Kernel of $\boldsymbol{Y}$
$\boldsymbol{w}^{(i)}$	Vector $\boldsymbol{w}$ at the i^{th} iteration.
$\boldsymbol{X}(:,j)$	j^{th} column of $\boldsymbol{X}$, where $j \in \{1,\ldots,p\}$
$\boldsymbol{X}(i,:)$	i^{th} row of $\boldsymbol{X}$, where $i \in \{1,\ldots,n\}$
$\boldsymbol{X}^\intercal$	Transpose of matrix $\boldsymbol{X}$
$\boldsymbol{X}$	Data matrix with n rows and p columns
$\boldsymbol{Y}$	Data matrix with n rows and q columns

Other variables

c_u and c_v	SPLS hyper-parameters
d	Singular value
n	Number of samples
p and q	Number of features

Abbreviations

AD Alzheimer's Disease

ADAS Alzheimer's Disease Assessment Scale

ALS Alternating Least Squares

CCA Canonical Correlation Analysis

CDR Clinical Dementia Rating

CSF Cerebrospinal Fluid

CV Cross-Validation

DTI Diffusion Tensor Imaging

FAQ Functional Assessment Questionnaire

fMRI Functional Magnetic Resonance Imaging

FTD Frontotemporal Dementia

FWHM Full Width at Half Maximum

GM Grey Matter

KCCA Kernel Canonical Correlation Analysis

KRR Kernel Ridge Regression

LASSO Least Absolute Shrinkage and Selection Operator

MCI Mild Cognitive Impairment

MMSE Mini-Mental State Examination

MRI Magnetic Resonance Imaging

MSE Mean Squared Error

PCA Principal Component Analysis

PLS Partial Least Squares

PLSR Partial Least Squares Regression

RAVALT Rey Auditory Verbal Learning Test

ROI Region Of Interest

SCCA Sparse Canonical Correlation Analysis

SES Socioeconomic Status

SNP Single Nucleotide Polymorphism

SPLS Sparse Partial Least Squares

SPLS-DA Sparse Partial Least Squares Discrimination Analysis

SPM Statistical Parametric Map

SVD Singular Value Decomposition

VBM Voxel-Based Morphometry

WM White Matter

Chapter 1

Introduction

1.1 Neuroimaging and machine learning

For many years, one of the main limitations when studying the human brain was the fact that acquiring *in vivo* data was only possible using very invasive procedures (e.g. electro-physiological recordings). The introduction of neuroimaging techniques, such as Magnetic Resonance Imaging (MRI), drastically changed the field, by making it possible to acquire *in vivo* brain images in a non-invasive way.

MRI allows the acquisition of 3D brain images by using the magnetic properties of the atoms in the body, more specifically, the hydrogen nuclei in the water molecules. These images are comprised of many *voxels*, where each voxel can be seen as a pixel in a 3D space, i.e. whereas a pixel is a 2D square containing information about a small region of a 2D image, a voxel is a 3D cube containing information about the signal in a small region of a magnetic resonance (MR) image. Depending on the particular MRI acquisition sequence, different types of images can be obtained. One of the most commonly used types of structural MR images is the T1-weighted image, due to the fact that it provides a reasonable contrast between the different brain tissues, more specifically, between Grey Matter (GM) and White Matter (WM), and between GM and Cerebrospinal Fluid (CSF) [Chu, 2009].

The raw images provided by the MRI scanners cannot be directly used to perform a statistical analysis, as the MR signal intensities and brain shapes will be different between the images coming from different subjects. Therefore, structural MR images are subjected to a pre-processing pipeline before they are analysed, which is usually comprised of three main steps: segmentation, normalisation, and smoothing [Ashburner and Friston, 2005]. The segmentation step is applied to identify

the different types of tissue in the brain and create separate images based on the probability of each voxel corresponding to a particular tissue (e.g. GM, WM, CSF), this is done in order to normalise the values of each pixel to a range between 0 and 1 [Chu, 2009], enabling comparisons across different MR images. These probability maps are then normalised to a standard brain template, so that brains from different subjects, which have different shapes, can be analysed together. Finally, the images are usually subjected to a final spatial smoothing step, in order to remove some of the higher spatial frequency signal, which is usually associated with noise [Chu, 2009]. This smoothing is performed by performing a convolution between the images and a Gaussian kernel with a pre-determined Full Width at Half Maximum (FWHM), essentially "blurring" the images.

One of the most common ways to analyse structural differences in the pre-processed MR images is by using Voxel-Based Morphometry (VBM) [Ashburner and Friston, 2000], which allows one to determine which specific voxels in the brain are significantly correlated with the variable that is being tested, by performing a statistical test on each individual voxel. Although VBM is still very popular in neuroscience, it is an univariate approach, which means that each voxel is tested individually, without taking into account the interaction between all the voxels. In order to model this interaction, one has to use multivariate approaches, such as *machine learning methods*. These methods change the question that VBM tries to answer; instead of estimating which voxels are individually correlated with the effect, machine learning methods look for general patterns in the data which allow one to perform several tasks, including: classifying the subjects as belonging to a particular class (e.g. diseased subjects vs. healthy subjects), predicting a clinical/demographic score (e.g. age), gaining novel insights by finding associations between different types of data (MRI scans and clinical/demographic scores), etc.

Machine learning methods have been successfully used to analyse neuroimaging data for a variety of applications, including the study of neurological and psychiatric diseases. So far, however, most of these studies have focused on supervised binary classification problems, i.e. they attempt to summarise clinical assessment into a single measure (e.g. diagnostic classification) and the output of the models is limited to a probability value and, in most cases, a binary decision (e.g. healthy

vs. patient) [Ecker et al., 2010, Mourão-Miranda et al., 2005, Nouretdinov et al., 2011, Orrù et al., 2012, Rao et al., 2011, Klöppel et al., 2008]. This paradigm may present itself as a limitation when studying brain diseases whose underlying disease process is not yet completely understood and, therefore, might have an unreliable categorical classification. Indeed, this is a well known problem in psychiatry [Insel et al., 2010], where insights regarding the associations between brain and behaviour are still limited.

The lack of understanding regarding the associations between brain and behaviour presents itself as an interesting challenge for more exploratory machine learning approaches, such as eigen-decomposition methods, including several variants of Canonical Correlation Analysis (CCA) and Partial Least Squares (PLS). These methods can model the associations between brain and behaviour, by finding a latent subspace where these associations are the strongest.

1.2 Outline and contributions

The aim of this thesis is to explore the possibility of using eigen-decomposition approaches to find multivariate associative effects between brain structure and behaviour. Several variants of these methods will be tested, including versions that allow for sparse and non-linear solutions. As a proof of concept, all the methods explored in this thesis used dementia datasets, as the relationships between brain regions and behaviour are better understood in dementia, allowing for more reliable comparisons with previous studies.

The thesis is laid out as follows:

- Chapter 2 will introduce some basic machine learning concepts, with an emphasis on supervised learning. It will conclude with some remarks regarding the limitations of these methods, and the motivation for exploring the use of eigen-decomposition methods in this thesis.

- Chapter 3 will describe some of the most relevant eigen-decomposition methods used in this thesis. It will serve as a general introduction and review of these methods, before presenting the main contributions of this thesis in the following chapters.

- Chapter 4 proposes an alternative matrix deflation step to be used with Sparse

Partial Least Squares (SPLS), which tries to address a limitation of the deflation approach originally proposed by Witten et al. [2009]. Publication associated with the chapter: Monteiro et al. [2014].

- Chapter 5 describes an alternative permutation based framework to optimise the SPLS hyper-parameters, along with an earlier application of SPLS to dementia using different levels of sparsity for both neuroimaging and clinical/demographic data. Publication associated with the chapter: Monteiro et al. [2015].

- Chapter 6 takes the type of analysis performed in Chapter 5 a step further. In this chapter, a multiple-holdout framework is proposed, which is able to find robust multivariate associations between brain and behaviour. The framework was applied to a novel experimental setup, using whole-brain structural MRI data and individual items from a clinical questionnaire. This experimental setup allowed us to find multivariate associations between a subset of brain voxels, and a subset of the individual questionnaire items, which was not previously shown in the literature. Addition comparisons regarding the influence of the sparsity constraints and the influence of the matrix deflation step were also made. Publication associated with the chapter: Monteiro et al. [2016].

- Chapter 7 presents the background for Chapter 8. It proposes an adaptation of the Alternating Least Squares (ALS) algorithm to solve several Sparse CCA (SCCA) and SPLS optimisation problems. It also compares seven different eigen-decomposition methods, some of these comparisons have not been previously made in the literature.

- Chapter 8 proposes a novel primal-dual SCCA method. This method allows one to model the multivariate associations between brain and behaviour by using both sparsity constraints, and non-linear transformations of the data.

- Chapter 9 provides a general summary of the conclusions of this thesis, along with considerations for future work.

Chapter 2

Pattern analysis

The automatic detection of patterns in a dataset is an essential part of solving machine learning problems. These patterns are defined as any relations, regularities or structure that can be used to extract any meaningful information from the data [Shawe-Taylor and Cristianini, 2004]. In order to detect these patterns, a number of examples must be provided to a machine learning method. These examples are called *samples*, which can be represented as vectors with a certain *dimension*.

To illustrate these concepts, consider a dataset of whole-brain structural MRI scans from 100 patients which will be used as input to a machine learning method. The number of samples is equal to the number of subjects ($n = 100$) and the dimension of each sample is equal to the number of *features*. In this case, each feature corresponds to a voxel in the image, i.e. if the number of voxels is 100 000, then each sample lies on a 100 000 dimensional space: $\boldsymbol{x}_i \in \mathbb{R}^{1\times 100000}, i \in \{1,\ldots,n\}$. The dataset can be organised in a *data matrix* $\boldsymbol{X} \in \mathbb{R}^{n\times p}$, where n is the number of samples, p is the number of features, and each row of $\boldsymbol{X}$ corresponds to a sample $\boldsymbol{x}_i$.

The information contained in the data matrix $\boldsymbol{X}$ can be used for many different tasks. The type of machine learning method used depends on the task at hand. In neuroimaging, the most popular machine learning methods belong to two main types: supervised learning and unsupervised learning.

Supervised learning methods are used when one is trying to predict a value y_i associated with an example $\boldsymbol{x}_i$, i.e. these methods try to answer the following question: "Given an example $\boldsymbol{x}_i$, what is its associated y_i?". The type of supervised learning method used to answer this question will depend on the nature of what one is trying to predict (y_i), thus, supervised learning problems are sub-divided into

two main categories: classification and regression. The former deals with cases in which the aim is to predict whether an example belongs to a certain pre-defined class (e.g. healthy vs. diseased), while the latter deals with cases in which the aim is to predict a continuous score (e.g. age). Therefore, in a classification setting, y_i will be an integer number which denotes the *label* of $\boldsymbol{x}_i$ (e.g. $y_i = 0$: healthy vs. $y_i = 1$: diseased), whereas in a regression setting y_i will be a continuous value which denotes the *target* that one is trying to predict (e.g. age).

Unsupervised learning focuses on trying to find patterns in the data without any concern for labels or targets. These are normally used as exploratory approaches to find structure in the data which can then be used for several tasks, including data compression, data visualisation, clustering, or even as a step for further analysis using supervised learning.

The aim of this chapter is to make an overview of some of the general machine learning concepts, with an emphasis on supervised learning (Sections 2.1 to 2.2.3). The chapter will conclude with a few examples of applications of supervised learning methods in the neuroimaging literature, and some considerations regarding their general limitations when applied to clinical problems in neuroimaging (Section 2.3). In an attempt to address some of these limitations, this project has used more exploratory machine learning approaches, which shall be presented in Chapter 3.

2.1 Model training

Some of the concepts presented in this chapter are applicable in both supervised and unsupervised learning settings. However, in order to keep the introduction to a reasonable size, the examples will be given in a supervised learning setting. More specifically, in a regression setting, as some of the concepts associated with regression problems will be important for Chapters 7 and 8.

Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ be a data matrix containing n samples with p features (with $n > p$), and $\boldsymbol{y} \in \mathbb{R}^{n \times 1}$ be a vector with the targets associated with each sample. In a supervised learning setting, the aim is to find a function f that maps each input $\boldsymbol{x}_i \in \boldsymbol{X}$ to each output $y_i \in \boldsymbol{y}$. The trivial solution to this problem would be to make an exact map of each input to its corresponding target ($f(\boldsymbol{x}_i) = y_i \, \forall \, \{1, \ldots, n\}$). However, one also wishes to make f a general function that can handle samples $\boldsymbol{x}'$ which were not in the original dataset $\boldsymbol{X}$. In other words, the aim is to train a model

f which, given a new example $\boldsymbol{x}'$, makes a reasonable prediction of its target y':

$$f: \boldsymbol{X} \rightarrow \boldsymbol{y}$$
$$f: \boldsymbol{x}' \rightarrow y'$$

The distance between the values given by $f(\boldsymbol{X})$ and the actual targets $\boldsymbol{y}$ is given by the *loss function* L. The optimal function f will be the one which minimises the loss:

$$f = \arg\min_{f} \; L(f, \boldsymbol{y}) \tag{2.1}$$

In some settings, one may train a model f which perfectly fits the data ($f(\boldsymbol{x}_i) = y_i \; \forall \, i \in \{1, \ldots, n\}$) while providing poor predictions for new examples $\boldsymbol{x}'$. In these cases, it is said that the model f is too complex and it *overfits* the data. One of the ways to avoid these scenarios, is by penalising solutions of f which are too complex, thus, an extra *penalty function* P is added to Equation 2.1. The loss function (L) plus the penalty function (P) will constitute the *objective function* J that one tries to minimise:

$$f = \arg\min_{f} \; J(f, \boldsymbol{y}) \;\; \Leftrightarrow \;\; f = \arg\min_{f} \; L(f, \boldsymbol{y}) + \gamma P(f) \tag{2.2}$$

where γ is the *hyper-parameter* of the model, which controls the trade-off between the loss and the penalty.

In general, more complex functions f lead to lower losses L, which decrease the cost J (Equation 2.2). However, the penalty term P increases with the complexity of f, which results in the increase of J. Therefore, in order to minimise J, there must be a compromise between the loss and the complexity. Note that if the penalty hyper-parameter γ is too small, the complexity of f will be too high, and the model will overfit. However, if γ is too large, then f will not have enough complexity to properly model the data, and the model f will *underfit*.

There have been several loss functions (L) and penalty functions (P) proposed in the literature, which contain different properties, leading to different solutions. In order to better illustrate these concepts, a regression example is provided below.

2.1.1 Linear regression

Consider a case in which one wishes to find a function f that uses the information from a sample vector $\boldsymbol{x}_i \in \boldsymbol{X}$ to predict a continuous target variable y_i. One of the ways to solve this problem is by using a model called *linear regression*. This model uses a loss function known as the *squared loss*:

$$L = \sum_{i=1}^{n} (y_i - f(\boldsymbol{x}_i))^2 \tag{2.3}$$

Since the model is linear, this implies that f is a linear function, thus, the predictions provided by f consist of a weighted linear combination of the input features:

$$f(\boldsymbol{x}) = \sum_{j=1}^{p} (x_j w_j) + b \tag{2.4}$$

where w_j are known as the *weights*, and b is known as the *bias* term.

If the targets $\boldsymbol{y}$ and the features of $\boldsymbol{X}$ are mean-centered, i.e. the mean of the targets $\boldsymbol{y}$ is equal to zero and the mean of each column of $\boldsymbol{X}$ is equal to zero, then $b = 0$ and Equation 2.4 can be re-written as:

$$f(\boldsymbol{x}) = \boldsymbol{x}\boldsymbol{w}$$

where $\boldsymbol{w} \in \mathbb{R}^{p \times 1}$ is known as the *weight vector*. By writing the problem using matrix notation, the squared loss (Equation 2.3) can be re-written as:

$$L(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{w}) = \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2$$

where $\|\cdot\|_2$ denotes the l_2-norm (also known as the Euclidean norm).

In its simplest version, linear regression does not contain a penalty term, therefore, the objective function will be equal to the loss function, i.e. $J = L$. In order to find the minimum of the objective function J, one has to find the weight vector $\boldsymbol{w}$ which makes the function f model the data with the smallest possible loss. The problem can thus be written as the following *optimisation problem*:

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \; J(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{w}) \;\Leftrightarrow\; \boldsymbol{w} = \arg\min_{\boldsymbol{w}} \; \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 \tag{2.5}$$

The optimisation problem expressed in Equation 2.5 can be re-written as:

$$J(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{w}) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w})^\intercal(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}) \tag{2.6}$$

which can then be differentiated to calculate its minimum [Shawe-Taylor and Cristianini, 2004]:

$$\begin{aligned} \frac{\partial J}{\partial \boldsymbol{w}} &= \mathbf{0} \\ -2\boldsymbol{X}^\intercal\boldsymbol{y} + 2\boldsymbol{X}^\intercal\boldsymbol{X}\boldsymbol{w} &= \mathbf{0} \\ \boldsymbol{X}^\intercal\boldsymbol{X}\boldsymbol{w} &= \boldsymbol{X}^\intercal\boldsymbol{y} \\ \boldsymbol{w} &= (\boldsymbol{X}^\intercal\boldsymbol{X})^{-1}\boldsymbol{X}^\intercal\boldsymbol{y} \end{aligned} \tag{2.7}$$

If $p = 1$, then the solution will be a line whose fit minimises the squared distance between y_i and $f(x_i)$, an example can be seen in Figure 2.1. For higher dimensional problems ($p > 1$), the solution given by $\boldsymbol{w}$ will correspond to a hyperplane.

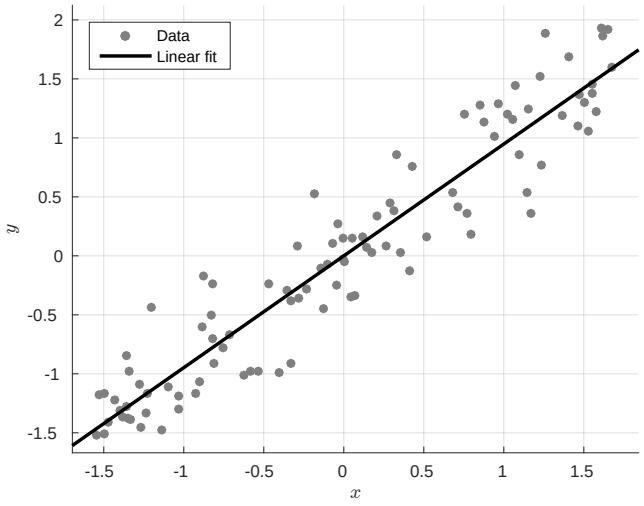


Figure 2.1: Example of a linear regression.

The solution to the optimisation problem expressed in Equation 2.5 may overfit the data, thus, it is often desirable to add a penalty term. This is especially true if $p > n$, which is known as an ill-conditioned problem, where the optimisation problem will not have an unique solution.

There are several types of penalties that can be chosen, which will result in different solutions for $\boldsymbol{w}$. One of these is the l_2-norm penalty ($P(\boldsymbol{w}) = \|\boldsymbol{w}\|_2^2$), also

known as the *ridge penalty*. By using this penalty, one has to re-write the problem expressed in Equation 2.5 as:

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \ \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \gamma\|\boldsymbol{w}\|_2^2 \tag{2.8}$$

Following the same procedure used to derive Equation 2.7, one obtains the following solution:

$$\boldsymbol{w} = (\boldsymbol{X}^\intercal\boldsymbol{X} + \gamma\boldsymbol{I})^{-1}\boldsymbol{X}^\intercal\boldsymbol{y}$$

where $\boldsymbol{I} \in \mathbb{R}^{p\times p}$ is an identity matrix.

The ridge penalty provides solutions for $\boldsymbol{w}$ in which all the features are included in the model, and the ones that are not informative are down-weighted. However, in some cases, it might be desirable to obtain a *sparse* solution, i.e. a solution where the contribution of certain features is set to zero ($w_j = 0$). These approaches perform *feature selection*, which is a desirable property to have if only a subset of the features contains relevant information, since non-informative features are excluded from the model.

One of the most popular sparse penalty functions is the l_1-norm penalty. When used in a regression setting, it gives rise to the popular method known as Least Absolute Shrinkage and Selection Operator (LASSO) [Tibshirani, 1996]:

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \ \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \gamma\|\boldsymbol{w}\|_1 \tag{2.9}$$

Despite the popularity of the LASSO, one of its disadvantages is that it tends to exclude variables which are correlated with other variables already included in the model, e.g. if feature x_1 is correlated with feature x_2, the tendency is for only one of the features to be included: $w_1 \neq 0, w_2 = 0$; or $w_1 = 0, w_2 \neq 0$. This may lead to informative features being excluded from the model, based solely on the fact they are correlated with other informative features. In order to address this issue, Zou and Hastie [2005] proposed an approach known as the *elastic net*, which consists in adding an extra l_2-norm penalty to the LASSO optimisation problem, resulting in a

method that provides sparse solutions while maintain a grouping effect:

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \ \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \gamma_1\|\boldsymbol{w}\|_1 + \gamma_2\|\boldsymbol{w}\|_2^2$$

where γ_1 and γ_2 denote the hyper-parameters that control the influence of the l_1-norm and l_2-norm penalties, respectively.

The description of the algorithms available to solve the LASSO and elastic net optimisation problems is beyond of the scope of this thesis, for more details, please refer to the papers by Efron et al. [2004] and Friedman et al. [2010].

2.1.2 Kernel methods

So far, all the methods presented were linear methods, i.e. the solution consists of a linear combination of the features, which is expressed by the weight vector. However, sometimes the patterns of interest in the data are non-linear. This means that linear methods may not be able to estimate a model f which is able to predict well the labels/targets y' for new data points $\boldsymbol{x}'$. One of the ways to address these situations is by mapping the data in the original input space into a higher dimensional space, where the problem is linearly solvable.

Consider the classification example on the left hand side of Figure 2.2. In this case, the aim is to classify the two groups of data points (red and blue) by finding a function f which defines a boundary separating the two groups. As one can see, it is impossible to do this with a linear function, i.e. there is no straight line which is able to separate the two groups (Figure 2.2 - left). However, if a new variable is introduced $x_3 = x_1^2 + x_2^2$, then the groups are linearly separable using a plane (Figure 2.2 - right).

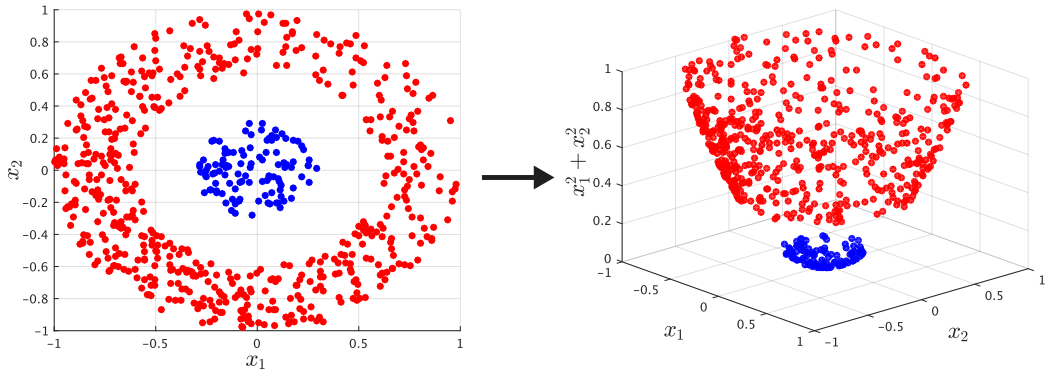


Figure 2.2: Mapping a dataset into a higher dimensional feature space. A new feature $x_3 = x_1^2 + x_2^2$ is introduced, making the classes linearly separable.

Performing this mapping can be done explicitly, by introducing new features (Figure 2.2), or done implicitly, by using a kernel.

This section will give a brief overview of some of the properties of kernels, and how they can be used in machine learning methods.

2.1.2.1 Kernels

In order to understand how kernels are constructed, consider an hypothetical input vector $\boldsymbol{x} \in \mathbb{R}^p$, and a function ϕ that maps this vector into a feature space $F \subseteq \mathbb{R}^P$, where $P \geqslant p$:

$$\phi : \boldsymbol{x} \in \mathbb{R}^p \mapsto \phi(\boldsymbol{x}) \in F \subseteq \mathbb{R}^P$$

A kernel function takes two input vectors $\boldsymbol{x}_i, \boldsymbol{x}_j \in \boldsymbol{X}$, maps them into a higher dimensional feature space and performs the dot product between the two:

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$$

The results are then stored in a *kernel matrix* (also known as a *kernel*):

$$\boldsymbol{K} = \begin{bmatrix} \kappa(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & \kappa(\boldsymbol{x}_1, \boldsymbol{x}_n) \\ \vdots & \ddots & \vdots \\ \kappa(\boldsymbol{x}_n, \boldsymbol{x}_1) & \cdots & \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n) \end{bmatrix}$$

Thus, the kernel can be seen as a matrix containing information about how close $\boldsymbol{x}_i$ is to $\boldsymbol{x}_j$ in the feature space F.

The type of kernel depends on the transformation $\phi(\cdot)$ that the samples are subjected to. If a linear kernel is used, then no transformation is performed on the input vectors prior to the dot product ($\phi(\boldsymbol{x}) = \boldsymbol{x}$). Thus, the kernel function will simply be the dot product of the arguments:

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$$

In this case, the results will be equivalent to running the machine learning method in the original input space [Shawe-Taylor and Cristianini, 2004]. The kernel can be

computed by multiplying the input data matrix with its transpose:

$$\boldsymbol{K} = \boldsymbol{X}\boldsymbol{X}^{\intercal} \tag{2.10}$$

This will provide some computational advantages for cases in which $p \gg n$, due to the fact that one will work with the kernel matrix $\boldsymbol{K} \in \mathbb{R}^{n \times n}$, instead of the larger data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$. However, this is not the main strength of kernel methods.

The main reason why these methods are popular, is due to a property known as the *kernel trick*, which allows one to obtain a mapping into a higher dimensional feature space F without explicitly performing the mapping $\phi(\cdot)$ in the original input space. This can be done by computing the kernel $\boldsymbol{K}$ using (2.10) and then applying a transformation on $\boldsymbol{K}$.

Some of the more popular types of non-linear kernels include the polynomial kernel (κ_d) and the Gaussian kernel (κ_G):

$$\begin{gathered} \kappa_d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle + R\right)^d \\ \text{and} \\ \kappa_G(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(\frac{\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle}{\sigma^2} - \frac{\langle \boldsymbol{x}_i, \boldsymbol{x}_i \rangle}{2\sigma^2} - \frac{\langle \boldsymbol{x}_j, \boldsymbol{x}_j \rangle}{2\sigma^2}\right) \end{gathered} \tag{2.11}$$

where d is the degree of the polynomial kernel, R is a hyper-parameter which controls the influence of the linear terms for the polynomial kernel, and σ is an hyper-parameter which controls the flexibility of the Gaussian kernel [Shawe-Taylor and Cristianini, 2004].

Note that the kernel matrix $\boldsymbol{K}$ contains all the information necessary to compute the distance between two vectors. However, some information is lost when the input matrix $\boldsymbol{X}$ is converted to $\boldsymbol{K}$, namely, the orientation of the dataset with respect to the origin [Shawe-Taylor and Cristianini, 2004]. In order to make the concept of the kernel trick clearer, an example is provided below.

Example: Non-linear mapping using a polynomial kernel

Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ denote a data matrix with n samples and p features. For a low dimensional problem (e.g. $p = 2$), if one would want to include all the possible

combinations of polynomials of degree 2, one could perform the following mapping:

$$\phi(\boldsymbol{x}) = [x_1^2, x_2^2, \sqrt{2}x_1x_2, \sqrt{2R}x_1, \sqrt{2R}x_2, R] \tag{2.12}$$

As mentioned previously (Equation 2.11), the term R controls the contribution of the linear terms, i.e. the larger it is, the stronger the influence of the linear terms is, which should help prevent overfitting.

As one can see in Equation 2.12, this mapping increased the number of variables. However, one could use a kernel instead, which avoids performing this explicit mapping.

Let $\boldsymbol{x}$ and $\boldsymbol{z}$ be two row vectors from matrix $\boldsymbol{X}$ (i.e. two samples from the dataset). The dot product between $\langle\phi(\boldsymbol{x}), \phi(\boldsymbol{z})\rangle$ is equivalent to:

$$\langle\phi(\boldsymbol{x}), \phi(\boldsymbol{z})\rangle = (\langle\boldsymbol{x}, \boldsymbol{z}\rangle + R)^d$$

The proof for this equivalence is done by expanding the polynomial kernel κ_d (Equation 2.11) using the binomial theorem [Shawe-Taylor and Cristianini, 2004]:

$$\begin{aligned}
\kappa_d(\boldsymbol{x}, \boldsymbol{z}) = (\langle\boldsymbol{x}, \boldsymbol{z}\rangle + R)^d &= \sum_{s=0}^{d} \binom{d}{s} R^{d-s} \langle\boldsymbol{x}, \boldsymbol{z}\rangle^s = \\
&= \binom{2}{0} R^2 \langle\boldsymbol{x}, \boldsymbol{z}\rangle^0 + \binom{2}{1} R^1 \langle\boldsymbol{x}, \boldsymbol{z}\rangle^1 + \binom{2}{2} R^0 \langle\boldsymbol{x}, \boldsymbol{z}\rangle^2 \\
&= R^2 + 2R\langle\boldsymbol{x}, \boldsymbol{z}\rangle + \langle\boldsymbol{x}, \boldsymbol{z}\rangle^2 \\
&= R^2 + 2R(x_1z_1 + x_2z_2) + (x_1z_1 + x_2z_2)^2 \\
&= R^2 + 2Rx_1z_1 + 2Rx_2z_2 + x_1^2z_1^2 + 2x_1z_1x_2z_2 + x_2^2z_2^2
\end{aligned} \tag{2.13}$$

Note that performing the explicit mapping of $\boldsymbol{x}$ and $\boldsymbol{z}$ using Equation 2.12, and then performing the dot product ($\langle\phi(\boldsymbol{x}), \phi(\boldsymbol{z})\rangle$) will lead to the same expression in the last line of Equation 2.13, therefore, both operations are equivalent. However, using the kernel trick is much more efficient, as the number of non-linear features will grow very large with the number of features in $\boldsymbol{X}$ and the degree of the polynomial.

A kernel method is essentially a machine learning method which takes kernels $\boldsymbol{K}$ as inputs, instead of the original data matrices $\boldsymbol{X}$. The algorithms used in kernel methods work with different types of kernels, making the methods more

modular [Shawe-Taylor and Cristianini, 2004], i.e. one can use different types of linear and non-linear models simply by swapping the kernel that is used. In order to illustrate some of the concepts associated with kernel methods, the adaptation of ridge regression (Equation 2.8) into its kernel formulation is presented below.

2.1.2.2 Kernel Ridge Regression (KRR)

The ridge regression problem expressed in Equation 2.8 can be adapted to be solved using kernels. The first step is to write the objective function in matrix notation (similar to Equation 2.6):

$$J(\boldsymbol{X},\boldsymbol{y},\boldsymbol{w}) = (\boldsymbol{y}-\boldsymbol{X}\boldsymbol{w})^\intercal(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{w})+\gamma\boldsymbol{w}^\intercal\boldsymbol{w}$$

Then, the problem is transformed from its *primal representation*, into its *dual representation*, by representing the weight vectors $\boldsymbol{w}$ as a combination of the training samples, $\boldsymbol{w} = \boldsymbol{X}^\intercal\boldsymbol{\alpha}$:

$$J(\boldsymbol{X},\boldsymbol{y},\boldsymbol{\alpha}) = (\boldsymbol{y}-\boldsymbol{X}\boldsymbol{X}^\intercal\boldsymbol{\alpha})^\intercal(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{X}^\intercal\boldsymbol{\alpha})+\gamma\boldsymbol{\alpha}^\intercal\boldsymbol{X}\boldsymbol{X}^\intercal\alpha \tag{2.14}$$

By using Equation 2.10, one can re-write Equation 2.14 as:

$$J(\boldsymbol{K},\boldsymbol{y},\boldsymbol{\alpha}) = \boldsymbol{y}^\intercal\boldsymbol{y}-\boldsymbol{y}^\intercal\boldsymbol{K}\boldsymbol{\alpha}-\boldsymbol{\alpha}^\intercal\boldsymbol{K}\boldsymbol{y}+\boldsymbol{\alpha}^\intercal\boldsymbol{K}^2\boldsymbol{\alpha}+\gamma\boldsymbol{\alpha}^\intercal\boldsymbol{K}\boldsymbol{\alpha} \tag{2.15}$$

This in turn can be differentiated with respect to $\boldsymbol{\alpha}$ and set to $\mathbf{0}$, in order to find its minimum:

$$\begin{aligned}
\frac{\partial J}{\partial \boldsymbol{\alpha}} &= \mathbf{0} \\
\mathbf{0}-\boldsymbol{K}\boldsymbol{y}-\boldsymbol{K}\boldsymbol{y}+2\boldsymbol{K}^2\boldsymbol{\alpha}+2\gamma\boldsymbol{K}\boldsymbol{\alpha} &= \mathbf{0} \\
-2\boldsymbol{K}\boldsymbol{y}+2\boldsymbol{K}(\boldsymbol{K}+\gamma\boldsymbol{I})\boldsymbol{\alpha} &= \mathbf{0} \\
\boldsymbol{\alpha} &= (\boldsymbol{K}+\gamma\boldsymbol{I})^{-1}\boldsymbol{y}
\end{aligned}$$

where $\boldsymbol{I} \in \mathbb{R}^{n\times n}$ is an identity matrix.

This method is known as the Kernel Ridge Regression (KRR), it has a *dual solution*, given by $\boldsymbol{\alpha}$, which is referred to as a *dual variable* [Shawe-Taylor and Cristianini, 2004]. Its solution is equivalent to the one provided by ridge regression

(as described in Section 2.1.1) if $\boldsymbol{K}$ corresponds to a linear kernel. In this case, the *primal variable* $\boldsymbol{w}$ can be recovered using the dual variable $\boldsymbol{\alpha}$, i.e. $\boldsymbol{w} = \boldsymbol{X}^\intercal \boldsymbol{\alpha}$.

As previously mentioned, one of the advantages of kernel methods is that $\boldsymbol{K}$ can be non-linear, which means that the optimisation problem will be equivalent to solving a ridge regression problem in a non-linear, higher dimensional feature space F. This may provide better results, if the patterns that one is trying to detect are non-linear. However, it may lead to overfitting, and moreover, it will not be straightforward to recover the weights $\boldsymbol{w}$ in the original input space.

2.2 Model validation

As mentioned in Section 2.1, when training a supervised machine learning model, the main goal is to find a function f which, given a new sample $\boldsymbol{x}'$, is able to make a reasonable prediction of its corresponding label/target y'. In order to check if the training was successful, the ability of the model to generalise for new data should be assessed, i.e. one should validate the model. For a classification problem, this can be formulated as the following question: "Given a new set of data $\{\boldsymbol{X}', \boldsymbol{y}'\}$ with N new samples, how many samples does the model f correctly classify?". More specifically, one wants to determine the fraction of times $f(\boldsymbol{x}'_i) = y'_i$. This metric is known as the *accuracy* of the model.

In the case of a regression problem, the question one is trying to answer is: "Given a new set of data $\{\boldsymbol{X}', \boldsymbol{y}'\}$ with N new samples, how large is the distance between the predictions $f(\boldsymbol{x}'_i)$ and the true targets y'_i?". More specifically, one is interested in quantifying the error that the model f makes when predicting the targets $\boldsymbol{y}'$. One of the most commonly used metrics to quantify this error is the Mean Squared Error (MSE):

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y'_i - f(\boldsymbol{x}'_i))^2$$

The strategies used for validation will depend not only on the research question, but also on the model itself and the amount of data available. One of the main factors which influences the choice in validation strategy is the existence of hyper-parameters that have to be optimised.

A brief overview of the classical methods used for model validation (with and

without hyper-parameter optimisation) will be made in the following sections.

2.2.1 Validation without hyper-parameter optimisation

A hyper-parameter will be defined as any term which controls the complexity of a model. For example, this could be the penalty hyper-parameter γ (as defined in Equation 2.2), or an hyper-parameter which controls the complexity of a kernel, i.e. the d hyper-parameter for polynomial kernels, or the σ hyper-parameter for Gaussian kernels (as described in Equation 2.11).

Let f be a model without hyper-parameters (or with hyper-parameter values fixed *a priori*) that one wishes to validate, the simplest ways to validate a model, is simply to take all the available data and split it into a train set and a test set (Figure 2.3). The model f is then trained using the train data, and the test data is used to compute the performance metric (e.g. accuracy, MSE, etc.).

Figure 2.3: Single train/test data split scheme.

This approach will usually work well if the dataset has a large number of samples. However, if this is not the case, then the estimated model may not be able to achieve a good performance, since it will be trained using a very small dataset. Moreover, the estimation of the model's performance may be very different from the "true" performance, since a very small dataset is used to estimate it.

The datasets available in neuroimaging usually do not contain enough samples to justify using the approach described in Figure 2.3. One of the most common ways to address this issue is to use an approach known as k-fold Cross-Validation (k-fold CV). The process is illustrated in Figure 2.4, which starts by splitting the data into k non-overlapping sets with the same size. For the first fold ($k = 1$) the set k is left out of the dataset ($\{\boldsymbol{X}_k, \boldsymbol{y}_k\}$) and the model is trained with the remaining data ($\{\boldsymbol{X}_{(-k)}, \boldsymbol{y}_{(-k)}\}$). Then, the set of data that was left out is used to compute the performance metric. The process is repeated for each fold, and the performance is estimated based on the average of the performances across all the test folds.

Due to the scarce amount of data available in neuroimaging, it is very common to use an extreme case of CV where $k = n$, i.e. the number of folds is equal to the

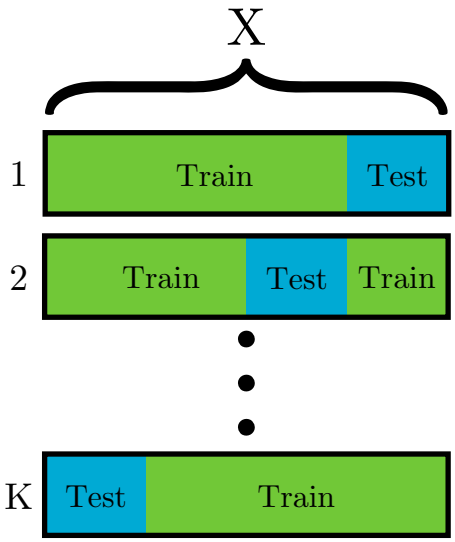


Figure 2.4: k-fold cross-validation scheme.

number of subjects. This is known as the leave-one-out CV. From all the classical CV approaches, this provides the greatest amount of data available for training at each fold. However, due to the large similarity between the train sets, the estimation of the model's performance will have a very high variance [Hastie et al., 2009]. Although this approach is still very popular in neuroimaging, it has recently received some criticism in the literature, e.g. Varoquaux et al. [2017] have shown that it leads to biased and unstable results. One of the alternatives to k-fold CV is to use several random splits of the data, instead of non-overlapping folds. This idea is further explored in Chapter 6.

2.2.2 Validation with hyper-parameter optimisation

Sometimes, it is necessary to add penalty terms to a machine learning model in order to prevent overfitting (Section 2.1). The penalties are controlled by one or more hyper-parameters, whose values will affect the solutions provided by the model. Therefore, it is usually desirable to optimise these hyper-parameters, in order to obtain the best performance.

Let f be a model with one hyper-parameter to optimise. In this setting, the aim is usually to select the value of this hyper-parameter from a set of pre-defined candidates (e.g. $\gamma \in \{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$), such that f achieves the best possible performance for new data. Although this may seem like a trivial task, it has to be performed carefully. If one were to perform a single train/test split (Figure 2.3) to

train several models using each one of the candidates (e.g. $\{f_{\gamma=10^{-3}}, \ldots, f_{\gamma=10^{3}}\}$), and then select the one with the best performance on the test data, one would probably be over-estimating the performance of the model. This is due to the fact that the hyper-parameter was optimised based on the performance of model f for this particular test set, and not for an independent test set which was not used before. Therefore, the validation schemes described in the previous section (Figures 2.3 and 2.4) should be modified to accommodate for the hyper-parameter optimisation step.

The train/test split validation scheme (Figure 2.3) can be adapted to the three-way split validation scheme (Figure 2.5). In this validation scheme, the data are split in three sets (Figure 2.5): train, test, and hold-out. The first, is used to train the model with the different candidates for the hyper-parameter values. The performance of each hyper-parameter value is assessed on the test set. Finally, the model performance is estimated on the hold-out dataset.

Figure 2.5: Three-way split validation scheme.

By using this splitting approach, one guarantees that the data which were used to evaluate the different hyper-parameter values (test data), are not the same as the data that were used to evaluate the final model (hold-out data). However, the three-way split validation suffers from the same problem as the single train/test split (Figure 2.3): it will not work well when the amount of available data is small.

One of the ways to validate a model with hyper-parameter optimisation for small datasets, is to adapt the CV procedure to accommodate for the hyper-parameter optimisation step. This type of CV is known as nested cross-validation (nested-CV), and is summarised in Figure 2.6. The procedure starts by splitting the dataset in a train set and in a test set, these are known as the outer folds. Then, for each candidate value of γ, the train set $\{\boldsymbol{X}_{(-k)}, \boldsymbol{y}_{(-k)}\}$ is subjected to an inner CV with K_{in} inner folds. The value of γ that leads to the best performance on the inner CV is fixed and used to train the model f using the outer train fold $\{\boldsymbol{X}_{(-k)}, \boldsymbol{y}_{(-k)}\}$, and the performance is estimated on the outer test fold $\{\boldsymbol{X}_k, \boldsymbol{Y}_k\}$. The process is then repeated for all K_{out} folds, and the overall model performance is estimated as the

average performance on all the outer test folds.

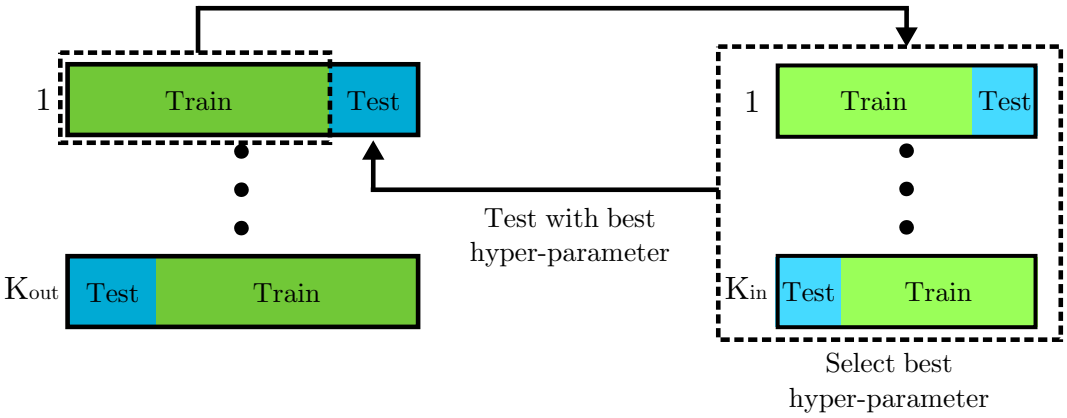


Figure 2.6: Nested cross-validation scheme. Sets represented with lighter shades of blue and green correspond to the sets used for hyper-parameter selection in each inner CV fold.

This procedure can be quite computationally expensive, since several inner CVs have to be performed for each fold of the outer CV. Some strategies to decrease the computational time include using a smaller number of inner folds ($K_{\text{in}} < K_{\text{out}}$) and a small number of candidates for the hyper-parameter values. However, it can still be quite computationally expensive, especially if there is more than one hyper-parameter that has to be optimised. This will be further discussed in Chapter 6.

2.2.3 Permutation tests

After one estimates a model's performance, it is often desirable to assess if that performance could have been achieved by chance alone. One of the ways to assess this is by performing a *permutation test*.

The test consists of randomly permuting the order of the samples while maintaining the order of the targets unchanged, i.e. generating a new data matrix $\boldsymbol{X}^*$ in which the order of the rows no longer corresponds to the order of the elements on $\boldsymbol{y}$. Then, the validation scheme is repeated B times, permuting the order of the rows of $\boldsymbol{X}$ each time, and the performance c_b for each permutation is saved. Finally, the p-value associated with the permutation test is computed based on the number of times the models trained using the permuted datasets performed at least as well as the model trained using the non-permuted data:

$$p = \frac{1 + \sum_{b=1}^{B} 1_{c_b \geq c}}{1 + B} \tag{2.16}$$

where B is the number of permutations, c is the performance metric obtained when training the model using the non-permuted data, and c_b is the performance metric obtained when training the models using the permuted dataset during permutation b. The addition of 1 in the nominator and denominator of Equation 2.16 is equivalent to including c in the permutations. This will guarantee that $p > 0$ and that a minimum number of permutations will have to be performed in order to obtain a low enough p-value.

Note that the inequality $c_b \geqslant c$ in Equation 2.16 only makes sense if the metric used to evaluate the model increases with the increase in performance (e.g. accuracy). For metrics which decrease with the increase in model performance (e.g. MSE), the inequality is changed to $c_b \leqslant c$.

The model's performance is considered statistically significant if p is smaller than a pre-defined threshold α. In the neuroscience literature, this value is usually set to $\alpha = 0.05$.

2.3 Applications in neuroimaging

The use of supervised machine learning approaches in neuroimaging is quite widespread. These have been used not only in neuroscience applications [Mourão-Miranda et al., 2005], but also in clinical applications, such as: autism [Ecker et al., 2010], depression [Costafreda et al., 2009, Nouretdinov et al., 2011], schizophrenia [Mourao-Miranda et al., 2012], Alzheimer's disease [Klöppel et al., 2008, Davatzikos et al., 2008, Rao et al., 2011], etc. Although supervised machine learning methods have been used during this project [Schrouff et al., (accepted)], they are not the main focus of this thesis, for a detailed review of some of the earlier applications of supervised learning in neuroimaging, please refer to the papers by Pereira et al. [2009] and Orrù et al. [2012].

Despite their valuable contributions to the study of several brain diseases, these methods still have one fundamental limitation: they rely on the labeling of the subjects. More specifically, they only allow one to study a disease within the framework of pre-defined clinical labels. In some fields, e.g. psychiatry, the labels provided by current diagnostic tools are considered to be unreliable and, not only have they failed to predict treatment response, but they have also failed to align with results from neuroscience and genetics [Insel et al., 2010].

By using more exploratory machine learning approaches, one could potentially gain new insights into different brain disorders, which could help redefine the way diagnosis is performed. Some of the methods that can be used to try to address this issue are called *eigen-decomposition methods*. These will be the focus of the next chapter.

Chapter 3

Eigen-decomposition methods

Not all machine learning problems fall under the category in which one tries to estimate a function f that predicts a single label or target y based on an input $\boldsymbol{x}$. In some cases, one might wish to analyse the data in an exploratory way, in order to find novel multivariate relationships between two sets of variables. These relationships might help us to better understand the observed phenomenon, and discover underlying patterns in the data. One way to achieve this is by finding components which encode these relationships, these can be computed using a few methods generally referred to as eigen-decomposition methods.

This chapter will start by introducing some of the basic concepts necessary to understand eigen-decomposition methods (Section 3.1). It will then give a brief overview of some of these methods (Sections 3.1.1, 3.1.2, 3.2, 3.3), and conclude by describing some of its applications in neuroimaging (Section 3.5).

3.1 Eigenvalues and eigenvectors

The first step to understand eigen-decomposition methods, is to understand the concepts of eigenvalues and eigenvectors. This section aims to provide a basic introduction to these concepts, along with an explanation of two eigen-decomposition methods: Singular Value Decomposition (Section 3.1.1) and Principal Component Analysis (Section 3.1.2). These will be important to introduce the main eigen-decomposition methods in the following sections (Sections 3.2 to 3.4).

Let $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ be a non-singular matrix, i.e. invertible. λ and $\boldsymbol{v}$ are an *eigenvalue* and *eigenvector* of $\boldsymbol{A}$ if:

$$\boldsymbol{A}\boldsymbol{v} = \lambda \boldsymbol{v}$$

where $\|\boldsymbol{v}\|_2 = 1$ and $\boldsymbol{v} \neq \boldsymbol{0}$ [Shawe-Taylor and Cristianini, 2004]. This means that if the matrix $\boldsymbol{A}$ is projected onto its eigenvector $\boldsymbol{v}$, the result will be the scaling of $\boldsymbol{v}$ by the eigenvalue λ, i.e. $\boldsymbol{v}$ will not change direction.

Note that a matrix may contain more than one eigenvalue/eigenvector pair $\{\lambda_h, \boldsymbol{v}_h\}$. For symmetric matrices (i.e. $\boldsymbol{A} = \boldsymbol{A}^\intercal$), different eigenvectors $\boldsymbol{v}_h$ are orthogonal: $\langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle = 0 \ \forall \ i \neq j$. These pairs are often obtained individually, using the first pair $\{\lambda_1, \boldsymbol{v}_1\}$ (corresponding to the largest eigenvalue) to transform matrix $\boldsymbol{A}$ before obtaining the next pair $\{\lambda_2, \boldsymbol{v}_2\}$, and so on: $\{\{\lambda_1, \boldsymbol{v}_1\}, \ldots, \{\lambda_k, \boldsymbol{v}_k\}\}$, where $\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_k$. This is done using a process called *matrix deflation*, where the pair $\{\lambda_h, \boldsymbol{v}_h\}$ is used to transform $\boldsymbol{A}_h$, such that the corresponding λ_h is shrank to zero, and then computing the next eigenvalue/eigenvector pair [Shawe-Taylor and Cristianini, 2004]:

$$\boldsymbol{A}_{h+1} = \boldsymbol{A}_h - \lambda_h \boldsymbol{v}_h \boldsymbol{v}_h^\intercal \tag{3.1}$$

Note that $\boldsymbol{v}_h$ is orthogonal to the subspace spanned by $\boldsymbol{A}_{h+1}$:

$$\boldsymbol{A}_{h+1}\boldsymbol{v}_h = \left(\boldsymbol{A}_h - \lambda_h \boldsymbol{v}_h \boldsymbol{v}_h^\intercal\right) \boldsymbol{v}_h = \boldsymbol{A}_h \boldsymbol{v}_h - \lambda_h \boldsymbol{v}_h \boldsymbol{v}_h^\intercal \boldsymbol{v}_h = \lambda_h \boldsymbol{v}_h - \lambda_h \boldsymbol{v}_h = \boldsymbol{0}$$

The matrix deflation method expressed in Equation 3.1 is sometimes referred to as the Hotelling's deflation [Mackey, 2008]. This method will be further explored in Chapter 4, with special emphasis on cases in which the properties of Equation 3.1 do not hold, and some practical consequences of this.

If all the eigenvectors are grouped as columns of a matrix $\boldsymbol{V}$, and the eigenvalues are placed in a diagonal matrix $\boldsymbol{\Lambda}$, with $\boldsymbol{\Lambda}(h,h) = \lambda_h$, then:

$$\boldsymbol{A}\boldsymbol{V} = \boldsymbol{V}\boldsymbol{\Lambda} \tag{3.2}$$

Equation 3.2 is known as the *eigen-decomposition* of $\boldsymbol{A}$ [Shawe-Taylor and Cristianini, 2004]. Note that $\boldsymbol{V}$ is an *orthonormal matrix*, i.e. it exhibits the properties of orthogonal matrices ($\boldsymbol{V}\boldsymbol{V}^\intercal = \boldsymbol{V}^\intercal\boldsymbol{V} = \boldsymbol{I} \implies \boldsymbol{V}^\intercal = \boldsymbol{V}^{-1}$), and each column $\boldsymbol{v}_h$ of $\boldsymbol{V}$ has l_2-norm equal to 1 ($\|\boldsymbol{v}_h\|_2 = 1$). This means that Equation 3.2 can be written as:

$$\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^\intercal \tag{3.3}$$

Therefore, matrix $\boldsymbol{A}$ can be decomposed using its eigenvalues and eigenvectors. Moreover, one could compute an approximation of matrix $\boldsymbol{A}$ by using a subset of its top eigenvalue/eigenvector pairs.

There are several methods that can be used to perform an eigen-decomposition, one of the simplest is the *power method* [Strang, 2006], which is described in Algorithm 3.1. The method starts by randomly initialising a vector $\boldsymbol{v}$ and updating it by $\boldsymbol{Av}$, then, $\boldsymbol{v}$ is normalised and the process is repeated until $\boldsymbol{v}$ converges, i.e. the difference between the current $\boldsymbol{v}$ and the $\boldsymbol{v}$ from the previous iteration is smaller than a pre-defined threshold. Finally, matrix $\boldsymbol{A}$ is deflated using Equation 3.1, and the process is repeated until k eigenvalue/eigenvector pairs are computed.

Algorithm 3.1 Power method. $\boldsymbol{V}(:,h)$ denotes the column h of matrix $\boldsymbol{V}$

$k = \text{rank}(\boldsymbol{A})$ ▷ $\boldsymbol{A} \in \mathbb{R}^{n \times n}$
Initialise $\boldsymbol{V} = \boldsymbol{0}$ ▷ $\boldsymbol{V} \in \mathbb{R}^{n \times k}$
Initialise $\boldsymbol{\Lambda} = \boldsymbol{0}$ ▷ $\boldsymbol{\Lambda} \in \mathbb{R}^{k \times k}$
for $h = 1, \ldots, k$ **do**
    Initialise $\boldsymbol{v} \in \mathbb{R}^{n \times 1}$ as a random vector with $\|\boldsymbol{v}\|_2 = 1$
    repeat
        $\boldsymbol{v} \leftarrow \boldsymbol{Av}$
        $\boldsymbol{v} \leftarrow \boldsymbol{v} / \|\boldsymbol{v}\|_2$
    until convergence
    $\lambda = \boldsymbol{v}^\intercal \boldsymbol{Av}$ ▷ Compute eigenvalue
    $\boldsymbol{A} \leftarrow \boldsymbol{A} - \lambda \boldsymbol{v}\boldsymbol{v}^\intercal$ ▷ Deflate matrix $\boldsymbol{A}$ using (3.1)
    $\boldsymbol{V}(:,h) = \boldsymbol{v}$ and $\boldsymbol{\Lambda}(h,h) = \lambda$
end for
return $\boldsymbol{V}$, $\boldsymbol{\Lambda}$

The power method is used as the basis of several machine learning methods, since eigenvector computation is often an essential part of the procedure. Some of those methods include Singular Value Decomposition (SVD) (Section 3.1.1), Principal Component Analysis (PCA) (Section 3.1.2), Canonical Correlation Analysis (CCA) (Section 3.2), and Partial Least Squares (PLS) (Section 3.3).

3.1.1 Singular Value Decomposition (SVD)

When one wishes to decompose a rectangular matrix $\boldsymbol{M} \in \mathbb{R}^{n \times p}$, the eigen-decomposition (Equation 3.3) cannot be used, as this can only be applied to square matrices. However, there is a similar method called Singular Value Decomposition (SVD):

$$\boldsymbol{M} = \boldsymbol{UDV}^\intercal \tag{3.4}$$

where $\boldsymbol{U} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{V} \in \mathbb{R}^{p \times p}$ are orthonormal matrices (Section 3.1), and $\boldsymbol{D} \in \mathbb{R}^{n \times p}$ is a diagonal matrix whose entries $d_h \in \{1, \ldots, k\}$ are called the *singular values* of $\boldsymbol{M}$. The columns of $\boldsymbol{U}$ are eigenvectors of $\boldsymbol{M}\boldsymbol{M}^\intercal$, the columns of $\boldsymbol{V}$ are eigenvectors of $\boldsymbol{M}^\intercal\boldsymbol{M}$, and the singular values are the square roots of the non-zero eigenvalues of both $\boldsymbol{M}\boldsymbol{M}^\intercal$ and $\boldsymbol{M}^\intercal\boldsymbol{M}$ [Strang, 2006].

This decomposition can be performed using the power method. However, since matrix $\boldsymbol{M}$ is not square, Algorithm 3.1 has to be modified to Algorithm 3.2.

Algorithm 3.2 Singular Value Decomposition (SVD).

$k = \text{rank}(\boldsymbol{M})$ ▷ $\boldsymbol{M} \in \mathbb{R}^{n \times p}$
Initialise $\boldsymbol{U} = \boldsymbol{0}$ ▷ $\boldsymbol{U} \in \mathbb{R}^{n \times k}$
Initialise $\boldsymbol{V} = \boldsymbol{0}$ ▷ $\boldsymbol{V} \in \mathbb{R}^{p \times k}$
Initialise $\boldsymbol{D} = \boldsymbol{0}$ ▷ $\boldsymbol{D} \in \mathbb{R}^{k \times k}$
for $h = 1, \ldots, k$ **do**
    Initialise $\boldsymbol{u} \in \mathbb{R}^{n \times 1}$ as a random vector with $\|\boldsymbol{u}\|_2 = 1$
    Initialise $\boldsymbol{v} \in \mathbb{R}^{p \times 1}$ as a random vector with $\|\boldsymbol{v}\|_2 = 1$
    repeat
        $\boldsymbol{u} \leftarrow \boldsymbol{M}\boldsymbol{v}$
        $\boldsymbol{u} \leftarrow \boldsymbol{u}/\|\boldsymbol{u}\|_2$
        $\boldsymbol{v} \leftarrow \boldsymbol{M}^\intercal\boldsymbol{u}$
        $\boldsymbol{v} \leftarrow \boldsymbol{v}/\|\boldsymbol{v}\|_2$
    until convergence
    $d = \boldsymbol{u}^\intercal\boldsymbol{M}\boldsymbol{v}$ ▷ Compute singular value
    $\boldsymbol{M} \leftarrow \boldsymbol{M} - d\boldsymbol{u}\boldsymbol{v}^\intercal$ ▷ Deflate matrix $\boldsymbol{M}$
    $\boldsymbol{U}(:, h) = \boldsymbol{u}$, $\boldsymbol{D}(h, h) = d$, $\boldsymbol{V}(:, h) = \boldsymbol{v}$
end for
return $\boldsymbol{U}$, $\boldsymbol{D}$, $\boldsymbol{V}$

SVD is a widely used method for matrix decomposition, whose application spans many fields of research. However, the interest of SVD to the present work is its connection with Partial Least Squares (PLS), which shall be explained in Section 3.3.

3.1.2 Principal Component Analysis (PCA)

As mentioned in Section 3.1, the power method (Algorithm 3.1) is a very important procedure for several eigen-decomposition methods. In this section, one of the most popular eigen-decomposition methods will be described: Principal Component Analysis (PCA). This will serve not only as an example of a practical application of eigen-decomposition, but also to introduce some of the concepts necessary for the description of the main eigen-decomposition methods used in this thesis (Sections 3.2 to 3.4).

PCA is an unsupervised machine learning method, which allows one to explore the covariance of the features in a dataset. Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ be a data matrix containing n samples with $p > 1$ mean-centered features, which have some linear dependencies. The objective of PCA is to find a set of orthogonal principal components $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$, $k < p$ which capture the directions of maximum covariance in the data:

$$\underset{\boldsymbol{v}_h}{\text{maximise}}\ \boldsymbol{v}_h^\intercal \boldsymbol{C}_{xx} \boldsymbol{v}_h$$

$$\text{subject to}$$

$$\|\boldsymbol{v}_h\|_2 = 1 \,\forall\, h \in \{1, \ldots, k\} \quad \text{and} \quad \boldsymbol{v}_i \perp \boldsymbol{v}_j \,\forall\, i \neq j$$

where $\boldsymbol{C}_{xx} \in \mathbb{R}^{p \times p}$ is the covariance matrix $\boldsymbol{X}^\intercal \boldsymbol{X}$.

These components can be obtained by performing an eigen-decomposition (Equation 3.3) of matrix $\boldsymbol{C}_{xx}$. Figure 3.1 shows a simulated two dimensional dataset, and the principal components computed by PCA.

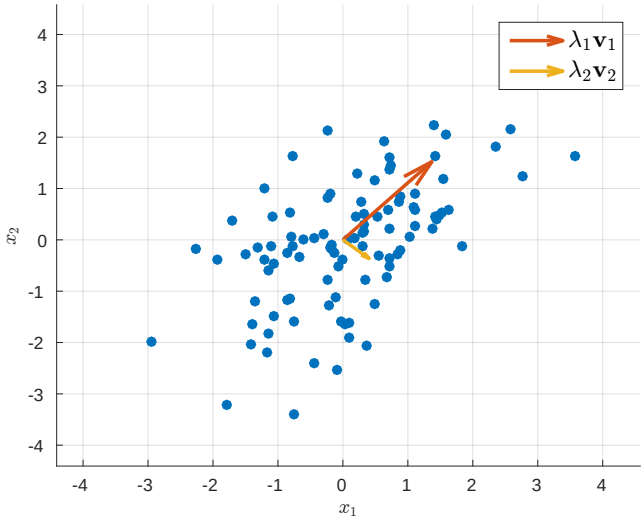


Figure 3.1: Simulated two dimensional dataset (blue dots), and its principal components multiplied by the corresponding eigenvalues $\{\lambda_1 \boldsymbol{v}_1, \lambda_2 \boldsymbol{v}_2\}$.

PCA allows one to encode the data in a lower dimensional subspace, subject to a certain amount of information loss, depending on how many components k are used. In other words, it allows one to compress the data. Thus, PCA is considered a dimensionality reduction technique, where each sample is represented by a smaller number of k dimensions. These dimensions are obtained by projecting the original dataset $\boldsymbol{X}$ onto a subset of k principal components.

A common application of PCA is in data denoising, which can be achieved by

performing PCA, and then projecting the data onto a subset of the top k principal components. By doing this, one removes the variance explained by the smaller eigenvalues, which is usually associated with noise [Shawe-Taylor and Cristianini, 2004]. However, one should keep in mind that PCA is an unsupervised learning method. If PCA is used as a step to reduce the dimensionality of the data before a supervised machine learning method is applied, one has to bear in mind that the components with the smallest eigenvalues might actually contain the signal that one wishes to use for the supervised learning task [Shawe-Taylor and Cristianini, 2004].

3.2 Canonical Correlation Analysis (CCA)

There are situations in which one is not interested in how the data vary in $\boldsymbol{X}$, but in the relationships between data coming from two sources $\boldsymbol{X}$ and $\boldsymbol{Y}$. One of the ways to explore these relationships is by using Canonical Correlation Analysis (CCA) [Hotelling, 1936].

Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ and $\boldsymbol{Y} \in \mathbb{R}^{n \times q}$ be two matrices with data coming from the same n subjects. $\boldsymbol{X}$ and $\boldsymbol{Y}$ are called the *views* of a paired dataset, where the information expressed in each one is originated from a latent process, i.e. the data in $\boldsymbol{X}$ and $\boldsymbol{Y}$ are generated by an underlying process which is not directly observed. For example, $\boldsymbol{X}$ may contain data coming from MRI scans, while $\boldsymbol{Y}$ may contain data coming from clinical exams, both from the same subjects.

CCA tries to find a weight vector pair $\boldsymbol{u} \in \mathbb{R}^{p \times 1}$ and $\boldsymbol{v} \in \mathbb{R}^{q \times 1}$, such that the projections of the data matrices onto the weight vector pair are maximally correlated. In other words, let the projections be denoted by $\boldsymbol{\xi} = \boldsymbol{X}\boldsymbol{u}$ and $\boldsymbol{\omega} = \boldsymbol{Y}\boldsymbol{v}$, CCA computes $\boldsymbol{u}$ and $\boldsymbol{v}$, such that ρ is maximised:

$$\rho = \frac{\mathrm{Cov}(\boldsymbol{\xi}, \boldsymbol{\omega})}{\mathrm{Var}(\boldsymbol{\xi})\mathrm{Var}(\boldsymbol{\omega})} \tag{3.5}$$

Equation 3.5 can be re-written as:

$$\rho = \frac{\boldsymbol{u}^\intercal \boldsymbol{C}_{xy} \boldsymbol{v}}{\sqrt{\boldsymbol{u}^\intercal \boldsymbol{C}_{xx} \boldsymbol{u}} \sqrt{\boldsymbol{v}^\intercal \boldsymbol{C}_{yy} \boldsymbol{v}}} \tag{3.6}$$

where $\boldsymbol{C}_{xy} = \boldsymbol{X}^\intercal \boldsymbol{Y}$ (covariance matrix of $\boldsymbol{X}$ and $\boldsymbol{Y}$), $\boldsymbol{C}_{xx} = \boldsymbol{X}^\intercal \boldsymbol{X}$ (covariance matrix of $\boldsymbol{X}$), and $\boldsymbol{C}_{yy} = \boldsymbol{Y}^\intercal \boldsymbol{Y}$ (covariance matrix of $\boldsymbol{Y}$).

The solution to this problem is invariant to the rescaling of $\boldsymbol{u}$ and $\boldsymbol{v}$, thus it can be obtained by solving the following optimisation problem [Shawe-Taylor and Cristianini, 2004]:

$$\begin{aligned} &\underset{\boldsymbol{u},\boldsymbol{v}}{\text{maximise}} \;\; \boldsymbol{u}^\intercal \boldsymbol{C}_{xy} \boldsymbol{v} \\ &\text{subject to } \; \boldsymbol{u}^\intercal \boldsymbol{C}_{xx} \boldsymbol{u} = 1 \text{ and } \boldsymbol{v}^\intercal \boldsymbol{C}_{yy} \boldsymbol{v} = 1 \end{aligned} \tag{3.7}$$

There are several ways in which classical CCA (Equation 3.7) can be solved, including: solving a standard eigenvalue problem [Hotelling, 1936], solving a generalised eigenvalue problem [Bach and Jordan, 2002], and using SVD [Healy, 1957]. However, a detailed description of these algorithms is beyond the scope of this thesis. For a more detailed review, please refer to the paper by Uurtio et al. [(accepted)].

3.2.1 Regularised Canonical Correlation Analysis

As mentioned in Section 2.1.1, it is often necessary to add some regularisation to the model, in order to limit its complexity. One of the most common ways to do this is by imposing penalties on the norm of the weight vectors, this can also be done for CCA, by re-writing Equation 3.6 as:

$$\rho = \frac{\boldsymbol{u}^\intercal \boldsymbol{C}_{xy} \boldsymbol{v}}{\sqrt{\left((1-\tau_x)\, \boldsymbol{u}^\intercal \boldsymbol{C}_{xx} \boldsymbol{u} + \tau_x \left\|\boldsymbol{u}\right\|_2^2\right) \left((1-\tau_y)\, \boldsymbol{v}^\intercal \boldsymbol{C}_{yy} \boldsymbol{v} + \tau_y \left\|\boldsymbol{v}\right\|_2^2\right)}} \tag{3.8}$$

where τ_x and τ_y are regularisation hyper-parameters.

The regularisation hyper-parameters can be varied between 0 and 1. If $\{\tau_x, \tau_y\} = \{0, 0\}$, the optimisation problem is equal to non-regularised CCA. However, if $\{\tau_x, \tau_y\} = \{1, 1\}$, then the optimisation problem maximises the covariance between the projections, instead of the correlation. This is known as a Partial Least Squares (PLS) problem, and will be addressed in Section 3.3. Thus, the hyper-parameters τ_x and τ_y allow for a smooth interpolation between CCA and PLS [Shawe-Taylor and Cristianini, 2004].

Note that there are other strategies to regularise CCA, including strategies using sparsity constraints [Hardoon and Shawe-Taylor, 2011, Chi et al., 2013, Avants et al., 2014]. These ideas will be further explored in Chapters 7 and 8.

3.2.2 Kernel Canonical Correlation Analysis (KCCA)

As previously mentioned in Section 2.1.2, sometimes the patterns of interest in the data are not linear, which means that linear methods may not be able to properly detect them. Both CCA (Equation 3.6) and regularised CCA (Equation 3.8) are linear methods, which means that they may not be able to model the relationships between $\boldsymbol{X}$ and $\boldsymbol{Y}$, if these happen to be non-linear. One of the ways to address this issue, is by adapting CCA into Kernel CCA (KCCA) using non-linear kernels [Lai and Fyfe, 2000].

The first step is to change from the original primal formulation into a dual formulation. Just like in Section 2.1.2.2, this consists in expressing the weight vectors as a function of the input data:

$$\boldsymbol{u} = \boldsymbol{X}^\intercal \boldsymbol{\alpha}_x \;\text{ and }\; \boldsymbol{v} = \boldsymbol{Y}^\intercal \boldsymbol{\alpha}_y$$

The optimisation problem expressed in Equation 3.7, can then be re-written as:

$$\begin{gathered} \underset{\boldsymbol{\alpha}_x, \boldsymbol{\alpha}_y}{\text{maximise}} \;\; \boldsymbol{\alpha}_x^\intercal \boldsymbol{X}\boldsymbol{X}^\intercal \boldsymbol{Y}\boldsymbol{Y}^\intercal \boldsymbol{\alpha}_y \\ \text{subject to} \\ \alpha_x^\intercal \boldsymbol{X}\boldsymbol{X}^\intercal \boldsymbol{X}\boldsymbol{X}^\intercal \alpha_x = 1 \quad \text{and} \quad \alpha_y^\intercal \boldsymbol{Y}\boldsymbol{Y}^\intercal \boldsymbol{Y}\boldsymbol{Y}^\intercal \alpha_y = 1 \end{gathered} \tag{3.9}$$

By using Equation 2.10, the data matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$ in Equation 3.9 can be substituted by kernels:

$$\begin{gathered} \underset{\boldsymbol{\alpha}_x, \boldsymbol{\alpha}_y}{\text{maximise}} \;\; \boldsymbol{\alpha}_x^\intercal \boldsymbol{K}_x \boldsymbol{K}_y \boldsymbol{\alpha}_y \\ \text{subject to} \\ \alpha_x^\intercal \boldsymbol{K}_x^2 \alpha_x = 1 \quad \text{and} \quad \alpha_y^\intercal \boldsymbol{K}_y^2 \alpha_y = 1 \end{gathered} \tag{3.10}$$

An analogous KCCA optimisation problem can be written for the regularised version of CCA expressed in Equation 3.8:

$$\begin{gathered} \underset{\boldsymbol{\alpha}_x, \boldsymbol{\alpha}_y}{\text{maximise}} \;\; \boldsymbol{\alpha}_x^\intercal \boldsymbol{K}_x \boldsymbol{K}_y \boldsymbol{\alpha}_y \\ \text{subject to} \\ (1-\tau_x)\alpha_x^\intercal \boldsymbol{K}_x^2 \alpha_x + \tau_x \alpha_x^\intercal \boldsymbol{K}_x \alpha_x = 1 \quad \text{and} \quad (1-\tau_y)\alpha_y^\intercal \boldsymbol{K}_y^2 \alpha_y + \tau_y \alpha_y^\intercal \boldsymbol{K}_y \alpha_y = 1 \end{gathered} \tag{3.11}$$

Just as in the case of CCA (Section 3.2), there are several ways in which these KCCA problems (Equation 3.10 and 3.11) can be solved, including some strategies using matrix decomposition methods, such as the Cholesky decomposition [Shawe-Taylor and Cristianini, 2004]. However, as previously mentioned, a detailed review of these methods is beyond the scope of this thesis.

3.3 Partial Least Squares (PLS)

When $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ or $\boldsymbol{Y} \in \mathbb{R}^{n \times q}$ are ill-conditioned, i.e. $p > n$ or $q > n$, CCA may encounter numerical issues. Indeed, some CCA algorithms (e.g. using SVD) rely on the computation of $(\boldsymbol{X}^\intercal \boldsymbol{X})^{-1/2}$ and $(\boldsymbol{Y}^\intercal \boldsymbol{Y})^{-1/2}$, which cannot be properly computed if $\boldsymbol{X}$ or $\boldsymbol{Y}$ are ill-conditioned, as the necessary operations involve matrix inversions. One of the ways to address this issue is by regularising CCA (Section 3.2.1), or by using Partial Least Squares (PLS). The latter could be seen as an extreme form of regularised CCA, i.e. solving Equation 3.8 with $\{\tau_x, \tau_y\} = \{1, 1\}$.

PLS is not a single method, but a class of different methods which, unlike CCA, aim to maximise the covariance between the projections, instead of the correlation. In this section, an overview of the general concepts of PLS will be made, along with some of its classical variants.

Let $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ and $\boldsymbol{Y} \in \mathbb{R}^{n \times q}$ be the two mean-centered and normalised matrices (each feature has zero mean and standard deviation equal to 1) of a paired dataset, i.e. each row of $\boldsymbol{X}$ and $\boldsymbol{Y}$ contains information about a sample coming from two sources of information (e.g. neuroimaging and clinical scores). The first PLS approaches were introduced by Wold [1985] and, essentially, try to find a set of projections $\boldsymbol{\xi} = \boldsymbol{X}\boldsymbol{u}$ and $\boldsymbol{\omega} = \boldsymbol{Y}\boldsymbol{v}$ which have a maximum covariance [Wegelin, 2000]:

$$\text{maximise } \operatorname{Cov}(\boldsymbol{\xi}, \boldsymbol{\omega}) = \underset{\|\boldsymbol{u}\|_2 = 1, \|\boldsymbol{v}\|_2 = 1}{\text{maximise}} \operatorname{Cov}(\boldsymbol{X}\boldsymbol{u}, \boldsymbol{Y}\boldsymbol{v}) \tag{3.12}$$

In order to compute the solution to the problem expressed in Equation 3.12, one can use the Non-linear Iterative Least Squares (NIPALS) [Wold, 1974] (Algorithm 3.3).

After a pair of projection vectors $\{\boldsymbol{\xi}, \boldsymbol{\omega}\}$ is computed using Algorithm 3.3, this can be used to deflate the data matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$ in order to find subsequent weight vector pairs. The main difference between the classical types of PLS is in

Algorithm 3.3 NIPALS algorithm [Rosipal and Krämer, 2006].

1: Initialise $\boldsymbol{\omega}$ randomly.
2: **repeat**
3: $\quad \boldsymbol{u} \leftarrow \boldsymbol{X}^\intercal \frac{\boldsymbol{\omega}}{\boldsymbol{\omega}^\intercal \boldsymbol{\omega}}$
4: $\quad \boldsymbol{u} \leftarrow \boldsymbol{u} / \|\boldsymbol{u}\|_2$
5: $\quad \boldsymbol{\xi} \leftarrow \boldsymbol{X}\boldsymbol{u}$
6: $\quad \boldsymbol{v} \leftarrow \boldsymbol{Y}^\intercal \frac{\boldsymbol{\xi}}{\boldsymbol{\xi}^\intercal \boldsymbol{\xi}}$
7: $\quad \boldsymbol{v} \leftarrow \boldsymbol{v} / \|\boldsymbol{v}\|_2$
8: $\quad \boldsymbol{\omega} \leftarrow \boldsymbol{Y}\boldsymbol{v}$
9: **until** convergence

the deflation procedure, which means that these PLS variants will provide the same solution for the first weight vector pair, but subsequent weight vector pairs will be different. Some of the most common PLS variants are briefly described below, for a more detailed review of these methods, please refer to the papers by Wegelin [2000] and Rosipal and Krämer [2006].

PLS Mode-A

The data matrices are deflated using the *loadings*, which are computed as follows [Rosipal and Krämer, 2006]:

$$\boldsymbol{\gamma}_h = \boldsymbol{X}_h^\intercal \frac{\boldsymbol{\xi}_h}{\boldsymbol{\xi}_h^\intercal \boldsymbol{\xi}_h} \quad \text{and} \quad \boldsymbol{\delta}_h = \boldsymbol{Y}_h^\intercal \frac{\boldsymbol{\omega}_h}{\boldsymbol{\omega}_h^\intercal \boldsymbol{\omega}_h}$$

The matrix deflations are then performed as [Rosipal and Krämer, 2006]:

$$\boldsymbol{X}_{h+1} = \boldsymbol{X}_h - \boldsymbol{\xi}_h \boldsymbol{\gamma}_h^\intercal \quad \text{and} \quad \boldsymbol{Y}_{h+1} = \boldsymbol{Y}_h - \boldsymbol{\omega}_h \boldsymbol{\delta}_h^\intercal \tag{3.13}$$

Since the deflation of both data matrices is done in the same way, i.e. each data matrix is deflated using its corresponding projections and loadings, PLS Mode-A is considered a symmetric variant of PLS.

PLS1 and PLS2

PLS2 is a method used to predict the variables in one view from the other, i.e. it can be seen as a regression problem in which the projections of $\boldsymbol{X}$ ($\{\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_k\}$) are used to predict $\boldsymbol{Y}$. The deflation is performed in the following way [Rosipal and Krämer, 2006]:

$$\boldsymbol{X}_{h+1} = \boldsymbol{X}_h - \boldsymbol{\xi}_h \boldsymbol{\gamma}_h^\intercal \quad \text{and} \quad \boldsymbol{Y}_{h+1} = \boldsymbol{Y}_h - \frac{\boldsymbol{\xi}_h \boldsymbol{\xi}_h^\intercal \boldsymbol{Y}_h}{\boldsymbol{\xi}_h^\intercal \boldsymbol{\xi}_h} = \boldsymbol{Y}_h - \boldsymbol{\xi}_h \boldsymbol{v}_h$$

PLS1 is a special case of this approach, in which $\boldsymbol{Y}$ corresponds to a single column. Unlike PLS Mode-A, PLS1 and PLS2 are asymmetric versions of PLS, due to the fact that $\boldsymbol{X}$ and $\boldsymbol{Y}$ are deflated differently.

PLS-SVD

PLS-SVD is a symmetric variant of PLS. Unlike PLS Mode-A, it does not perform matrix deflation based on $\boldsymbol{\xi}$, $\boldsymbol{\omega}$, $\boldsymbol{\gamma}$ and $\boldsymbol{\delta}$. Instead, the method computes $\boldsymbol{u}$ and $\boldsymbol{v}$ by applying SVD (Section 3.1.1) to the covariance matrix $\boldsymbol{X}^\intercal\boldsymbol{Y}$ and then uses these weight vectors to directly deflate the covariance matrix [Wegelin, 2000]. In other words, the vectors that solve the optimisation problem described in Equation 3.12, are the principal components of the covariance matrix. The whole procedure (including the deflation) is equivalent to the one described in Algorithm 3.2, however, matrix $\boldsymbol{M}$ is substituted by $\boldsymbol{C} = \boldsymbol{X}^\intercal\boldsymbol{Y}$. Although it is possible to use this variant for prediction, it is usually used in the context of modeling [Wegelin, 2000].

One of the main differences between PLS-SVD and PLS Mode-A, is in the orthogonality of the projections. Let $\boldsymbol{\Xi}$ be a matrix where each column corresponds to a projection of $\boldsymbol{X}$ ($\{\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_k\}$), and $\boldsymbol{\Omega}$ be a matrix where each column corresponds to a projection of $\boldsymbol{Y}$ ($\{\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_k\}$). One of the consequences of PLS-SVD is that, usually, neither $\boldsymbol{\Xi}^\intercal\boldsymbol{\Xi}$ nor $\boldsymbol{\Omega}^\intercal\boldsymbol{\Omega}$ are diagonal, but $\boldsymbol{\Xi}^\intercal\boldsymbol{\Omega}$ is. The opposite is true for PLS Mode-A, i.e. $\boldsymbol{\Xi}^\intercal\boldsymbol{\Xi}$ and $\boldsymbol{\Omega}^\intercal\boldsymbol{\Omega}$ are diagonal, but $\boldsymbol{\Xi}^\intercal\boldsymbol{\Omega}$ is not [Wegelin, 2000].

3.4 Sparse Partial Least Squares (SPLS)

In high-dimensional settings, such as when using neuroimaging data, it is often desirable to remove non-informative features from the model. This will usually improve both the model's performance, and the interpretability of the solution, since the latter will include only a subset of the features. Sparse Partial Least Squares (SPLS) is an extension of PLS which allows the computation of sparse solutions, i.e. sparse weight vectors pairs. Several SPLS methods have been proposed around the same time with different names [Lê Cao et al., 2008, Parkhomenko et al., 2009, Witten et al., 2009], but with very similar properties. This section will start by focusing on one of the most popular ones [Witten et al., 2009], and explain the differences between this method and two similar methods from around the same time. It will then described some of the methods proposed in the literature for SPLS hyper-parameter optimisation (Section 3.4.1), and for statistical evaluation

(Section 3.4.2).

Witten et al. [2009] derived a sparse version of PLS by starting with a CCA optimisation problem and writing it as follows:

$$\underset{\boldsymbol{u},\boldsymbol{v}}{\text{maximise}} \;\; \boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{Y} \boldsymbol{v} \;\; \text{subject to} \;\; \boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{X} \boldsymbol{u} \leqslant 1, \, \boldsymbol{v}^\intercal \boldsymbol{Y}^\intercal \boldsymbol{Y} \boldsymbol{v} \leqslant 1$$

Then, a sparse version of CCA was obtained by adding two l_1-norm constraints to the CCA optimisation problem:

$$\begin{gathered} \underset{\boldsymbol{u},\boldsymbol{v}}{\text{maximise}} \;\; \boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{Y} \boldsymbol{v} \\ \text{subject to} \\ \boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{X} \boldsymbol{u} \leqslant 1, \;\; \boldsymbol{v}^\intercal \boldsymbol{Y}^\intercal \boldsymbol{Y} \boldsymbol{v} \leqslant 1, \;\; ||\boldsymbol{u}||_1 \leqslant c_u, \;\; ||\boldsymbol{v}||_1 \leqslant c_v \end{gathered}$$

Due to the high dimensionality of the data, the matrices $\boldsymbol{X}^\intercal \boldsymbol{X}$ and $\boldsymbol{Y}^\intercal \boldsymbol{Y}$ were substituted by identity matrices, which lead to the final optimisation problem [Witten et al., 2009]:

$$\begin{gathered} \underset{\boldsymbol{u},\boldsymbol{v}}{\text{maximise}} \;\; \boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{Y} \boldsymbol{v} \\ \text{subject to} \\ ||\boldsymbol{u}||_2^2 \leqslant 1, \;\; ||\boldsymbol{v}||_2^2 \leqslant 1, \;\; ||\boldsymbol{u}||_1 \leqslant c_u, \;\; ||\boldsymbol{v}||_1 \leqslant c_v \end{gathered} \tag{3.14}$$

where c_u and c_v are the regularisation hyper-parameters that control the l_1-norm ($\|\cdot\|_1$) constraints of $\boldsymbol{u}$ and $\boldsymbol{v}$, respectively. The l_1-norm constraints impose sparsity, which means that the lower the values of c_u and c_v are, the stronger the l_1 constraint on the corresponding view is and, consequently, the fewer features are included in the model. In order for both l_1-norm and l_2-norm constraints to be active, the values of the hyper-parameters must be $1 \leqslant c_u \leqslant \sqrt{p}$ and $1 \leqslant c_v \leqslant \sqrt{q}$ [Witten et al., 2009]. Although the l_2-norm constraints are expressed as inequalities, in practice the l_2-norms of both $\boldsymbol{u}$ and $\boldsymbol{v}$ will be set to 1 (Algorithm 3.4).

This method is referred to as "diagonal penalised CCA" in the original paper by Witten et al. [2009]. However, what is being maximised is no longer the correlation between the projections, but the covariance, which makes this problem equivalent to SPLS.

The weight vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ have the same length as the number of features in

the corresponding view, i.e. $\boldsymbol{u} \in \mathbb{R}^{p\times 1}$ and $\boldsymbol{v} \in \mathbb{R}^{q\times 1}$. They represent the latent space found by SPLS, capturing multivariate associative effects between the two views. Since sparsity constraints are applied to the model, several entries of $\boldsymbol{u}$ and $\boldsymbol{v}$ will be equal to zero. By looking at the paired vectors, one can identify the features in each view related with each associative effect.

The problem expressed in Equation 3.14 is solved by using Algorithm 3.4, which allows one to obtain k pairs of sparse weight vector pairs.

Algorithm 3.4 SPLS algorithm [Witten et al., 2009].

1: Initialise $\boldsymbol{U} = \boldsymbol{0}$ ▷ $\boldsymbol{U} \in \mathbb{R}^{n\times k}$
2: Initialise $\boldsymbol{V} = \boldsymbol{0}$ ▷ $\boldsymbol{V} \in \mathbb{R}^{p\times k}$
3: Initialise $\boldsymbol{D} = \boldsymbol{0}$ ▷ $\boldsymbol{D} \in \mathbb{R}^{k\times k}$
4: Let $\boldsymbol{C} \leftarrow \boldsymbol{X}^\intercal \boldsymbol{Y}$
5: **for** $h = 1, \ldots, k$ **do**
6: Initialise $\boldsymbol{v}$ to have $\|\boldsymbol{v}\|_2 = 1$
7: **repeat**
8: $\boldsymbol{u} \leftarrow \boldsymbol{C}\boldsymbol{v}$
9: $\boldsymbol{u} \leftarrow \frac{\mathrm{S}(\boldsymbol{u},\gamma_u)}{\|\mathrm{S}(\boldsymbol{u},\gamma_u)\|_2}$, where $\gamma_u = 0$ if this results in $\|\boldsymbol{u}\|_1 \leqslant c_u$; otherwise, γ_u is set to be a positive constant such that $\|\boldsymbol{u}\|_1 = c_u$.
10: $\boldsymbol{v} \leftarrow \boldsymbol{C}^\intercal \boldsymbol{u}$
11: $\boldsymbol{v} \leftarrow \frac{\mathrm{S}(\boldsymbol{v},\gamma_v)}{\|\mathrm{S}(\boldsymbol{v},\gamma_v)\|_2}$, where $\gamma_v = 0$ if this results in $\|\boldsymbol{v}\|_1 \leqslant c_v$; otherwise, γ_v is set to be a positive constant such that $\|\boldsymbol{v}\|_1 = c_v$.
12: **until** convergence
13: $d = \boldsymbol{u}^\intercal \boldsymbol{C}\boldsymbol{v}$
14: $\boldsymbol{C} \leftarrow \boldsymbol{C} - d\boldsymbol{u}\boldsymbol{v}^\intercal$ ▷ Deflate matrix $\boldsymbol{C}$
15: $\boldsymbol{U}(:,h) = \boldsymbol{u}$, $\boldsymbol{D}(h,h) = d$, $\boldsymbol{V}(:,h) = \boldsymbol{v}$
16: **end for**
17: **return** $\boldsymbol{U}$, $\boldsymbol{D}$, $\boldsymbol{V}$

The operator $S(\cdot,\cdot)$ (steps 9 and 11) is the soft-thresholding operator, defined as $S(a,\gamma) = \mathrm{sgn}(a)(|a| - \gamma)_+$, where $\gamma > 0$ is a constant and x_+ is equal to x if $x > 0$ and $x = 0$ if $x \leqslant 0$ [Witten et al., 2009]. The initialisation of $\boldsymbol{v}$ in step 6 can be done in several ways [Witten et al., 2009, Parkhomenko et al., 2009, Waaijenborg et al., 2008], in this thesis, it was done by taking vector $\boldsymbol{v}$ from the first weight vector pair given by the SVD of $\boldsymbol{C}$ [Witten et al., 2009]. γ_u and γ_v have to be set so that the l_1-norm constraints are obeyed. This is done by iteratively searching for γ_u and γ_v, such that $\|\boldsymbol{u}\|_1 \approx c_u$ and $\|\boldsymbol{v}\|_1 \approx c_v$.

Note that Algorithm 3.4 has some similarities with SVD using the power method (Algorithm 3.2), the key difference between the two algorithms is the addition of the

soft-thresholding operators $S(\cdot,\cdot)$, which impose sparsity on the solutions. Indeed, the power method has been used to solve PLS-SVD (Section 3.3), which means that SPLS can be seen as a sparse version of PLS-SVD.

In other SPLS algorithms, such as the ones by Lê Cao et al. [2008] and Parkhomenko et al. [2009], the sparsity is set by adjusting γ_u and γ_v instead of the l_1-norm constraints (c_u and c_v). This will make the algorithms faster, since γ_u and γ_v do not have to be searched iteratively. However, by setting γ_u and γ_v directly, there are situations in which these values might be too high and all the entries of $\boldsymbol{u}$ or $\boldsymbol{v}$ will be set to zero, i.e. no variables are included in the model. The exact values of γ_u and γ_v for which this happens is dataset depended. On the other hand, by using c_u or c_v to set $\|\cdot\|_1 = 1$, there is a guarantee that at least one entry of the corresponding weight vector will be different than zero (i.e. at least one variable is included), making the range of the regularisation hyper-parameters easier to define.

Another difference between the algorithm proposed by Witten et al. [2009] and the one proposed by Lê Cao et al. [2008], is the deflation step. Witten et al. [2009] uses the Hotelling's deflation (i.e. the symmetric deflation step used in SVD and PLS-SVD), whereas Lê Cao et al. [2008] uses an asymmetric deflation step, in order to use SPLS in a regression formulation (Section 3.3).

Several other variants of SPLS have been proposed, including supervised versions and formulations with more than two views [Witten and Tibshirani, 2009]. However, their description is outside the scope of this thesis.

Although Algorithm 3.4 describes how to solve the SPLS optimisation problem, it can be adapted to solve a Sparse Canonical Correlation Analysis (SCCA) problem. Parkhomenko et al. [2009] proposed a similar algorithm in which steps 9 and 11 are substituted by simple soft-thresholding operators (like Lê Cao et al. [2008]), but $\boldsymbol{C}$ was computed as:

$$\boldsymbol{C} \leftarrow (\boldsymbol{X}^\intercal \boldsymbol{X})^{-1/2}(\boldsymbol{X}^\intercal \boldsymbol{Y})(\boldsymbol{Y}^\intercal \boldsymbol{Y})^{-1/2} \tag{3.15}$$

Note that if $\boldsymbol{X}^\intercal \boldsymbol{X}$ and $\boldsymbol{Y}^\intercal \boldsymbol{Y}$ are substituted by identity matrices in Equation 3.15, then, SPLS is recovered.

3.4.1 Hyper-parameter optimisation

There are two sparsity hyper-parameters which have to be set when using SPLS (c_u and c_v). It has been argued that, in some cases, these can be fixed *a priori* [Witten

et al., 2009, Lê Cao et al., 2008, 2009], however, this does require a fair amount of assumptions about the data, i.e. one assumes to know approximately how many informative features exist in each view of the dataset. By letting SPLS select the hyper-parameters automatically, one takes a more data-driven approach to the problem, by letting the model choose the adequate amount of sparsity necessary to model the multivariate associations in the data. This section aims to give an overview of some of the approaches proposed in the literature for hyper-parameter optimisation in SPLS.

Waaijenborg et al. [2008] selected the hyper-parameters using a CV framework. For each hyper-parameter combination, the authors performed a k-fold CV where the average difference between the train set correlation and test set correlation was computed:

$$\Delta\rho = \frac{1}{K}\sum_{k=1}^{K}\left|\left|\text{Crr}\left(\boldsymbol{X}_{(-k)}\boldsymbol{u}_{(-k)}, \boldsymbol{Y}_{(-k)}\boldsymbol{v}_{(-k)}\right)\right| - \left|\text{Crr}\left(\boldsymbol{X}_{k}\boldsymbol{u}_{(-k)}, \boldsymbol{Y}_{k}\boldsymbol{v}_{(-k)}\right)\right|\right|$$

where, for fold k, $\boldsymbol{X}_{(-k)}$ and $\boldsymbol{Y}_{(-k)}$ are the train data matrices (subset k has been removed); $\boldsymbol{X}_k$ and $\boldsymbol{Y}_k$ are the test data matrices; $\boldsymbol{u}_{(-k)}$ and $\boldsymbol{v}_{(-k)}$ are the weight vectors computed using the train data; and $\text{Crr}(\boldsymbol{Xu}, \boldsymbol{Yv})$ denotes the correlation between the projections, as described in Equation 3.6. Thus, the selected hyper-parameter combination will be the one in which the correlations on the train data and test data agree more closely, i.e. $\Delta\rho$ is small. This criteria was also used more recently by Lin et al. [2014].

Parkhomenko et al. [2009] argued that the criteria proposed by Waaijenborg et al. [2008] was not well-motivated, since it penalises cases in which the correlation on the test set is higher than on the train set [Parkhomenko et al., 2009]. Moreover, note that there might exist some cases in which both the correlation on the train and test data are close to zero, which will lead to a small $\Delta\rho$, even thought the model does not fit the data well. Parkhomenko et al. [2009] adopted the average absolute correlation on the test set as the criteria:

$$\bar{\rho} = \frac{1}{K}\sum_{k=1}^{K}\left|\text{Crr}\left(\boldsymbol{X}_{k}\boldsymbol{u}_{(-k)}, \boldsymbol{Y}_{k}\boldsymbol{v}_{(-k)}\right)\right|$$

Witten et al. [2009] proposed a hyper-parameter selection procedure based on permutations. In this case, for each hyper-parameter combination $\{c_u, c_v\}$, matrix $\boldsymbol{X}$ was fixed, and matrix $\boldsymbol{Y}$ was permuted B times. The model was estimated each time and the correlation of the projections was computed:

$$\rho_b = \text{Crr}(\boldsymbol{X}\boldsymbol{u}_b, \boldsymbol{Y}_b\boldsymbol{v}_b)$$

where $\boldsymbol{Y}_b$ denotes the permuted data matrix; $\boldsymbol{u}_b$ and $\boldsymbol{v}_b$ denote the weight vectors obtained using the data matrices $\boldsymbol{X}$ and $\boldsymbol{Y}_b$, for permutation b. The p-value was then given by:

$$p = \frac{1}{B}\sum_{b=1}^{B} 1_{\rho_b \geqslant \rho}$$

where $\rho = \text{Crr}(\boldsymbol{X}\boldsymbol{u}, \boldsymbol{Y}\boldsymbol{v})$. The hyper-parameter combination $\{c_u, c_v\}$ was chosen based on the smallest p-value [Witten et al., 2009]. Alternatively, the standard deviation (σ) of ρ_b can be taken into account [Witten and Tibshirani, 2009]. In such cases, the hyper-parameter combination was chosen so that $(\rho - \frac{1}{N}\sum_b \rho_b)/\sigma$ is the largest. Note that these approaches do not use a CV framework (as in the papers by Waaijenborg et al. [2008] and Parkhomenko et al. [2009]), the data matrices are not split into train and test sets.

In SPLS regression frameworks, such as the one proposed by Lê Cao et al. [2008], the hyper-parameters were set using a CV framework, with the prediction error as the criteria for selection. However, this criteria was only used for simulated data, in the real dataset, the authors arbitrary set the number of non-zero weights [Lê Cao et al., 2008].

Despite the emphasis of this thesis on hyper-parameter selection, there have been studies in the literature which made use of other approaches, such as stability selection. This approach was originally introduced by Meinshausen and Bühlmann [2010] for the LASSO, however, other applications and variations have been proposed [Rondina et al., 2014]. The method consists in fixing the sparsity hyper-parameters, and subsampling the data set (e.g. sampling random rows of the data matrix $\boldsymbol{X}$), fitting a model for each subsample. Then, the variables are selected based on how often they were included in the models trained with the different subsamples. The final model is trained using only the features whose frequency of selection was above a

certain threshold [Rondina et al., 2014]. Although stability based approaches have been proposed for SPLS [Lê Cao et al., 2011, Labus et al., 2015], they are very computationally expensive. Further comments on this computational cost are made in the discussion of Chapter 6.

3.4.2 Statistical evaluation

As mentioned in Section 2.2.3, it is often desirable to assess the statistical significance of the results given by machine learning approaches. One of the most popular ways to perform this assessment is by using permutation tests. However, there is no unified procedure to perform these tests in the SPLS literature. In this section, a brief overview of some of the previously proposed approaches will be made.

Waaijenborg et al. [2008] proposed a permutation framework using the projections of the test data, without re-training the model. For each CV fold k, and for B permutations, $\boldsymbol{\xi}_k = \boldsymbol{X}_k \boldsymbol{u}_{(-k)}$ were permuted while $\boldsymbol{\omega}_k = \boldsymbol{Y}_k \boldsymbol{v}_{(-k)}$ were fixed, then, the differences between the correlations of the train and the permuted test sets were computed:

$$\Delta\rho_{k,b} = \mathrm{Crr}(\boldsymbol{\xi}_{(-k)}, \boldsymbol{\omega}_{(-k)}) - \mathrm{Crr}(\boldsymbol{\xi}_{k,b}, \boldsymbol{\omega}_k), \ \forall k \in \{1, \ldots, K\}, \forall b \in \{1, \ldots, B\},$$

where the train data projections are denoted as $\boldsymbol{\xi}_{(-k)} = \boldsymbol{X}_{(-k)} \boldsymbol{u}_{(-k)}$ and $\boldsymbol{\omega}_{(-k)} = \boldsymbol{Y}_{(-k)} \boldsymbol{v}_{(-k)}$. These were then compared with the same procedure performed on the non-permuted test sets:

$$\Delta\rho_k = \mathrm{Crr}(\boldsymbol{\xi}_{(-k)}, \boldsymbol{\omega}_{(-k)}) - \mathrm{Crr}(\boldsymbol{\xi}_k, \boldsymbol{\omega}_k), \ \forall k \in \{1, \ldots, K\}$$

Note that $\mathrm{Crr}(\boldsymbol{\xi}_{k,b}, \boldsymbol{\omega}_k)$ should be close to zero, thus, if there is no statistically significant association, then $\Delta\rho_{k,b} \approx \Delta\rho_k$.

In the papers by Witten et al., the statistical significance of the weight vector pairs was given by the same procedure which was used to select the SPLS hyper-parameters (Section 3.4.1) since this was based on the computation of p-values [Witten et al., 2009, Witten and Tibshirani, 2009].

Le Floch et al. [2012] performed a permutation test using a CV for each hyper-parameter combination, i.e. for a fixed hyper-parameter combination, the data were subjected to B permutations of $\boldsymbol{Y}$ and, for each permutation b, a 10-fold CV was

performed and the average correlation $\bar{\rho}_b$ was computed:

$$\bar{\rho}_b = \frac{1}{K}\sum_{k=1}^{K}\left|\mathrm{Crr}\left(\boldsymbol{X}_k\boldsymbol{u}_{(-k),b}, \boldsymbol{Y}_{k,b}\boldsymbol{v}_{(-k),b}\right)\right|$$

The p-value was then obtained by comparing these correlations with $\bar{\rho}$ computed with the non-permuted data. This procedure was repeated for each hyper-parameter combination, i.e. each hyper-parameter combination had an associated p-value. The p-values were then corrected for multiple comparisons. Note that the authors did not performed a nested-CV (Section 2.2), this was due to the very large computational time associated with nested-CV [Le Floch et al., 2012].

Lin et al. [2014] tested the statistical significance of their results by obtaining $\boldsymbol{u}$ and $\boldsymbol{v}$ based on the optimal hyper-parameters and calculating the correlation between the projections, i.e. $\mathrm{Crr}(\boldsymbol{X}\boldsymbol{u}, \boldsymbol{Y}\boldsymbol{v})$. Then, the order of the samples in one view was randomly permuted B times, and the correlation was computed using the same weight vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ to project the data [Lin et al., 2014]. The p-value was then estimated by counting how many times the correlations computed during the permutations were higher or equal to the one computed with non-permuted data. Note that this approach does not re-train the model after each permutation.

3.5 Applications in neuroimaging

This section will give a brief overview of some applications of CCA and PLS in neuroimaging studies. It will cover several variants of the methods, including its sparse versions: SCCA and SPLS.

3.5.1 Canonical Correlation Analysis (CCA)

As mentioned in Section 3.2, the objective of CCA is to find pairs of weight vectors which maximise the correlation between the projections of both views.

Usually, CCA is applied with some sort of regularisation. However, there are a few examples of non-regularised CCA in the literature, including studies which explore the autocorrelation of Functional Magnetic Resonance Imaging (fMRI) signals [Borga et al., 2002], and model the hemodynamic response with spatial relationships in fMRI [Friman et al., 2001]. Some studies have also used non-regularised CCA, but combined with dimensionality reduction steps like PCA [Smith et al., 2015, Rosa et al., (in preparation)] and ICA [Miller et al., 2016].

In its kernel version, KCCA has been used to find associations between: fMRI data and visual stimuli [Hardoon et al., 2007], brain volume and genetics [Hardoon et al., 2009], fMRI and hemodynamic responses [Wang et al., 2005], fMRI and multivariate labels [Blaschko et al., 2011], cortical thickness and clinical/demographic variables for autism [Misaki et al., 2012], etc.

Although it is outside the scope of this thesis, there have been extensions of CCA algorithms proposed in the literature which accommodate for more than two views, including: combining multi-set canonical correlation analysis (mCCA) with joint independent component analysis (jICA) to study schizophrenia using three views (fMRI, diffusion tensor imaging, and methylation data) [Sui et al., 2012]; using mCCA to study fMRI and EEG data on a trial-to-trial covariation across the modalities [Correa et al., 2010]; and using mCCA to study schizophrenia using three types of data (functional MRI, diffusion MRI, and structural MRI) [Sui et al., 2015a].

3.5.2 Partial Least Squares (PLS)

PLS has been applied in neuroimaging for many different tasks, including modeling and prediction. The specific name given to the PLS version applied in each study is often not related with the algorithm itself, but with the type of experimental design. The experimental designs of each one of these studies is outside the scope of this thesis, for a more detailed review, please refer to the paper by Krishnan et al. [2011].

The technique was fist introduced in functional neuroimaging by McIntosh et al. [1996] to compare encoding and recognition of faces using PET data as one view, and the reaction time of the subjects during a face matching and recognition task as the other view. Since then, PLS has been applied in many different studies, including: the study of Alzheimer's disease (AD), by combining PET data with other clinical information [Price et al., 2004]; to examine the relationship between brain and neuropsychological test scores in a schizophrenia dataset [Nestor et al., 2002]; to find relationships between cognitive abilities and grey matter morphology in healthy subjects [Ziegler et al., 2013]; to study emotional processing [Keightley et al., 2003b]; memory [Nyberg et al., 1996, Della-Maggiore et al., 2000]; personality [Keightley et al., 2003a], etc.

As stated in Section 3.3, PLS can be used to predict variables in a set $\boldsymbol{Y}$ from variables in a set $\boldsymbol{X}$, which is is sometimes referred to as Partial Least Squares

Regression (PLSR) [Krishnan et al., 2011]. This method had been applied for behaviour prediction, e.g. Giessing et al. [2007] used it to predict the effects of nicotine on behaviour. However, it has also been applied in other tasks, such as, finding shape dependencies between sub-cortical brain regions [Rao et al., 2008].

3.5.3 Sparse CCA and Sparse PLS

Since there seems to be a lack of a unified naming convention in the literature, it is not always clear which algorithms the authors apply in the various SCCA/SPLS studies, i.e. some studies claim to solve a SCCA optimisation problem, when it fact it is an approximation which makes it closer/equivalent to SPLS. Therefore, the applications of SCCA/SPLS will be described together in the same section.

SCCA/SPLS has been used in several genetics studies, since it is a field which deals with high dimensional data, thus, the ability to select smaller and more interpretable subsets of genes is an advantage. Indeed, several papers which proposed the first SCCA/SPLS approaches used such datasets to demonstrate their performance [Waaijenborg et al., 2008, Lê Cao et al., 2008, Parkhomenko et al., 2009, Witten et al., 2009, Witten and Tibshirani, 2009].

The use of SCCA/SPLS is still not as common in neuroimaging. However, there are a few examples in the literature of combining neuroimaging information with genetics. Le Floch et al. [2012] applied SPLS to a dataset comprised of 1 054 068 Single Nucleotide Polymorphisms (SNPs) and 34 fMRI Regions of Interest (ROIs) from subjects performing a general cognitive assessment fMRI task. The framework included a dimensionality reduction step applied to the SNPs, using an univariate filter prior to applying SPLS. The first two weight vector pairs were computed, and sparsity was only applied on the filtered SNPs (not on the fMRI ROIs) [Le Floch et al., 2012]. Both the sparsity hyper-parameter and the univariate filter threshold were optimised. SPLS was also compared with other methods, including PLS and CCA. Other SPLS studies using genetic and neuroimaging data include the ones by Lin et al. [2014] and Grellmann et al. [2015], which applied it to datasets from patients with schizophrenia. Unlike the study by Le Floch et al. [2012], these studies used whole-brain data, and did not apply a dimensionality reduction step. Moreover, Lin et al. [2014] applied a group sparse version of SPLS, which took into account *a priori* assumptions regarding the data structure.

SCCA/SPLS has been used in studies related to neurodegenerative diseases. Avants et al. [2010] applied the method proposed by Witten and Tibshirani [2009] to the study of Frontotemporal Dementia (FTD) and Alzheimer's Disease (AD), using imaging data in two views: structural T1-weighted MRI and Diffusion Tensor Imaging (DTI). One of the most interesting results of the study is that it applied a linear regression model using the projections of both views to predict the scores of neuropsychological tests. The results showed that the projections of each disease had an association with their corresponding clinical neuropsychological test, i.e. the images from patients with AD showed a significant association with the Mini-Mental State Examination (MMSE) and no significant association with verbal fluency, while the reverse was observed for patients with FTD [Avants et al., 2010]. Despite these interesting results, there were some limitations of the study, namely: the sparsity levels were set *a priori*; only the first weight vector pair was considered; and only unidirectional associations were investigated, i.e. positivity constrains were imposed on the weights in order to improve the interpretability of the model [Avants et al., 2010], however, these constraints may also limit the relationships captured by the weight vector pairs. The use of neuroimaging data in both views has also been used in other settings. For example, Rosa et al. [2015] used SPLS to estimate the similarity between two Arterial Spin Labeling (ASL) datasets from the same subjects using different drugs, and Sui et al. [2015b] used it to analyse a dataset comprised of T1-weighted structural MR images and DTI data.

In another study by Avants et al., neurodegenerative diseases were again the focus. However, this time the authors used SCCA to find correlations between structural MR data (grey matter) and clinical variables coming from the Philadelphia Brief Assessment of Cognition (PBAC) [Avants et al., 2014]. This test contains 20 variables grouped into 5 psychometric sub-scales, which test different cognitive and behavioral/comportment deficits. Avants et al. [2014] performed 5 tests, each one with the MR images as one view and the clinical variables of a specific sub-scale as the other view. The results showed that the dimensionality reduction provided by SCCA enhanced the ability to detect associations between multivariate psychometric batteries and network level grey matter density measures, claiming to be the first study to have done it [Avants et al., 2014]. However, this study applied sparsity on

the image voxels only and, once again, imposed positivity constraints on the weight vectors.

Other studies have applied SPLS to study the relationship between clinical scores and brain regions. Olson Hunt et al. [2014] have applied the method to a dataset comprised of brain ROIs ($\boldsymbol{X}$) and the final score of the Modified Mini-Mental State Examination ($\boldsymbol{Y}$). The authors then fitted several models with different number of components ($\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k\}$), and sparsity levels for $\boldsymbol{X}$, in order to determine how often each ROI was selected across all the models.

SPLS can also be used in a supervised classification setting, in this case, the variables of $\boldsymbol{Y}$ are categorical, this is known as Sparse Partial Least Squares Discrimination Analysis (SPLS-DA). Labus et al. [2015] applied SPLS-DA to study pain, the authors used a dataset with patients with Irritable Bowel Syndrome (IBS) and controls, the features in $\boldsymbol{X}$ were derived from structural brain ROIs while $\boldsymbol{Y}$ contained the labels. The sparsity was set by stability selection, and the predictive ability of the final model was assessed on a hold-out dataset. This ability was characterised by supervised learning classification metrics, such as: sensitivity, specificity, positive predictive value, and negative predictive value.

Note that there have been other extensions of SPLS/SCCA methods with different kinds of sparsity penalties, which are usually chosen based on *a priori* assumptions regarding data structure [Lin et al., 2014, Du et al., 2015]. However, these specific adaptations are beyond the scope of this thesis.

3.5.4 Limitations

Although there is an increased interest in the application of sparse eigen-decomposition methods to neuroimaging data, there are still a few limitations. The first is the fact that the proposed SPLS methods use matrix deflation approaches which are inherited from their non-sparse versions [Lê Cao et al., 2008, Witten et al., 2009]. However, it is known that these deflation strategies rely on assumptions which do not hold in sparse settings, and will lead to weight vector pairs which are no longer orthogonal [Mackey, 2008]. This issue is addressed in Chapter 4 with the proposal of an alternative deflation approach [Monteiro et al., 2014]. The issue is later revisited in Chapter 6 [Monteiro et al., 2016].

Despite the previous use of SCCA/SPLS with datasets containing neuroimaging

and clinical/demographic data [Avants et al., 2014, Olson Hunt et al., 2014], these usually rely on the use of sparsity on a single view. However, there might be an advantage to use sparsity in both views, and to apply different levels of sparsity per weight vector pair and per view, in order to capture the effects without relying on the assumption that the data on the demographic/behavioral view is not sparse. The first steps towards this direction are presented in Chapter 5 [Monteiro et al., 2015], and are later explored in Chapter 6, by going one level deeper, and finding associations between whole-brain MRI data and the individual items of a clinical exam [Monteiro et al., 2016].

Due to the high dimensionality of the neuroimaging data, most studies with eigen-decomposition methods in neuroimaging have focused on the use of linear models [Hardoon et al., 2007, Blaschko et al., 2011]. There are a few examples in the literature which use non-linear versions of CCA for specific applications with low dimensional data [Dong et al., 2015]. In exploratory approaches using neuroimaging and behaviour, the dimensionality of the view containing the behaviour data is usually much lower than the view containing the neuroimaging data. Therefore, it is interesting to explore the possibility of modeling potential non-linearities on the lower dimensional view, while maintaining sparsity constraints on the higher dimensional view. In order to explore this possibility, a novel primal-dual SCCA approach is proposed in Chapter 8, which contains both sparsity and non-linear properties.

Chapter 4

Alternative matrix deflation strategy for SPLS

As mentioned in Section 3.5.4, the deflation step originally proposed by Witten et al. [2009] does not provide orthogonal sparse weight vector pairs. The consequence of this non-orthogonality is that each weight vector pair obtained after matrix deflation might not express a new effect in the data, instead, it may fit the same effect described by its predecessor.

In this chapter, an alternative deflation step is described, which tries to address this issue [Monteiro et al., 2014].

4.1 Introduction

Witten et al. [2009] proposed a SPLS approach which has gained some popularity in neuroimaging, being applied in a few studies, either directly or with some variant [Wan et al., 2011, Rosa et al., 2015]. The algorithm computes a set of sparse weight vector pairs, by computing each pair individually and deflating the covariance matrix between the two views in order to compute the next pair (Algorithm 3.4). However, this step is performed using Hotelling's deflation, which may not be the most adequate procedure for sparse methods, failing to provide sets of orthogonal weight vectors [Mackey, 2008].

In this chapter, the use of an alternative deflation method is proposed, which tries to enforce orthogonality between the weight vector pairs. This should be of utmost importance if one wants to acquire a set of sparse weight vector pairs which express different effects in the data.

In order to test the proposed deflation strategy, the SPLS method described by

Witten et al. [2009] was implemented and applied to an open-access dementia dataset consisting of T1-weighted MRI images and clinical/demographic information [Marcus et al., 2009]. The weight vectors were then obtained with the Hotelling's deflation step described by Witten et al. [2009] and the proposed deflation step. By comparing both methods in terms of the orthogonality of the computed weight vector pairs, the features that each one selects, and how these influence the projections of the subjects onto the latent space, this chapter aims to provide some intuition as to why one should adopt a deflation step different from the one proposed by Witten et al. [2009].

4.2 Materials and Methods

4.2.1 Proposed deflation

The Hotelling's deflation has been used in many algorithms, including PLS, SPLS, PCA, and Sparse PCA (SPCA). However, as noted by Mackey [2008], the use of this deflation strategy with a sparse method is usually not appropriate. Mackey [2008] focused on SPCA, and notes that by imposing sparsity on the optimisation problem, the solutions will no longer correspond to the eigenvectors of $\boldsymbol{X}^\intercal\boldsymbol{X}$, but to pseudo-eigenvectors. In other words, sparse weight vectors which try to explain the maximum covariance in the data, but some of the properties associated with eigenvectors no longer hold, including the orthogonality between them [Mackey, 2008]. A similar parallel can be drawn for the use of the Hotelling's deflation with SPLS.

Let $\boldsymbol{C}$ be the covariance matrix between $\boldsymbol{X}$ and $\boldsymbol{Y}$, and $\widetilde{\boldsymbol{C}}$ the covariance matrix after being deflated. $\widetilde{\boldsymbol{C}}$ can be computed by computing the deflated versions of $\boldsymbol{X}^\intercal$ and $\boldsymbol{Y}$, and multiplying them: $\widetilde{\boldsymbol{C}} = \widetilde{\boldsymbol{X}}^\intercal\widetilde{\boldsymbol{Y}}$. These can be computed using the projection deflation:

$$\widetilde{\boldsymbol{X}} = \boldsymbol{X}(\boldsymbol{I}_x - \boldsymbol{u}\boldsymbol{u}^\intercal) \quad \text{and} \quad \widetilde{\boldsymbol{Y}} = \boldsymbol{Y}(\boldsymbol{I}_y - \boldsymbol{v}\boldsymbol{v}^\intercal) \tag{4.1}$$

where $\boldsymbol{u}$ and $\boldsymbol{v}$ are the weight vectors of $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively; $\boldsymbol{I}_x$ and $\boldsymbol{I}_y$ are the identity matrices of $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively.

The deflated covariance matrix can be computed as:

$$\begin{aligned}\widetilde{\boldsymbol{C}} &= \widetilde{\boldsymbol{X}}^{\intercal}\widetilde{\boldsymbol{Y}} \\ &= (\boldsymbol{I}_x - \boldsymbol{u}\boldsymbol{u}^{\intercal})^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{Y}(\boldsymbol{I}_y - \boldsymbol{v}\boldsymbol{v}^{\intercal}) \\ &= \boldsymbol{X}^{\intercal}\boldsymbol{Y} - \boldsymbol{X}^{\intercal}\boldsymbol{Y}\boldsymbol{v}\boldsymbol{v}^{\intercal} - \boldsymbol{u}\boldsymbol{u}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{Y} + \boldsymbol{u}\boldsymbol{u}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{Y}\boldsymbol{v}\boldsymbol{v}^{\intercal} \\ &= \boldsymbol{X}^{\intercal}\boldsymbol{Y} - \boldsymbol{X}^{\intercal}\boldsymbol{Y}\boldsymbol{v}\boldsymbol{v}^{\intercal} - \boldsymbol{u}\boldsymbol{u}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{Y} + d\boldsymbol{u}\boldsymbol{v}^{\intercal} \qquad (4.2)\end{aligned}$$

It can be shown that the projection deflation (as expressed by Equation 4.2) is equivalent to the Hotelling's deflation, if $\boldsymbol{u}$ and $\boldsymbol{v}$ are true singular vectors and d its corresponding singular value.

Equation 3.4, states that SVD decomposes a matrix $\boldsymbol{M}$ in the following way: $\boldsymbol{M} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^{\intercal}$. Note that PLS-SVD (which is the basis for SPLS), is equivalent to applying SVD to the covariance matrix $\boldsymbol{C}$ (Section 3.3), therefore, let $\boldsymbol{M} = \boldsymbol{C}$, since $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthonormal matrices, Equation 3.4 can be re-written as:

$$\boldsymbol{C}\boldsymbol{V} = \boldsymbol{U}\boldsymbol{D} \quad \text{or} \quad \boldsymbol{U}^{\intercal}\boldsymbol{C} = \boldsymbol{D}\boldsymbol{V}^{\intercal}$$

This means that, for a singular weight vector pair, one can write the following:

$$\boldsymbol{X}^{\intercal}\boldsymbol{Y}\boldsymbol{v} = d\boldsymbol{u} \quad \text{and} \quad \boldsymbol{u}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{Y} = d\boldsymbol{v}^{\intercal} \qquad (4.3)$$

By substituting (4.3) in (4.2), the Hotelling's deflation is recovered:

$$\begin{aligned}\widetilde{\boldsymbol{C}} &= \boldsymbol{X}^{\intercal}\boldsymbol{Y} - \boldsymbol{X}^{\intercal}\boldsymbol{Y}\boldsymbol{v}\boldsymbol{v}^{\intercal} - \boldsymbol{u}\boldsymbol{u}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{Y} + d\boldsymbol{u}\boldsymbol{v}^{\intercal} \\ &= \boldsymbol{X}^{\intercal}\boldsymbol{Y} - d\boldsymbol{u}\boldsymbol{v}^{\intercal} - d\boldsymbol{u}\boldsymbol{v}^{\intercal} + d\boldsymbol{u}\boldsymbol{v}^{\intercal} \\ &= \boldsymbol{X}^{\intercal}\boldsymbol{Y} - d\boldsymbol{u}\boldsymbol{v}^{\intercal} \qquad (4.4)\end{aligned}$$

However, due to the sparsity constraints imposed by SPLS, $\boldsymbol{u}$ and $\boldsymbol{v}$ are not true singular vectors, which means that Equations 4.2 and 4.4 are not equivalent.

Witten et al. [2009] do acknowledge the lack of orthogonality of the weight vector pairs using their deflation method. However, their proposed alternative tries to enforce the orthogonality of $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k\}$, while not guaranteeing that the same will hold for $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$. Subsequent publications using SPLS with neuroimaging

data seem to adopt the original Hotelling's deflation. Thus, this chapter will focus on comparing the Hotelling's deflation (Equation 4.4) with projection deflation (Equation 4.2)

4.2.2 Dataset

A subset from the "OASIS: Longitudinal MRI Data in Nondemented and Demented Older Adults" dataset (`www.oasis-brains.org`) was used to compare both deflation strategies. The dataset includes T1-weighted MRI scans from subjects at different time points, with the corresponding clinical information [Marcus et al., 2009]. In this study, a subset of 142 subjects from the first time point was used. These included 58 male subjects and 84 female subjects with an average age of 75.4 years ± 7.7 years. Among these, 72 were considered as being healthy, 56 as demented, and 14 as "converted". The "converted" subjects correspond to subjects which were considered as healthy at the first time point, but later developed dementia.

All images were preprocessed using SPM12b [Ashburner et al., 2013]. The first step was to average all the repeats for each session followed by a grey matter segmentation, then, the segmented images were registered using DARTEL [Ashburner, 2007], normalised to MNI space [Mazziotta et al., 1995] with isotropic 2 mm voxels and smoothed with a Gaussian kernel with a Full Width at Half Maximum (FWHM) of 8 mm. In addition, the head size was regressed out of the data and a mask was applied to select voxels that had a probability of being grey matter equal or above 20%. The grey matter probability maps within the mask were used as the neuroimaging view ($\boldsymbol{X}$) of SPLS.

For each subject, the following clinical/demographic information was used as the clinical view ($\boldsymbol{Y}$) of SPLS: age, Socioeconomic Status (SES), education, Mini-Mental State Examination (MMSE), and Clinical Dementia Rating (CDR). The SES was assessed by the Hollingshead Index of Social Position and classified into five categories from 1 (highest SES) to 5 (lowest SES), the MMSE is a clinical score used to grade the cognitive state of patients from 0 (worst) to 30 (best), and the CDR is a score grouping the subjects into 4 categories: "no dementia" (CDR = 0), "very mild Alzheimer's Disease (AD)" (CDR = 0.5), "mild AD" (CDR = 1) and "moderate AD" (CDR = 2). For more information, please refer to the paper by Marcus et al. [2009].

All features in both views were mean-centered and normalised, such that each

feature vector had zero mean and l_2-norm equal to 1.

4.2.3 Comparison framework

In order to compare both deflation approaches in their ability to provide sets of orthogonal weight vectors, 100 random subsamples of 50% of the data were performed. For each subsample, SPLS was used to compute the first three weight vector pairs using Hotelling's deflation (Equation 4.4), and using the proposed approach with projection deflation (Equation 4.2), i.e. the covariance matrix $\boldsymbol{C}$ was deflated twice with each deflation method. This choice allows one not only to assess if consecutive weight vector pairs are orthogonal (i.e. $\boldsymbol{u}_1 \perp \boldsymbol{u}_2$ and $\boldsymbol{u}_2 \perp \boldsymbol{u}_3$), but also to assess if non-consecutive weight vector pairs are orthogonal (i.e. $\boldsymbol{u}_1 \perp \boldsymbol{u}_3$). The dot products between the weight vectors were then compared. In other words, for each random subsample, the following dot products were computed: $\langle \boldsymbol{u}_1, \boldsymbol{u}_2 \rangle$, $\langle \boldsymbol{u}_1, \boldsymbol{u}_3 \rangle$, $\langle \boldsymbol{u}_2, \boldsymbol{u}_3 \rangle$, $\langle \boldsymbol{v}_1, \boldsymbol{v}_2 \rangle$, $\langle \boldsymbol{v}_1, \boldsymbol{v}_3 \rangle$, and $\langle \boldsymbol{v}_2, \boldsymbol{v}_3 \rangle$.

The sparsity hyper-parameters were set to $c_u = 50$ and $c_v = \sqrt{5}$ for both deflation procedures. This hyper-parameter combination will not only guarantee that both l_1-norm and l_2-norm constraints are active (Section 3.4), but it will also force $\boldsymbol{u}$ to be sparse, while not being strong enough to force $\boldsymbol{v}$ to be sparse. This strategy allows one to observe the effects of the deflation strategies in a view with sparsity, and in a view without sparsity. Note that finding the optimal level of sparsity is not the aim of this chapter, therefore, the hyper-parameter values were fixed *a priori*. The main focus of this chapter is on the effects of the deflation step. Further investigations into strategies for hyper-parameter selection will be presented in Chapters 5 and 6.

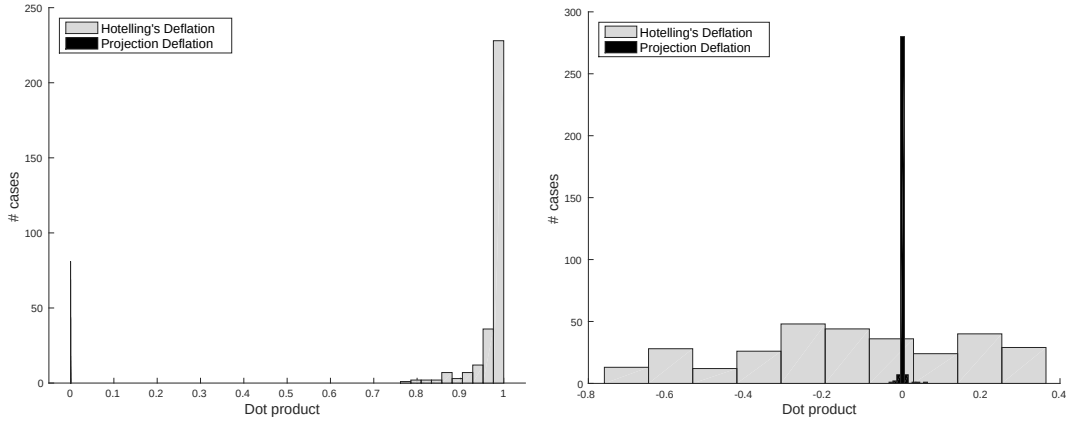
4.3 Results and Discussion

The dot products between the weight vector pairs were computed for each one of the 100 random subsamples of the data, these dot products were then plotted for both deflation methods (Figure 4.1). Note that the dot product between two unit vectors (i.e. $\|\cdot\|_2 = 1$) will be equal to 0 if they are orthogonal, and to -1 or 1 if they are collinear.

As one can see in Figure 4.1(a), the clinical weight vectors ($\boldsymbol{v}$) computed using Hotelling's deflation were not orthogonal, the dot products were actually close to 1, which means that the computed weight vectors were close to being collinear. On the

other hand, the dot products of the clinical weight vectors computed using projection deflation were all very close to 0 (they ranged between -1.04×10^{-15} and 1.05×10^{-15}), which means that projection deflation was able to provide approximately orthogonal clinical weight vectors.

Figure 4.1(b) shows the dot products of the image weight vectors ($\boldsymbol{u}$). Unlike the dot products for the clinical weight vectors using Hotelling's deflation, these were not gathered close to 1, they were distributed along a larger range of values. This may be due to the fact that they were sparse and had a much higher dimensionality, which means that some features might be selected for some random subsample of the data but not for another, making the dot products less consistent. However, projection deflation was still able to compute approximately orthogonal weight vectors. The distribution of the dot products was broader (Figure 4.1(b)) than the one for the clinical weight vectors (Figure 4.1(a)), but still quite narrow (they ranged between -0.03 and 0.06) and centered at 0.



(a) Clinical weight vectors ($\boldsymbol{v}$) **(b)** Image weight vectors ($\boldsymbol{u}$)

Figure 4.1: Dot product between the computed weight vectors.

One of the advantages of trying to enforce the orthogonality of the weight vectors can be seen in Figure 4.2, which shows the weight maps for the first three image weight vectors $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3\}$ computed using all available data. The first weight vector will be the same, since the covariance matrix ($\boldsymbol{C}$) has not been deflated yet. However, after the first deflation, one can see that the weight vector computed using Hotelling's deflation ($\boldsymbol{u}_2$) selected features surrounding the ones selected by the previous weight vector ($\boldsymbol{u}_1$), and the third weight vector ($\boldsymbol{u}_3$) selected features surrounding the ones selected by the second weight vector ($\boldsymbol{u}_2$). This is not the case for projection deflation,

where each new weight vector selected features located in different regions.

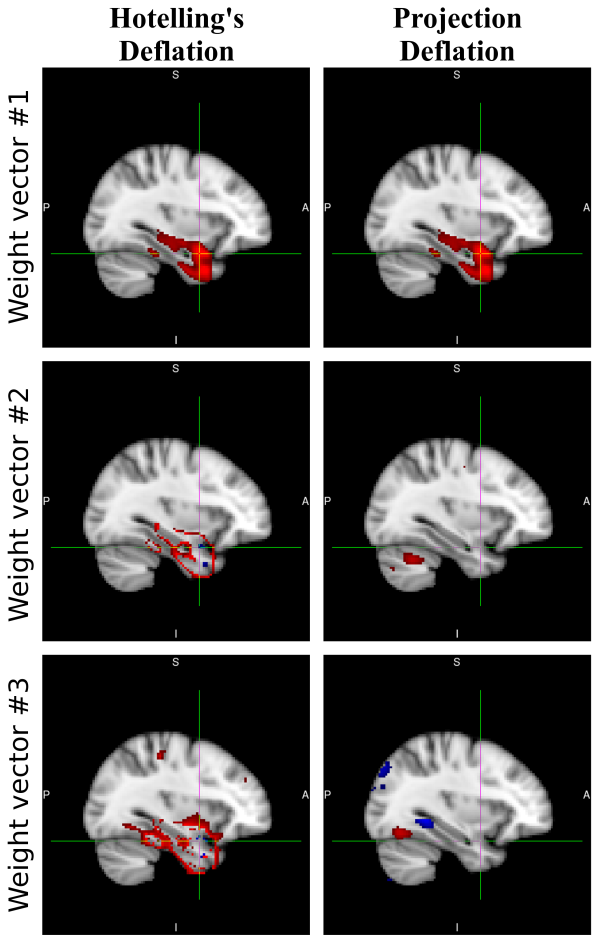


Figure 4.2: First three image weight vectors computed using each deflation step. Red regions correspond to positive weight values and blue regions correspond to negative weight values.

An analogous effect was observed in the clinical weight vectors (Figure 4.3). Even though these were not sparse, the use of Hotelling's deflation had little effect on the values of the entries from the second and third weight vector pairs. This was not the case for the clinical weight vectors computed using projection deflation, whose entries changed after each deflation step.

These results suggest that SPLS with Hotelling's deflation detected the same effect in the data for each weight vector pair. By trying to enforce the weight vectors to be orthogonal, the effects described by the previous weight vectors were removed from the data, allowing new effects to be detected.

The weight vectors were also used to plot the projections of the subjects onto the lower dimensional sub-space (Figure 4.4). As one can see in Figures 4.4(a) and 4.4(b),

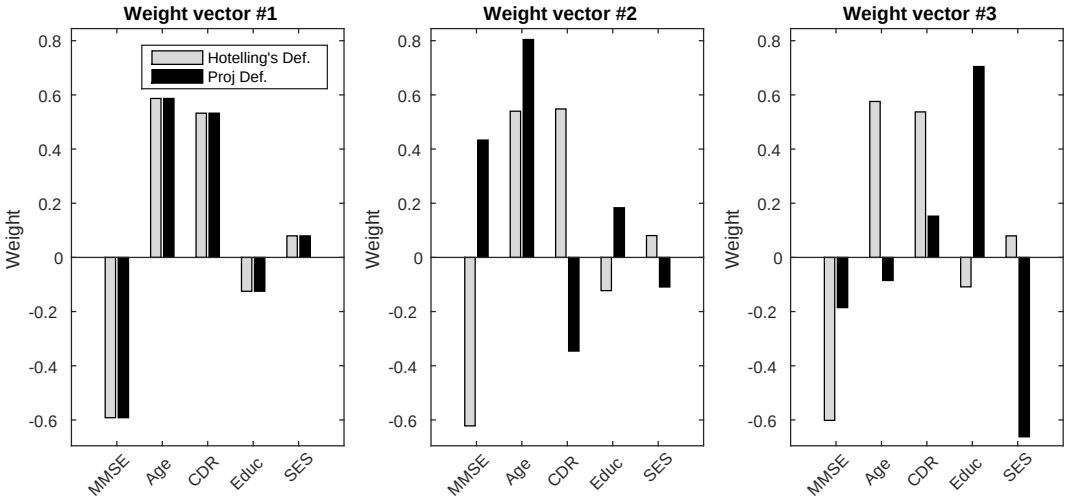
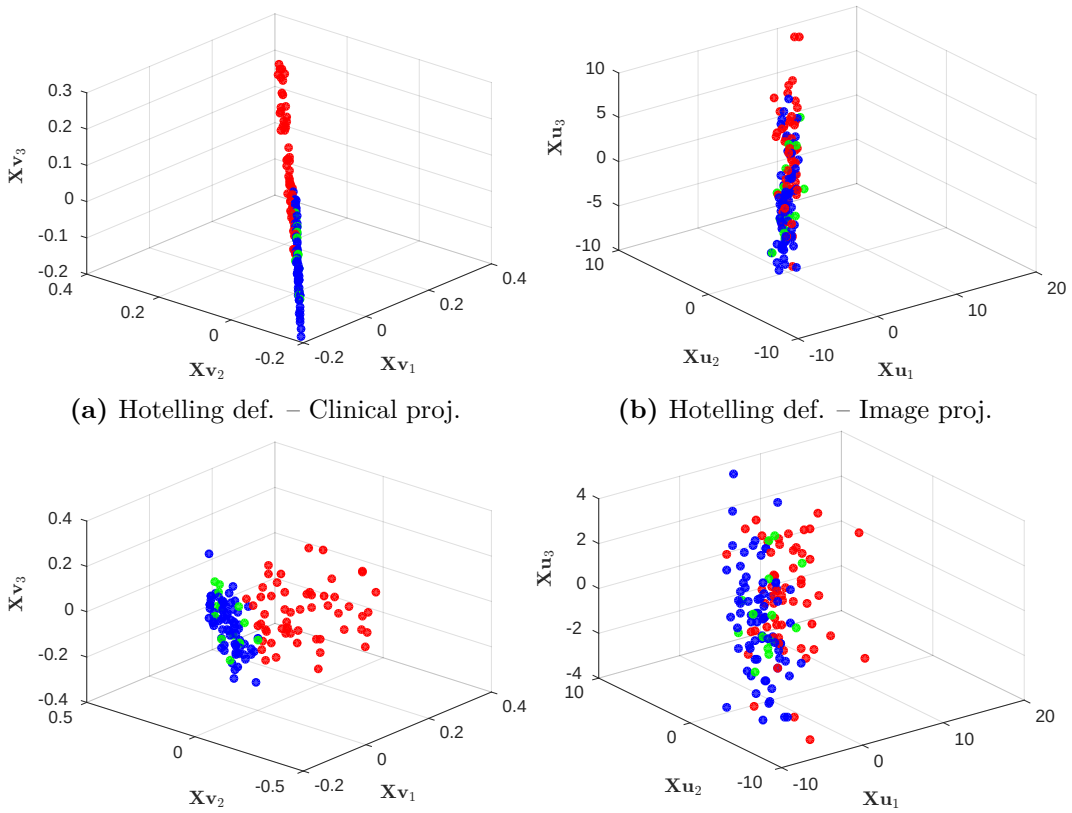


Figure 4.3: First three clinical weight vectors computed using each deflation step.

the non-orthogonality of the weight vectors computed using Hotelling's deflation lead to projections which lie along a line, failing to capture the covariance of the data.

By using a projection deflation scheme (Figures 4.4(c) and 4.4(d)), the covariance of the data is explored for subsequent weight vector pairs. It may even lead to situations in which the separation between groups becomes more apparent, this can be seen in Figure 4.4(c), where the demented (red) vs. non-demented (blue) subjects form two separate clusters. However, one should note that some of the clinical features are correlated with the group membership (i.e. demented and non-demented), which explains why the algorithm was able to explore this information to split the groups.

Figure 4.4(d) shows that the projection of the subjects onto $\boldsymbol{u}_2$ and $\boldsymbol{u}_3$ computed using projection deflation allowed from some demented subjects to move further away from the non-demented ones, when compared with the same projections using Hotelling's deflation (Figure 4.4(b)). However, the separation is not as clear as in the projection onto the clinical weight vectors (Figure 4.4(c)).



(a) Hotelling def. – Clinical proj.

(b) Hotelling def. – Image proj.

(c) Projection def. – Clinical proj.

(d) Projection def. – Image proj.

Figure 4.4: Projections of the data matrices onto the weight vectors computed using both deflation strategies. **Blue** – Non-demented; **Green** – Converted; **Red** – Demented.

4.4 Conclusion

Due to the increased interest in PLS and CCA, several neuroimaging and genetics studies have applied, and continue to apply, sparse variants of these methods using Hotelling's deflation [Lin et al., 2013, 2014, Rosa et al., 2015], perhaps unaware of the issues associated with this deflation strategy. More specifically, the lack of orthogonality between the weight vectors. Therefore, it is important to raise awareness for the practical consequences of this lack of orthogonality.

The results show that projection deflation performed better when trying to enforce the orthogonality of the weight vectors, for both sparse and non-sparse views. Even though the weight vectors computed using projection deflation were not exactly orthogonal, they were substantially closer to being orthogonal than the ones computed using Hotelling's deflation. Moreover, despite the introduction of two terms to the deflation operation, i.e. the proposed deflation method (Equation 4.2)

includes two extra terms which do not exist in the Hotelling's deflation (Equation 4.4), the increase in computational time was negligible when compared with the time necessary to compute each SPLS weight vector pair.

By trying to enforce orthogonality between the weight vectors, new brain regions were found, which suggests that using projection deflation may help uncover more associative effects in the data, instead of computing weight vector pairs which describe the same effect after each deflation.

In order to capture the covariance of the data in the sub-space spanned by the weight vector pairs, it is very important to try to enforce the orthogonality of these vectors. The results showed that new directions capturing more covariance in the data were found when applying projection deflation, as opposed to Hotelling's deflation.

The analysis of the projection sub-space could provide very useful information regarding the underlying relationships between the two views. These might contain information which may improve the understanding of brain function, assist with the diagnosis of brain diseases, or enable patient stratification. For example, one may be able to find different directions of covariance which are associated with different subgroups of patients, which may provide insights into previously unknown properties of certain brain disorders.

Note that there are other popular deflation strategies which could be used with SPLS besides the Hotelling's deflation, e.g. the deflation strategy used by PLS Mode-A (Section 3.3). Further comparisons between this deflation strategy and projection deflation will be presented in Chapter 6.

Chapter 5

SPLS using two-view sparsity constraints

So far, all the experiments with SPLS have been performed with fixed sparsity hyper-parameters $\{c_u, c_v\}$ (Chapter 4). Fixing one or both SPLS sparsity hyper-parameters has been proposed in the literature [Avants et al., 2010, 2014], however, this relies heavily on assumptions regarding the structure of the data, i.e. it assumes that one knowns *a priori* how many informative features exist in each view. In order to study the dataset in an exploratory way, one should select the hyper-parameters $\{c_u, c_v\}$ based on the data, and not on previous assumptions.

This chapter will describe a framework that was proposed to acquire multiple pairs of SPLS weight vectors $\{\boldsymbol{u}_h, \boldsymbol{v}_h\}$ with different sparsity levels for each one [Monteiro et al., 2015]. These results showed an early application of SPLS to cases in which one wishes to obtain solutions that are sparse in both views, where these are comprised of whole-brain neuroimaging data ($\boldsymbol{X}$) and clinical/demographic data ($\boldsymbol{Y}$).

5.1 Introduction

Exploratory approaches, such as SPLS and SCCA, may provide useful insights into the brain's mechanisms, by finding relationships between different types of measures (i.e. views) from the same subjects. Some applications of these methods include finding associations between genetic information and brain ROI information [Le Floch et al., 2012], genetics and whole-brain [Grellmann et al., 2015], two different neuroimaging modalities [Avants et al., 2010, Sui et al., 2015b, Rosa et al., 2015], and neuroimaging and clinical/demographic scores [Avants et al., 2014, Monteiro et al.,

2014]. However, previous studies often overlook the possibility of using a combination of sparsity in both views and different levels of sparsity for each weight vector pair. This idea was proposed by Lê Cao et al. [2008] for genetics. However, as far as we are aware, it had not been applied to explore the relationships between whole-brain voxel-wise structural MRI data, and clinical/demographic scores [Monteiro et al., 2015].

Each SPLS weight vector pair expresses a multivariate association between two data matrices ($\boldsymbol{X}$ and $\boldsymbol{Y}$), which can be viewed as a "multivariate associative effect" in the data. The use of sparsity constraints may improve the interpretability of a model, by removing uninformative features. However, the adequate amount of sparsity necessary to express each one of these multivariate associative effects is usually not known *a priori*. Therefore, by optimising the sparsity levels separately for each weight vector pair, one can hopefully estimate a data-driven model of the multivariate associative effects present in the data without relying on *a priori* assumptions.

This chapter will describe a proposed framework to find multivariate associations between combinations of clinical/demographic features and brain voxels, which provides a solution with different levels of sparsity per weight vector pair and per view. The framework automatically selects the sparsity level to describe the strongest effect in the data, removes it from the data using projection deflation (Chapter 4), and repeats the process until all the statistically significant weight vector pairs are found.

5.2 Materials and Methods

5.2.1 Proposed framework

Each view was organised in a data matrix where each row corresponded to a subject and each column to a feature. This was done for both image ($\boldsymbol{X}$) and clinical ($\boldsymbol{Y}$) views, i.e. $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ and $\boldsymbol{Y} \in \mathbb{R}^{n \times q}$, where n corresponds to the number of subjects, and p and q to the number of features in each view, respectively.

The SPLS hyper-parameters $\{c_u, c_v\}$ (Equation 3.14) were selected by grid-search. In other words, for each $\{c_u, c_v\}$ pair, the following procedure was performed:

1. Compute one SPLS weight vector pair using the procedure described in Algo-

rithm 3.4.

2. Project the data onto the computed weight vector pair and calculate the covariance between the projections: $\sigma = \mathrm{Cov}(\boldsymbol{X}\boldsymbol{u}, \boldsymbol{Y}\boldsymbol{v})$.

3. Repeat 1000 times:

   (a) Randomly permute the rows of $\boldsymbol{Y}$, i.e. permute the order of the subjects in one of the views.

   (b) Compute one SPLS weight vector pair using Algorithm 3.4.

   (c) Project the data onto the computed weight vector pair and calculate the covariance between the projections (σ_b^*).

4. Fit a Gaussian distribution to the covariance values calculated in Step 3(c) (σ_b^*), and compute the p-value by integrating the area under the distribution function for which the values are greater or equal σ (Step 2).

After the p-value for each hyper-parameter combination was determined, the chosen weight vector pair was the one with the lowest p-value. The reason a Gaussian distribution was fitted to the values calculated in Step 3(c) (σ_b^*) instead of merely calculating the p-value by computing the fraction of times $\sigma_b^* \geqslant \sigma$, is because several hyper-parameter combinations might have the same number of permutations where $\sigma_b^* \geqslant \sigma$, which leads to a situation where the best hyper-parameter combination cannot be determined. However, if a Gaussian distribution is fitted to the values of σ_b^*, then, it will also take into account the spread of σ_b^*, which makes it less likely that two hyper-parameter combinations will have the same p-value. Note that there might be a situation where several hyper-parameter combinations have the same p-value. In such cases, a criterion based on interpretability can be used, by choosing among the possible candidates the solution which provides the sparsest clinical weight vector which explains the maximum covariance.

The selection of the hyper-parameters allows the framework to find the sparsity combination in both views which expresses the strongest effect in the data.

The ranges of values used for grid search were $c_{\mathrm{x}} = \{10, 20, \ldots, \sqrt{p}\}$ and $c_{\mathrm{y}} = \{1.0, 1.1, \ldots, \sqrt{q}\}$. All models were run using the same permutations.

After the optimal hyper-parameter pair $\{c_u, c_v\}$ was selected, it was used to compute a weight vector pair $\{\boldsymbol{u}, \boldsymbol{v}\}$, which was then used to deflate the data matrices

using a projection deflation strategy (Equation 4.1). Then, the next weight vector pair was computed using the steps described above. The process was repeated until k weight vector pairs were computed, where $k = \min\{\text{rank}(\boldsymbol{X}), \text{rank}(\boldsymbol{Y})\}$.

By performing these deflations, one removes the previous effect from the data, allowing new effects to be detected, which will be "ranked" by the amount of covariance they explain. In other words, the first weight vector pair will explain the maximum covariance in the data, and after this is removed, the weight vector pair that explains the second largest covariance in the data will be obtained, and so on until the computed weight vector pairs are no longer statistically significant ($p \geqslant 0.05$). When that point is reached, the computed weight vectors will start to fit noise.

Note that a similar permutation based framework was proposed by Witten and Tibshirani [2009]. However, in that approach, a Gaussian function was not fitted to σ_b^*, the covariance matrix was deflated using Hotelling's deflation, and the correlation was used as a metric to compute the p-values. In the proposed framework, the covariance was used as a metric, so that it is consistent with the objective function that SPLS tries to maximise (Equation 3.14).

5.2.2 Dataset

The dataset used in the current chapter, including the image preprocessing, was the same as the one in Chapter 4. However, the head size was not regressed out of the data. Moreover, in order to keep the approach consistent with previous publications using SPLS, all features in both views were mean-centered and normalised, so that each one had a mean of zero and a standard deviation equal to 1.

The final dataset contained 174548 features in the data matrix $\boldsymbol{X}$ (i.e. each feature corresponding to a voxel), and six features in the data matrix $\boldsymbol{Y}$ (the "sex" variable was added to the model as well): sex, age, Socioeconomic Status (SES), education, Mini-Mental State Examination (MMSE) and Clinical Dementia Rating (CDR).

5.3 Results and Discussion

From the six weight vectors pairs computed by SPLS ($\min\{\text{rank}(\boldsymbol{X}), \text{rank}(\boldsymbol{Y})\} = 6$), three were considered statistically significant. As one can see in Table 5.1, the p-value

suffered a large increase from the 3rd to the 4th weight vector pair, which suggests that all the relevant effects were captured by the first three weight vector pairs.

Table 5.1: Optimal sparsity hyper-parameters per weight vector pair.

Weight vector pair	c_u	c_v	p-value
1	140	1.0	0.0000
2	90	2.2	5.1614e-13
3	10	1.5	2.1624e-07
4	10	1.0	0.3238
5	50	1.8	0.6316
6	390	1.0	0.7376

The statistically significant clinical weight vectors are shown in Figure 5.1, and the corresponding image weight vectors in Figure 5.2

The first weight vector pair captured the effect of age (Figure 5.1), which seems to be spread throughout multiple brain regions (Figure 5.2(a)). Grey matter density decreases with age, so the effect is expressed by a positive weight on age and corresponding large clusters of negative weights in the image view. After deflation, this effect is removed from the data, which explains why age is no longer selected in the remaining statistically significant clinical weight vectors (Figure 5.1).

The second weight vector pair described a combination of multiple clinical/demographic features, with MMSE and CDR having the largest absolute weights. These correspond to scores for clinical tests performed in individuals with dementia. It is interesting to note that the corresponding image weights selected large clusters of voxels in the hippocampus and temporal regions (Figure 5.2(b)), which have been previously associated with dementia [Jack et al., 2000]. Note that Figures 5.2(a) and 5.2(b) show the same image slices, if one compares the figures, it is possible to see that the effects observed in the hippocampus in the second weight vector were not present in the first weight vector, which suggests that the algorithm was able to separate effects related with aging (first weight vector) from effects related with dementia (second weight vector).

The third effect was mainly guided by the "sex" variable and it associates four clinical/demographic features to three small clusters: one in the thalamus (Figure 5.2(c)) and two on the posterior region of the temporal lobes.

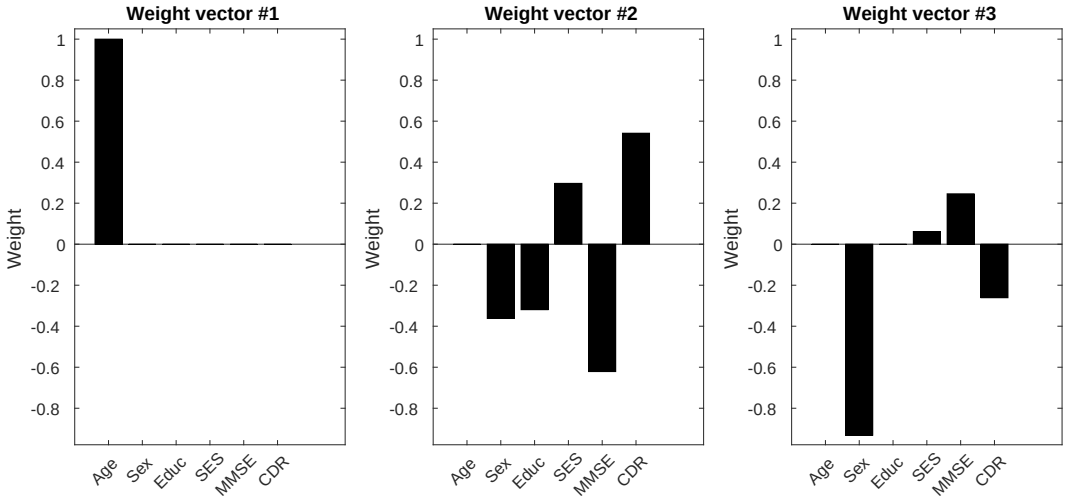
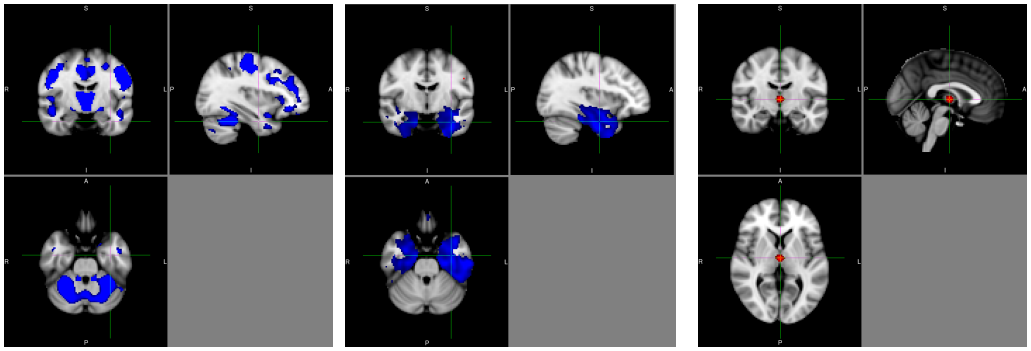


Figure 5.1: Statistically significant clinical weight vectors obtained with the corresponding clinical features.



(a) Image weight vector #1 **(b)** Image weight vector #2 **(c)** Image weight vector #3

Figure 5.2: Image weight vectors. Blue regions correspond to negative weight values, and red regions to positive weight values.

5.4 Conclusion

SPLS was able to find statistically significant multivariate associative effects in a dementia dataset between clinical/demographic information and whole-brain MRI scans. These were described by sparse weight vectors pairs, with adaptive sparsity levels per view and per pair. This contrasts with previous work using sparse eigen-decomposition methods to study brain and behaviour [Avants et al., 2014], where some *a priori* assumptions are made on the sparsity of the weight vectors.

The adaptive nature of the algorithm allowed not only to determine an optimal sparse solution, but also provided the flexibility to select the adequate number of clinical/demographic features and voxels to describe each multivariate associative effect, removing it from the data before finding the next one. This enabled it to

distinguish the effects associated with age from the ones associated with dementia.

Despite the encouraging results obtained with the proposed framework, one should bear in mind that it is also a computationally expensive approach, due to the fact that a grid search has to be performed to find the best hyper-parameter combination $\{c_u, c_v\}$ for each individual weight vector pair, which in this study resulted in the computation of SPLS 630 000 times per weight vector pair. Moreover, even though using a Gaussian function to fit the distribution of σ_b^* has shown good results for the considered dataset, it might not provide the best possible fit. Future work on permutation based frameworks should focus on determining the best way to fit the distribution of σ_b^*. Moreover, comparisons between metrics, i.e. correlation vs. covariance, should also be made.

Chapter 6

Multiple hold-out framework for SPLS

Despite the promising results obtained with the framework described Chapter 5, subsequent tests showed some disadvantages which would make it unsuitable to analyse some datasets. More specifically, the framework showed the tendency to give the same low p-value to several hyper-parameter combinations, which meant that one could not rely on a single metric for hyper-parameter selection. Moreover, the framework only selected a hyper-parameter combination based on how the model performed using all the data available. This did not give any insights into how SPLS would fit data which were not used for the training, and how robust were these fits to perturbations in the dataset.

This chapter will describe a multiple hold-out framework which was proposed to be an alternative to both permutation based frameworks (such as the one described in Chapter 5), and a nested-CV approach (Section 2.2.2). The framework was applied to investigate multivariate associations between voxel-wise neuroimaging data and individual questions/tasks from a cognitive test which, at the time this study was published, was a novel application [Monteiro et al., 2016].

6.1 Introduction

As shown in Chapter 5, exploratory machine learning approaches, such as SPLS, may provide useful insights into the brain's mechanisms by finding relationships between different measures (i.e. views) from the same subjects, more specifically, between neuroimaging and clinical/demographic data in a clinical population. By identifying these relationships, one can potentially improve the current understanding of disease mechanisms.

One of the advantages of using a sparse method, such as SPLS, is that it

removes noisy features from the model, which greatly improves interpretability. The number of features that are included in the model is controlled by a pair of sparsity hyper-parameters (one for each view). These should be tuned, in order to select the adequate amount of sparsity per view. As described in Section 3.4.1, Witten et al. [2009] proposed a framework to perform this selection based on permutations, i.e. for each hyper-parameter pair $\{c_u, c_v\}$ a permutation test was performed using the whole dataset and the hyper-parameter pair which resulted in the lowest p-value was chosen. A variant of this method was proposed and applied to a dementia dataset in Chapter 5. However, subsequent tests showed that it presented some disadvantages. In some cases, several hyper-parameter combinations tended to have the same very low p-value, which made the choice of the optimum not obvious. In addition, the behaviour of the framework for small values of c_u could be a bit unpredictable, e.g. some models would select a single voxel as being the optimal number of features for the neuroimaging view, which is not biologically plausible for structural MRI data. Moreover, a permutation based framework will select the hyper-parameter combination which gives rise to the lowest p-value obtained by permuting the whole dataset and re-training many times. However, there is no guarantee that this hyper-parameter combination will lead to a good performance in data that were not used to train the initial model. The framework is also unable to give any insights on whether this hyper-parameter combination is robust to perturbations in the initial dataset. This property is important, since one wishes to find an effect that is consistent in the dataset, and not a result that would arise by relying solely on the initial configuration of the data.

In order to train a model which generalises well for unseen data, one could use a three way split approach (Figure 2.5). However, since neuroimaging datasets are not large enough, one would usually adopt a nested-CV approach (Figure 2.6). Hyper-parameter tunning would then be performed in the following way: for each train fold, the optimal hyper-parameter combination is selected by performing a grid-search over all combinations using a cross-validation (CV) procedure to test each combination. The results are then statistically evaluated by performing a permutation test on the nested-CV, i.e. the order of the data is randomly permuted in one of the views, and the nested-CV procedure is repeated many times. Using this

procedure with SPLS for high-dimensional datasets with two hyper-parameters to optimise is not viable, due to the very large computational time. Indeed, Le Floch et al. [2012] acknowledged that they were unable to use nested-CV in their study, due to the fact that the computational cost required to statistically evaluate their results with a permutation test would be too high.

In order to make the method computationally feasible, and to check the reliability of the computation of the weight vector pairs, this work proposes a novel framework which uses multiple hold-out datasets. The framework selects the hyper-parameters based on train/test subsets created using random subsamples of the data, and investigates how robust is this selection for different splits of the dataset. The proposed framework addresses limitations of previous approaches, as it shows how generalisable are the significant associative effects, and finds the optimal sparsity levels for each effect with much lower computational cost than the one needed for a nested-CV framework.

By studying the effects described by the SPLS weight vector pairs, one can understand the relationships between patterns of brain anatomy and clinical variables in specific patient populations (e.g. dementia). In addition, by projecting the data onto these weights (latent space), one can try to stratify the patients in a less rigid way, which may help refine current clinical assessment tools. Despite previous applications of SPLS to study sparse associations between brain scans and clinical/demographic variables [Monteiro et al., 2015], the use of sparsity on the clinical scores is still often overlooked. Moreover, the clinical scores used are usually summary results of clinical exams. Some of these exams have several questions/tasks covering different areas of cognition, e.g. the Mini-Mental State Examination (MMSE), which is often used in patients with dementia. In some cases, it might be useful to study not how different clinical/demographic variables are associated with the brain, but how the questions/tasks from a specific clinical exam associate with the brain. Such studies might help to refine the current clinical exams, which can be achieved by finding which questions/tasks are associated with changes in brain structure. However, it is also a more challenging problem, since the information encoded in individual questions/tasks is noisier than a summarised clinical exam score.

As mentioned in Section 3.5, a closely related method to SPLS, Sparse Canonical

Correlation Analysis (SCCA), has been applied by Avants et al. [2014] to investigate correlations between structural MRI data (grey matter) and clinical variables coming from the Philadelphia Brief Assessment of Cognition (PBAC) [Avants et al., 2014]. The PBAC test contains 20 variables grouped into 5 psychometric sub-scales, which test different cognitive and behavioral/comportment deficits. The authors fitted 5 models, each one with the MRI scans as one view and the clinical variables of a specific sub-scale as the other view, using the first SCCA weight vector pair for each model. The results showed that SCCA was able to find relationships between psychometric batteries and grey matter density, claiming to be the first study to have done it [Avants et al., 2014].

The proposed SPLS framework was applied to a dementia dataset containing whole-brain structural MRI data as one view, and the scores for each individual MMSE question/task as the other view, which appears to be the first published study to do so [Monteiro et al., 2016]. Sparsity was applied to both views and no *a priori* information about the structure of the data was provided to the model, which distinguishes the present work from previous studies [Avants et al., 2014, Lin et al., 2014]. The lack of *a priori* assumptions allows the model to freely look for associations in the data without being constricted by pre-defined brain regions or sub-scales of clinical test scores. This is especially important when the condition being studied still lacks strong evidence to make assumptions regarding behaviour and its relationship with brain structure.

Finally, the performances of both PLS and SPLS were compared using the proposed framework, and two deflation strategies were compared in their ability to generate statistically significant weight vector pairs, and how well they generalise for unseen data.

6.2 Materials and Methods

6.2.1 Learning and validation framework

The proposed framework is divided into three main parts: hyper-parameter optimisation, statistical evaluation, and matrix deflation. These will be addressed in sections 6.2.1.1, 6.2.1.2, and 6.2.1.3, respectively.

6.2.1.1 Hyper-parameter optimisation

Several studies have used k-fold CV (Section 2.2) to select the optimal model hyper-parameters using the correlation between the projections as a metric [Parkhomenko et al., 2009]. Since the number of samples available in neuroimaging datasets is usually small, the natural tendency would be to use more folds, leading to an increase in train samples for each fold, and a decrease in test samples for each fold. However, this will increase the variance of the CV results [Hastie et al., 2009]. Indeed, this issue has received some recent attention in the neuroimaging literature, in which Varoquaux et al. [2017] showed that using a leave-one-out CV ($k = n$) in a classification setting lead to a very higher variance in the estimation of the model performance. Some of our initial investigations also showed that the number of folds affected the hyper-parameter estimation, i.e. the optimal hyper-parameter combination would change if the number of folds in the k-fold CV was changed. In order to overcome this limitation, the proposed framework uses an approach based on the random subsampling of the data.

The proposed framework started by removing 10% of the data randomly and keeping it as a hold-out dataset (Figure 6.1), which was used later for the statistical evaluation (Section 6.2.1.2). Then, the remaining dataset (i.e. train/test dataset) was randomly split 100 times into a train set (80% of the data) and a test set (20% of the data). For each split, the model was trained on the train set, the test data were projected onto the resulting weight vector pair, and the absolute correlation between the projections of the two views was computed, using the same correlation metric as Parkhomenko et al. [2009] (Section 3.4.1):

$$\rho_k = \left| \mathrm{Crr}\left(\boldsymbol{X}_k \boldsymbol{u}_{(-k)}, \boldsymbol{Y}_k \boldsymbol{v}_{(-k)} \right) \right|$$

where $\boldsymbol{X}_k$ and $\boldsymbol{Y}_k$ denote the test sets; and $\boldsymbol{u}_{(-k)}$ and $\boldsymbol{v}_{(-k)}$ are the weight vectors computed using the train data.

The average correlation of K splits (where $K = 100$) for a specific hyper-parameter combination $\{c_u, c_v\}$ was then computed using the arithmetic mean: $\bar{\rho}_{c_u,c_v} = \frac{1}{K} \sum_{k=1}^{K} \rho_k$. This procedure was repeated for several hyper-parameter combinations spanning the full hyper-parameter range, and the combination with the highest average correlation was selected, i.e. a grid-search was performed. The

selected hyper-parameter combination was then used to train the models in the statistical evaluation step (Section 6.2.1.2).

By doing this random subsampling procedure, both the number of correlation values (ρ_k) used to compute the average correlation ($\bar{\rho}_{c_u,c_v}$) and the size of the test datasets were increased, which should make the estimation of the average correlation per hyper-parameter combination ($\bar{\rho}_{c_u,c_v}$) more stable. Note that the same random splits were performed for each hyper-parameter combination. Also, the grid-search was performed using 40 equidistant points in $1 \leqslant c_u \leqslant \sqrt{p}$ and $1 \leqslant c_v \leqslant \sqrt{q}$, which made a total of 1600 hyper-parameter combinations. The plots showing the average absolute correlation for different hyper-parameter combinations (i.e. "hyper-parameter space") are provided in Appendix B.2.

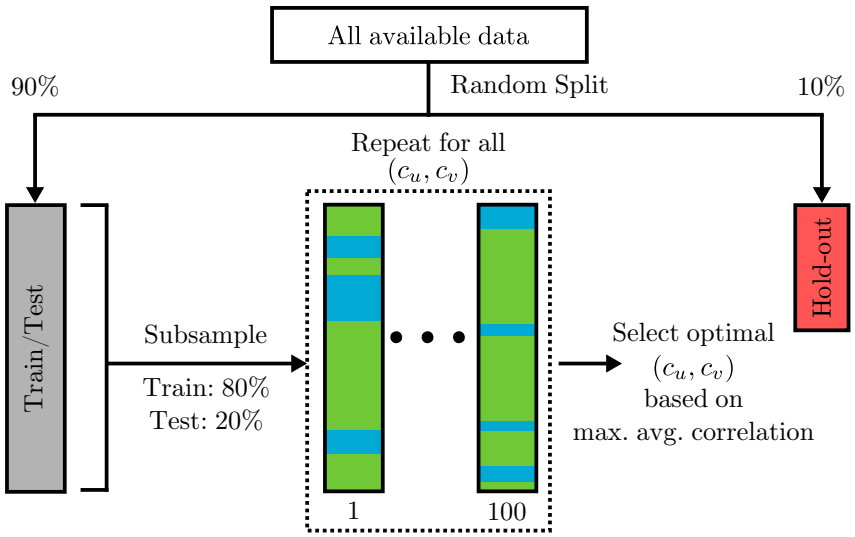


Figure 6.1: Hyper-parameter optimisation framework.

As previously mentioned, SPLS maximises the covariance between the projections, and not the correlation between the projections, the latter is maximised by using CCA. Indeed, this was the reason for using the covariance between the projections in the framework proposed in Chapter 5. However, one should keep in mind that the framework proposed in Chapter 5 selected the optimal hyper-parameter combinations by using a "distance" metric between the covariances obtained using permuted data from the covariances obtained using non-permuted data. In a train/test scheme, such as the one proposed in the present chapter, the covariance between the test data projections should not be used. When one maximises the covariance between the projections, one tries to find a sparse weight vector pair $\{\boldsymbol{u}, \boldsymbol{v}\}$, such that $\boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{Y} \boldsymbol{v}$

is maximised. Note that the more features are included in the model, the higher the covariance will be. In other words, if the covariance between the test data projections were to be used, the less sparse model would always be chosen. On the other hand, the absolute correlation between the projections is computed as:

$$\mathrm{Crr}(\boldsymbol{X}\boldsymbol{u}, \boldsymbol{Y}\boldsymbol{v}) = \left| \frac{\boldsymbol{u}\boldsymbol{X}^\intercal \boldsymbol{Y}\boldsymbol{v}}{\sqrt{\boldsymbol{u}\boldsymbol{X}^\intercal \boldsymbol{X}\boldsymbol{u}}\sqrt{\boldsymbol{v}\boldsymbol{Y}^\intercal \boldsymbol{Y}\boldsymbol{v}}} \right|$$

In this case, the increase in the number of included features will not always increase the absolute correlation, as the features will also increase $\sqrt{\boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{X}\boldsymbol{u}}$ and/or $\sqrt{\boldsymbol{v}^\intercal \boldsymbol{Y}^\intercal \boldsymbol{Y}\boldsymbol{v}}$, which in turn will decrease the absolute correlation. Therefore, the absolute correlation between the projections will be a value between 0 and 1, which will penalise models whose features increase $\sqrt{\boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{X}\boldsymbol{u}}$ or $\sqrt{\boldsymbol{v}^\intercal \boldsymbol{Y}^\intercal \boldsymbol{Y}\boldsymbol{v}}$ more than they increase $\boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{Y}\boldsymbol{v}$.

6.2.1.2 Statistical evaluation

When testing the statistical significance of an associative effect using a permutation test with a nested-CV framework, one has to re-train the model for every permutation, including the hyper-parameter optimisation step. Unfortunately, when dealing with very high-dimensional data, such as whole-brain MRI scans, this can be computationally prohibitive. In order to assess the statistical significance of the weight vector pairs without performing a hyper-parameter optimisation for each permutation, an approach which uses hold-out datasets $\{\boldsymbol{X}^*, \boldsymbol{Y}^*\}$ is proposed.

The statistical evaluation step is summarised in Figure 6.2, which started by training a model with all the train/test data using the optimal hyper-parameters selected in the previous step (Section 6.2.1.1), the hold-out data were projected onto these vectors and the absolute correlation between the projections was computed:

$$\rho = \left| \mathrm{Crr}\left(\boldsymbol{X}^*\boldsymbol{u}, \boldsymbol{Y}^*\boldsymbol{v}\right) \right| \tag{6.1}$$

where $\boldsymbol{u}$ and $\boldsymbol{v}$ are the weight vectors computed by training the model with the train/test dataset.

During the permutation, the order of the samples in one of the views was permuted while leaving the other view untouched, thereby destroying the relationship between the two views. The model was then trained again with the permuted data

and the absolute correlation between the projections was computed:

$$\rho_b = |\mathrm{Crr}\,(\boldsymbol{X}^*\boldsymbol{u}_b, \boldsymbol{Y}^*\boldsymbol{v}_b)|$$

where $\boldsymbol{u}_b$ and $\boldsymbol{v}_b$ are the weight vectors computed by training the model with the permuted train/test dataset for permutation b. The process was then repeated B times (re-training the model with the permuted dataset, and projecting the hold-out data onto the computed weight vectors).

Finally, the statistical significance of the weight vector pair was tested using the following null-hypothesis, H_s: "There is no relationship between the two views, therefore the correlation obtained with the original data is not different from the correlation obtained with the permuted data". If the probability of obtaining a correlation as large (or larger) than the original one was very low, i.e. p-value is very low, then one can reject the null-hypothesis and conclude that the found associative effect is statistically significant.

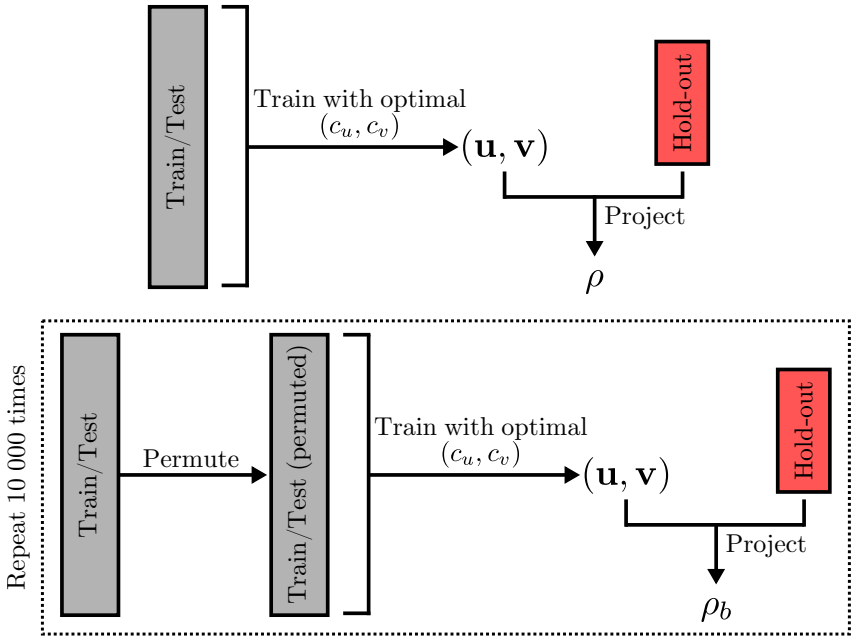


Figure 6.2: Permutation framework.

The p-value was computed using the expression described in Equation 2.16.

Due to the small sample sizes usually associated with neuroimaging datasets, the p-value may be estimated using a small hold-out dataset, which can lead to high variance in the results, depending on how the data were split. In order to make

the model estimation more robust, the proposed framework used several hold-out datasets. After the p-value was obtained, the process went back to the beginning (Section 6.2.1.1) and was repeated 9 times, which means that in the end 10 p-values were obtained: one for each random split of the hold-out dataset. A criteria was then necessary to determine if there were any statistically significant effects in the data that were being fitted by SPLS, this can be done by using the concept of the *omnibus hypothesis*. The omnibus hypothesis was previously proposed for mass-univariate statistical tests, where a statistical test is performed on each voxel j in region R, in order to test a null-hypothesis H_j. In this case, the combined hypothesis H_R over all the voxels is the following: "All the hypotheses H_j are true". This is known as the omnibus hypothesis and, as one can see, will be rejected if any of the H_j hypothesis is rejected [Holmes, 1994, Nichols and Holmes, 2001]. In the present work, the use of this concept to evaluate groups of random splits of the data is proposed. In this case, the omnibus hypothesis H_{omni} is: "All the null-hypothesis H_s are true". In other words, if any of the 10 p-values (obtained using the 10 random splits of the data) was statistically significant, then, the omnibus hypothesis would be rejected. All the p-values should be corrected for multiple comparisons by performing a Bonferroni correction, i.e. in order to have a family-wise error rate of 0.05: $\alpha = 0.05/10 = 0.005$. Therefore, the omnibus hypothesis was rejected if any of the 10 splits had $p < 0.005$.

Finally the statistically significant weight vector pair with the lowest p was selected to be used for matrix deflation (Section 6.2.1.2). In case several weight vector pairs had the same p-value, the one with the highest hold-out correlation (Equation 6.1) was selected.

6.2.1.3 Matrix deflation

If the omnibus hypothesis was rejected (Section 6.2.1.2), then the effect found by SPLS was statistically significant and needs to be removed from the data, in order to look for potential additional effects. This was done by matrix deflation, removing the effect described by the weight vector pair h before computing the next pair $(h+1)$.

As previously mentioned, Witten et al. [2009] proposed the used of the Hotelling's deflation to accomplish this. However, as shown in Chapter 4, the orthogonality property of the Hotelling's deflation does not hold. Thus, projection deflation was used instead (Equation 4.1).

As described in Section 3.3, there is another family of deflation strategies which uses the projections of the data to deflate the data matrices, and which are commonly used with PLS Mode-A, PLS1, and PLS2 (Section 3.3) [Wegelin, 2000, Rosipal and Krämer, 2006]. These type of deflation strategies have been previously used in SPLS [Waaijenborg et al., 2008, Lê Cao et al., 2008, 2009, Chun and Keleş, 2010, Chun et al., 2011, Yoshida et al., 2013], although, it was mostly used in its asymmetric version, i.e. in SPLS regression variants.

The symmetric deflation version, used in PLS Mode-A (Equation 3.13), was compared with the projection deflation approach (Equation 4.1) used in the proposed framework. As far as we are aware, there is no study comparing the results obtained using both these deflation strategies with SPLS. By performing this comparison, this thesis aims to provide some insights into how projection deflation performs compared with the most popular symmetric deflation strategies proposed for SPLS: Hotelling's deflation (Chapter 4), and PLS Mode-A deflation (this chapter). These two deflation methods are not equivalent, please refer to Appendix A.1 for the proof.

The SPLS results were also compared with the results acquired by applying the same framework with PLS. In this case, there are no hyper-parameters to optimise, thus, the step described in Section 6.2.1.1 was not performed.

6.2.2 Projection onto the SPLS latent space

The data can be projected onto the statistically significant weight vectors from both image and clinical views. As previously mentioned, these weight vectors represent the SPLS latent space. The projection of the data onto this space may bring insights about their structure, which can potentially be used for patient stratification.

6.2.3 Dataset

The data used in this chapter were obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database (`adni.loni.usc.edu`). The ADNI was launched in 2003 as a public-private partnership, led by Principal Investigator Michael W. Weiner, MD. The primary goal of ADNI has been to test whether serial Magnetic Resonance Imaging (MRI), Positron Emission Tomography (PET), other biological markers, and clinical and neuropsychological assessment can be combined to measure the progression of mild Mild Cognitive Impairment (MCI) and early Alzheimer's Disease (AD). For up-to-date information, see `www.adni-info.org`.

SPLS was applied to investigate the association between the grey matter maps and the individual scores of the questions/tasks of the MMSE, which is a quite widely used exam that is performed on patients with dementia [Folstein et al., 1975]. The dataset consisted of a subset of 592 unique subjects from the ADNI: 309 males (average age 74.68 ± 7.36) and 283 females (average age 72.18 ± 7.50). These subjects were clinically labeled as being either healthy, suffering from MCI, or suffering from AD. Further details about the demographics of the dataset can be seen in Table 6.1.

Table 6.1: Demographic information of the dataset. The gender information represents the number of males/females.

	Healthy	**MCI**	**AD**
Gender	69/78	184/151	56/54
Age	73.6279 ± 5.8357	71.6475 ± 7.7534	74.3409 ± 7.9054
MMSE	28.9728 ± 1.1642	27.7403 ± 2.0579	21.9364 ± 4.1003

The T1 weighted MRI scans were segmented into grey matter probability maps using SPM12, normalised using DARTEL [Ashburner, 2007], converted to MNI space [Mazziotta et al., 1995] with isotropic 2 mm voxels and smoothed with a Gaussian kernel with a FWHM of 2 mm. A mask was then generated, this selected voxels which had an average probability of being grey matter equal or higher than 10% for the whole dataset. This resulted in 168 130 voxels per subject being used.

Each question/task of the MMSE was encoded in the following way: the subjects were given a score of 1 if the answer was correct, or the task was performed correctly; and a score of 2 if the answer was wrong, or the task was not performed correctly. The exam is conducted by a clinician and is divided into five categories, each containing different questions/tasks, which test five different cognitive domains [Folstein et al., 1975]:

- Orientation (questions 1 to 10) — These are related with temporal and spatial orientation.
- Registration (questions 11 to 13) — The clinician names three objects and asks the patient to repeat all three. There is an extra question (13.a) in which the clinician writes down the number of trials that the subject had to take.
- Attention and Calculation (questions 14 to 18) — The subject is asked to spell the word “world” backwards (i.e. “D”, “L”, “R”, “O”, “W”). A score is

attributed for each letter, and the subject is only given a good score if the letter is in the correct order.

- Recall (questions 19 to 21) — The subject is asked to name the three objects named before (questions 11 to 13).

- Language (questions 22 to 30) — These questions/tasks involve recognising and naming objects (e.g. naming a watch and a pencil), repeating a sentence, understanding verbal commands (e.g. "take a paper with the right hand", "fold it in half", "put it on the floor"), reading, writing, and drawing.

For a detailed list of the questions/tasks, please refer to Appendix B.1. All the features in both views (image and clinical) were mean-centered and normalised to have standard deviation equal to 1.

6.3 Results

6.3.1 Statistical significance testing

Table 6.2 shows the p-values obtained by using PLS with the proposed framework, as one can see, the omnibus hypothesis H_{omni} (Section 6.2.1.2) could not be rejected, i.e. $p \geqslant 0.005$. Although the proposed framework would stop as soon as a statistically significant weight vector pair could not be found, i.e. in the first associative effect for the considered dataset, we allowed the algorithm to run until 3 associative effects were found, in order to assess how the different deflation methods behave with PLS and SPLS.

The p-values obtained with SPLS can be seen in Table 6.3. In this case, H_{omni} was rejected twice: for the first and second associative effect using projection deflation. No statistically significant results were obtained when using a PLS Mode-A deflation.

6.3.2 Generalisability of the weight vectors

Figure 6.3 shows the average absolute correlation on the 10 hold-out datasets obtained with both PLS and SPLS, using the two types of deflation. The average absolute correlation on the hold-out datasets decreased with subsequent weight vector pairs (i.e. decreased after each deflation), moreover, it seemed to be higher when PLS Mode-A deflation was applied with PLS. However, when SPLS was used, projection deflation seemed to perform better, exhibiting higher average correlation values on

Table 6.2: PLS p-values computed with 10 000 permutations. All p-values are rounded to 4 decimal places.

Split	PLS $\{\boldsymbol{u}, \boldsymbol{v}\}$ pair				
		PLS deflation		Proj. deflation	
	1	2	3	2	3
1	0.0690	0.0193	0.8619	0.4635	0.9853
2	0.2825	0.0655	0.0422	0.0323	0.3175
3	0.0120	0.2718	0.0173	0.4599	0.0609
4	0.0902	0.4255	0.4968	0.0742	0.4836
5	0.0924	0.9607	0.9855	0.9152	0.2106
6	0.0844	0.3984	0.1593	0.3412	0.5270
7	0.0866	0.1860	0.7745	0.9767	0.8342
8	0.0894	0.0479	0.1417	0.3052	0.6869
9	0.1233	0.1396	0.3932	0.3170	0.5775
10	0.0224	0.0289	0.1805	0.9831	0.7565
Rej. H_{omni}	No	No	No	No	No

Table 6.3: SPLS p-values computed with 10 000 permutations (statistically significant results are shown in bold). All p-values are rounded to 4 decimal places.

Split	SPLS $\{\boldsymbol{u}, \boldsymbol{v}\}$ pair				
		PLS deflation		Proj. deflation	
	1	2	3	2	3
1	**0.0007**	0.2476	0.3754	0.0376	0.0583
2	0.0068	0.2365	0.3585	**0.0041**	0.1769
3	**0.0002**	0.6051	0.0460	0.5298	0.5029
4	**0.0002**	0.9637	0.2013	0.0509	0.2841
5	**0.0001**	0.5711	0.9273	**0.0012**	0.3978
6	**0.0005**	0.6613	0.1107	0.0782	0.3267
7	**0.0001**	0.6073	0.3526	0.0256	0.0066
8	**0.0016**	0.9777	0.4515	0.0405	0.1126
9	**0.0004**	0.0713	0.4301	**0.0002**	0.0692
10	**0.0001**	0.1618	0.1817	0.2745	0.4399
Rej. H_{omni}	**Yes**	No	No	**Yes**	No

the hold-out datasets, and having smaller standard deviation values (which was reflected by the smaller error bars).

6.3.3 Weight vectors or associative effects

6.3.3.1 PLS

Since PLS was not able to reject the omnibus hypothesis, no weight vectors are presented in this section. For comparative purposes, the average of the weight vectors

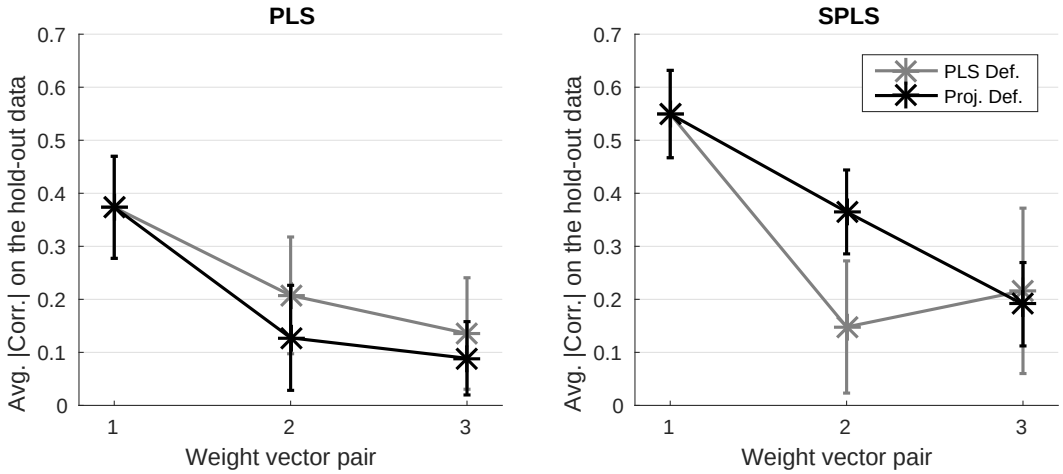


Figure 6.3: Average absolute correlation on the hold-out datasets.

for the first effect can be seen in Appendix B.3.1.

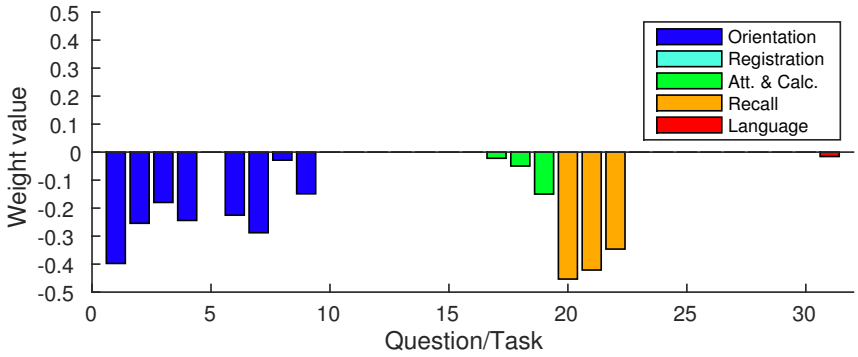
6.3.3.2 SPLS

Unlike PLS, SPLS found statistically significant sparse weight vectors, representing associative effects between clinical (Figure 6.4) and image views (Figure 6.5). This section will only present the statistically significant weight vectors, which were obtained using projection deflation. For comparative purposes, the averages of the weight vectors for the second effect using PLS Mode-A deflation and projection deflation are presented in Appendix B.3.2.

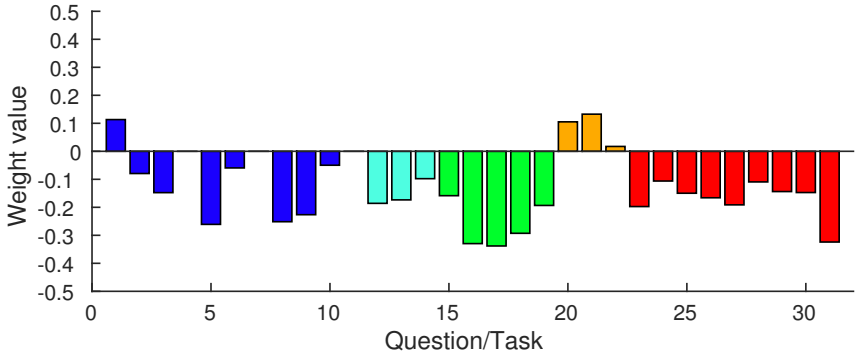
First associative effect

As previously mentioned, each weight vector pair represents a multivariate associative effect between the two views (brain voxels and clinical variables), i.e. the clinical weight vector will show a subset of clinical variables associated with a subset of brain voxels displayed in the image weight vector. Figure 6.4(a) shows the first clinical weight vector. It is possible to see that only 15 out of 31 clinical variables were selected. These belonged mainly to the "Orientation", "Attention and Calculation", and "Recall" domains. One variable was selected in the "Language" domain. The weight vector corresponding to the first image weight vector can be seen in Figure 6.5(a). As one can see, the weight map is very sparse and the regions found have been previously associated with memory (e.g. hippocampus and amygdala) [Jack et al., 2000].

Using the Automated Anatomical Labeling (AAL) atlas [Tzourio-Mazoyer et al.,



(a) First clinical weight vector.



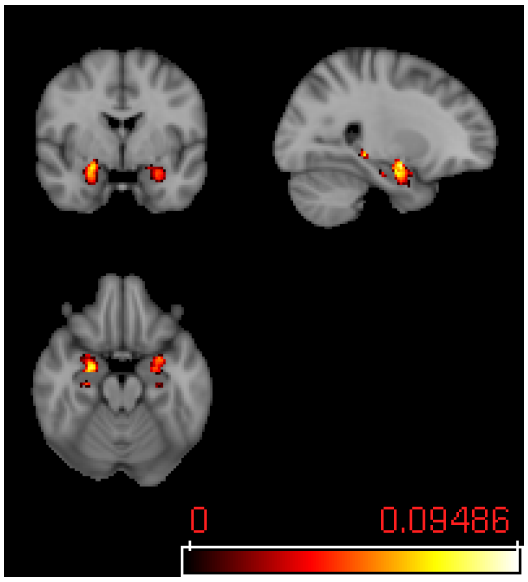
(b) Second clinical weight vector.

Figure 6.4: SPLS clinical weight vectors. The sign of the second weight vector was inverted for visualisation only (in order to be consistent with the first weight vector pair).

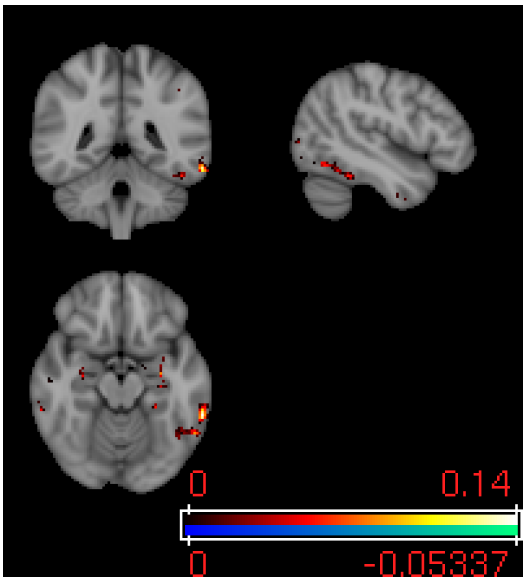
2002], it is possible to summarise the image weight vectors by ranking the regions of the atlas by their average absolute weight value. The average was used to take into account the different atlas region sizes, i.e. the larger the fraction of voxels equal to zero in a region is, the lower the average absolute weight in that region will be. Table 6.4 shows the top 10 regions for the first image weight vector. For the complete list of regions, please refer to Appendix B.4.

Second associative effect

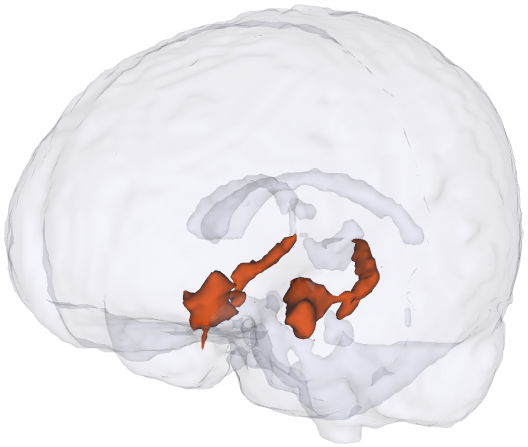
The second clinical weight vector (Figure 6.4(b)) was not as sparse as the previous one: 28 out of 31 variables were selected. The magnitudes of the weights for the "Recall" domain were substantially smaller than on the previous weight vector pair, while the absolute values of the weights on the "Registration", "Attention and Calculation", and "Language" domains were greater. The voxels found by the second image weight vector (Figures 6.5(b) and 6.5(d)) were less localised than the ones in the first image weight vector, these were present mostly in the temporal lobes,



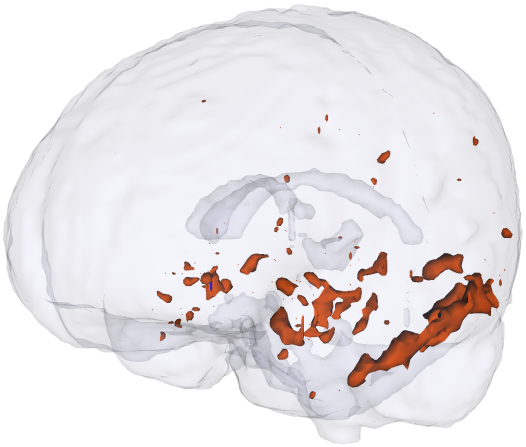
(a) First image weight vector.



(b) Second image weight vector.



(c) 3D visualisation of the features selected for the first image weight vector.



(d) 3D visualisation of the features selected for the second image weight vector.

Figure 6.5: SPLS image weight vectors. Red regions denote positive weights and blue regions denote negative weights (very small region on the second weight vector). The sign of the second weight vector was inverted for visualisation purposes only (in order to be consistent with the first weight vector pair).

hippocampus, and amygdala. The second associative effect seems to capture an association between all domains of the MMSE score and mainly temporal regions in the left brain hemisphere.

The top 10 regions for the second image weight vector can be seen in Table 6.5. For the complete list of regions, please refer to Appendix B.4.

Note that most voxels in Figures 6.5(a) and 6.5(b) have positive weights, while most entries of the corresponding clinical weight vector have negative signs (Fig-

Table 6.4: Top 10 atlas regions for the first image weight vector.

Atlas Region	# voxels found
Amygdala_L	98
Amygdala_R	90
Hippocampus_R	175
Hippocampus_L	152
ParaHippocampal_R	92
ParaHippocampal_L	44
Lingual_L	9
Precuneus_L	2
Precuneus_R	1
Temporal_Pole_Sup_L	1

Table 6.5: Top 10 atlas regions for the second image weight vector.

Atlas Region	# voxels found
Amygdala_L	36
Temporal_Inf_L	292
Hippocampus_L	88
Amygdala_R	11
ParaHippocampal_L	53
Fusiform_L	78
Temporal_Inf_R	64
Hippocampus_R	22
Occipital_Inf_L	12
Temporal_Mid_L	76

ures 6.4(a) and 6.4(b)). This means that both effects follow the same tendency: high grey matter density (high image weights) are associated with generally low values in the clinical questions/tasks (i.e. the task was performed correctly, Section 6.2.3), and *vice versa*.

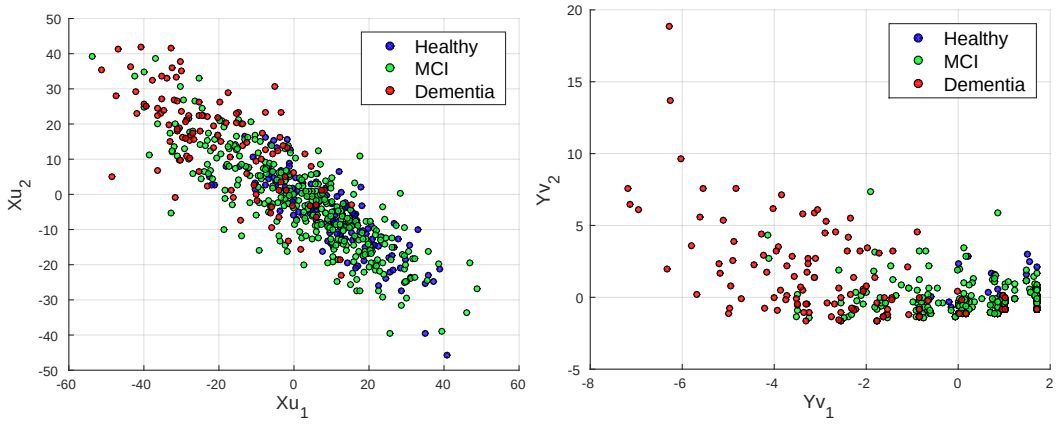
6.3.4 Projection onto the SPLS latent space

The data were projected onto the weight vector pairs computed using SPLS (Figures 6.4 and 6.5), in order to bring insights about structure in the data, and to potentially stratify patients (Section 6.2.2). Since PLS was not able to find statistically significant weight vector pairs, the projections for this method will not be presented.

Figure 6.6(a) shows the projection of the data onto both SPLS image weight vectors, while Figure 6.6(b) shows the projection of the data onto both SPLS clinical weight vectors. Each point represents the projection of one subject's data onto the

subspace defined by the weight vector pair, where its color is given based on the clinical diagnosis. The horizontal axes ($\boldsymbol{X}\boldsymbol{u}_1$ and $\boldsymbol{Y}\boldsymbol{v}_1$) correspond to the projections onto the first weight vector pair, and the vertical axes ($\boldsymbol{X}\boldsymbol{u}_2$ and $\boldsymbol{Y}\boldsymbol{v}_2$) correspond to the projections onto the second weight vector pair. For the plots showing the projections per weight vector pair ($\{\boldsymbol{X}\boldsymbol{u}_1, \boldsymbol{Y}\boldsymbol{v}_1\}$ and $\{\boldsymbol{X}\boldsymbol{u}_2, \boldsymbol{Y}\boldsymbol{v}_2\}$), please refer to Appendix B.5.

As one can see, there were no defined clusters, however, there seems to be a continuous distribution of subjects from lower to higher degrees of neurodegeneration.



(a) Projection of the image data onto the image weights $\{\boldsymbol{u}_1, \boldsymbol{u}_2\}$.

(b) Projection of the clinical data onto the clinical weights $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$.

Figure 6.6: Projection of the data onto the SPLS weight vector pairs.

6.4 Discussion

The results show that the proposed SPLS framework was able to detect two statistically significant associative effects between grey matter maps and individual questions/tasks of the MMSE score when using sparsity constraints in both views. These results were particularly interesting as the information encoded on the individual question/task level is very noisy, however, it also expresses more subtle effects in the data when compared with a summarised final exam score. The first effect captured an association mainly between the "Orientation", "Attention and Calculation" and the "Recall" domains on the clinical view, and brain regions such as the amygdala and hippocampus. The second effect captured an association between most clinical variables, and regions mainly on the left brain hemisphere, including temporal regions. These results were achieved by imposing sparsity in both views, and without using any *a priori* assumption regarding data structure, which might be

useful when it is not possible to have one. Moreover, the projection of the subjects onto the latent SPLS space showed a consistent distribution of the subjects from lower to higher degrees of neurodegeneration.

Projection deflation has shown to provide more reliable weight vectors when compared with the commonly used PLS Mode-A deflation method. When comparing the different deflation approaches, the results showed that only by using projection deflation was it possible to find a second statistically significant associative effect with SPLS. Moreover, projection deflation provided a higher average correlation on the hold-out datasets, i.e. the model generalised better for unseen data.

The proposed framework was also tested with PLS. The results showed that SPLS performed better than PLS, not only by being able to find statistically significant associative effects, but also by improving the interpretability of the weight vector pairs due to their sparsity, and by generalising better for unseen data (which can be demonstrated by an increase in average correlation obtained in hold-out datasets).

6.4.1 Multiple hold-out framework

In this study, a SPLS framework which uses multiple random splits for the hold-out dataset was proposed. By performing a significance test on each random split, the framework checked how reliable the weight vector computation is to data perturbations, making it more robust than approaches based on a single hold-out split [Labus et al., 2015].

The estimation of the sparsity levels for both views without *a priori* assumptions allows for greater flexibility when trying to find the best model to describe a particular associative effect in the data. Moreover, these levels were not fixed for every weight vector pair, which means that each associative effect will be described by the right level of sparsity in each view, i.e. the proposed approach will find the necessary number of voxels and clinical variables to describe each associative effect.

One of the main advantages of the proposed framework when compared with a more widespread nested-CV approach is its computational time. Nested-CV consists in performing a two level CV (Section 2.2.2), where, for each train fold, an inner CV procedure is performed for every hyper-parameter combination, in order to select the optimal hyper-parameter combination to be applied in the outer fold (Section 2.2). This is a quite computationally intensive procedure, since hyper-parameter selection

will have to be repeated for each permutation during the statistical evaluation. Even if nested-CV was applied to this problem with a small number of folds (5 inner folds and 5 outer folds) and permutations (1000), it would require 40 045 005 SPLS computations, whereas the proposed framework with 100 subsamples (which should provide a more stable hyper-parameter selection) and 10 000 permutations, computed SPLS 1 700 010 times, which corresponds to approximately 4% of the number of computations that would have been necessary with a nested-CV framework. For the details of how these values are calculated, please refer to Appendix B.6.

There are other approaches in the literature for selecting the level of sparsity based on stability criteria [Lê Cao et al., 2011, Nybo et al., Labus et al., 2015]. These consist of fixing the sparsity hyper-parameters, and then computing the SPLS weight vector pairs multiple times with many subsamples of the data, in order to select the features which are stable across the several splits. This procedure is very computationally expensive, due to the fact that it has to be repeated for every CV fold in every permutation. For example, even with a common validation procedure using 1000 subsamples, a 5-fold CV (just like the approach by Nybo et al.), and a 1000 permutation test, one would have to perform SPLS at least 5 000 000 times (as opposed to the 1 700 010 times required by the multiple hold-out framework).

Witten and Tibshirani [2009] proposed a hyper-parameter optimisation procedure based on permutations where, for each hyper-parameter combination, the p-value of the correlation using all the data was computed and the combination with the lowest p was selected. This method will choose the hyper-parameters for which the distance between the correlation computed with the non-permuted data and the null distribution of the correlations is the largest, however, this might not be the same hyper-parameters that maximise the correlation between the projections using test data, which is what the proposed subsampling approach will try to achieve (Section 6.2.1.1). Although a permutation based framework would require the computation of SPLS slightly less times than the proposed framework (1 600 000 times per weight vector pair), there is no guarantee that the actual computational time would be less, as some of our earlier experiments indicated that SPLS takes longer to converge when computing weight vector pairs with permuted data.

6.4.2 Statistical significance testing

SPLS with projection deflation was able to find two statistically significant associative effects in the data, while PLS was not able to find any. This result may be related to the fact that SPLS provided a higher average hold-out correlation (Figure 6.3). The sparsity constraints used by SPLS allowed the method to exclude noisy features, which resulted in a model with a better ability to generalise for unseen data.

6.4.3 Comparison between deflation approaches

Projection deflation performed substantially better when applied to SPLS than the commonly used PLS Mode-A deflation, being able to provide higher values of average correlation on the hold-out datasets (Figure 6.3).

The role of the deflation step is to remove the associative effect expressed in the obtained weight vector pair from the data matrices. As previously mentioned, both PLS and SPLS capture the strongest effect in the data with the first weight vector pair, after the covariance explained by this vector pair is removed, the signal-to-noise ratio in the data will decrease. This property allows the effects to be ranked, since each weight vector pair will explain more covariance in the data than the following ones.

These results reinforce the conclusion from Chapter 4 [Monteiro et al., 2014]: using deflation strategies inherited from non-sparse methods might not be the best approach when dealing with sparse methods, these algorithms should have their own appropriate deflation steps.

6.4.4 Multivariate associative effects

The first SPLS clinical weight vector is particularly interesting (Figure 6.4(a)). The domains with the most prominent contribution to the weight vector were consistent with what it would be expected from a population with AD patients. All the questions in the "Recall" domain were chosen, while none of the questions in the "Registration" domain were selected. This suggests that whether patients remember words that were recently presented to them explains more covariance in the data than whether they repeat the words when they are first presented (which is reflected by the "Registration" domain). This was expected, since the inability to remember new information is one of the earliest and most prominent symptoms of AD [Galton et al., 2001]. The regions associated with these variables, displayed by the image weight

vector (Figures 6.5(a) and 6.5(c), and Table 6.4), have been previously described as being associated with dementia [Jack et al., 2000, Chan et al., 2001, Galton et al., 2001]. Moreover, Table 6.4 and Figure 6.5(c) show that there is symmetry in the regions selected (e.g. hippocampus, parahippocampal gyrus, and amygdala), which is consistent with the symmetric brain region atrophy that has been described in the AD literature [Chan et al., 2001, Galton et al., 2001].

Another interesting result is that only the last three variables (out of five) in the "Attention and Calculation" domain were selected, and with increasing absolute weight. During this part of the MMSE, the subject is trying to spell the word "world" backwards, each variable corresponds to whether the subject replied with the correct letter or not. The results suggest that there might be some effect during this task that is dependent on the order of the letters.

Unlike previous studies [Avants et al., 2014, Lin et al., 2014], no *a priori* assumptions regarding the structure of the data were used. The exclusion of virtually two entire domains ("Registration" and "Language") was purely data-driven. Also, no information regarding brain structure was provided to the algorithm, e.g. region information based on a brain atlas.

For the second weight vector pair, it is interesting to see that most of the top regions are in the left side of the brain. Previous studies comparing AD with Semantic Dementia (SD), found that SD is characterised by a greater atrophy of structures on the left side of the brain (compared with the corresponding structures on the right side), particularly, the temporal lobe, parahippocampal gyrus, and fusiform gyrus [Chan et al., 2001, Galton et al., 2001]. Patients with SD exhibit impairment of semantic memory, e.g. patients have difficulties with word meaning [Chan et al., 2001]. SPLS selected these regions when the questions/tasks from the "Language" domain were included in the model (Figure 6.4(b)), which suggests that the second weight vector pair is picking up effects primarily related with semantics, while the first is picking up effects primarily related with memory.

6.4.5 Projection onto the SPLS latent space

The projection of the subjects' data onto the SPLS weight vectors (Figure 6.6) shows that, although there are no defined clusters, the subjects seem to form a continuous distribution from healthier subjects, to progressively worse cases of neurodegeneration.

It is also interesting to note that on the projections of the subjects onto the two clinical weight vectors (Figure 6.6(b)), the second clinical weight vector does not seem to separate subjects based on diagnosis as well as the first, which suggests that the effect detected is less directly associated with the clinical labels.

6.4.6 Limitations and future work

The proposed framework should provide more reliable results, by evaluating multiple splits of the data. However, despite the promising results that were obtained with this framework, one of its possible limitations is the fact that the statistical evaluation step, using 10 different splits of the data with a Bonferroni correction for multiple comparisons, is rather strict, which might lead to some false negative results. Therefore, the framework may require large amounts of data in order to detect statistically significant associative effects, which may not be easily available. Nevertheless, the current trend in neuroscience is to invest in projects which collect large amounts of data [Thompson et al., 2014, Sudlow et al., 2015, nsp], which means that exploratory approaches, such as the one proposed, may become more relevant in the future.

In future work, it would be interesting to apply this framework to data in which clinical categories have shown to have limitations, such as, psychiatric data [Insel et al., 2010]. In these cases, the use of SPLS to find associative effects between brain images and individual exam questions/tasks may provide valuable information that could help with patient stratification.

6.5 Conclusion

The results presented in this chapter represent a proof of concept that the proposed multiple hold-out framework is able to find meaningful multivariate associative effects. This framework was used to find associations between the individual items from the MMSE and whole-brain grey matter maps, which had not been previously demonstrated [Monteiro et al., 2016]. Furthermore, the comparison of two deflation methods (projection deflation vs. PLS Mode-A deflation) showed that projection deflation performed better in this dataset, which provides further evidence that this should be the preferred deflation method when using SPLS in an exploratory setting.

Chapter 7

Alternating Least Squares (ALS) method for SCCA and SPLS

Although SPLS has shown to provide good results using the multiple hold-out framework (Chapter 6), one of its possible limitations is that the correlation between the projections is used as a metric, even though SPLS maximises the covariance between the projections. This means that the model may under-perform, due to the fact that the hyper-parameters are chosen based on a different performance metric. One of the ways to address this problem would be to adopt the permutation based framework proposed in Chapter 5, however, this has shown to have several limitations (Chapter 6). Another solution to the problem would be to use a sparse eigen-decomposition method which maximises the correlation between the projections, such as, Sparse Canonical Correlation Analysis (SCCA). Note that the under-performance issue is not specific to the multiple hold-out framework, but to any SPLS framework which uses the correlation as a metric [Witten et al., 2009, Witten and Tibshirani, 2009, Parkhomenko et al., 2009, Lin et al., 2014, Grellmann et al., 2015].

There have been several extensions of CCA using different regularisation constraints, however, each one takes a slightly different approach to solve the corresponding optimisation problem, which means that there is no unified framework to perform CCA with different regularisation constraints. Some of these methods include: solving a standard eigenvalue problem [Hotelling, 1936], solving a generalised eigenvalue problem [Bach and Jordan, 2002], using SVD [Healy, 1957], and using gradients [Fu et al., 2016]. For an extensive review of some of these CCA methods, please refer to the paper by Uurtio et al. [(accepted)].

This chapter will explore the Alternating Least Squares (ALS) method as a general framework to solve several CCA problems with different regularisation constraints. The ALS approach for CCA was popularised partially by Golub and Zha [1992], although the idea is older [van der Burg, 1988]. Indeed, this has been used in the literature to solve different CCA problems [Lykou and Whittaker, 2010, Chi et al., 2013, Wilms and Croux, 2015, Polajnar, 2015], although not always explicitly mentioned.

The ALS consists of solving the CCA problem by alternately performing two regression steps: one to compute $\boldsymbol{u}$ by having the projection $\boldsymbol{Yv}$ as a target, and another to compute $\boldsymbol{v}$ by having the projection $\boldsymbol{Xu}$ as a target. One of the main advantages of the ALS is its flexibility, by changing the regression steps, one can use this approach to solve different kinds of CCA problems with different types of penalties in a flexible way.

This chapter will serve mainly as a background introduction to the novel SCCA method proposed in Chapter 8. However, it does describe two contributions, which are presented in two separate sections. The first contribution (Section 7.1) is an adaptation of the ALS algorithm which forces it to converge in situations where it would otherwise oscillate between a set of similar solutions. The second contribution (Section 7.2) is a comparison between seven CCA and PLS formulations:

- non-penalised CCA;
- non-penalised PLS;
- SCCA solved using ALS with the LASSO;
- SCCA solved using ALS with the elastic net;
- SPLS solved using the power method (as in Chapters 4, 5, and 6);
- SPLS solved using ALS with the LASSO;
- SPLS solved using ALS with the elastic net;

The use of the power method and the ALS to solve SPLS have been previously proposed in the literature [Witten et al., 2009, Chi et al., 2013]. However, as far as we are aware, no comparisons between their performances have been made.

7.1 SCCA using ALS

7.1.1 Materials and Methods

As described in Chapter 3, CCA tries to solve the following optimisation problem:

$$\begin{aligned} &\underset{\boldsymbol{u},\boldsymbol{v}}{\text{maximise}} \ \boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{Y} \boldsymbol{v} \\ &\text{subject to } \boldsymbol{u}^\intercal \boldsymbol{X}^\intercal \boldsymbol{X} \boldsymbol{u} = 1 \text{ and } \boldsymbol{v}^\intercal \boldsymbol{Y}^\intercal \boldsymbol{Y} \boldsymbol{v} = 1 \end{aligned}$$

Note that the CCA optimisation problem can also be written in terms of the projections of the data, i.e. one is trying to find the projections $\boldsymbol{\xi} = \boldsymbol{X}\boldsymbol{u}$ and $\boldsymbol{\omega} = \boldsymbol{Y}\boldsymbol{v}$ which solve the following:

$$\begin{aligned} &\underset{\boldsymbol{\xi},\boldsymbol{\omega}}{\text{maximise}} \ \langle \boldsymbol{\xi}, \boldsymbol{\omega} \rangle \\ &\text{subject to } \|\boldsymbol{\xi}\|_2^2 = 1 \text{ and } \|\boldsymbol{\omega}\|_2^2 = 1 \end{aligned}$$

which is equivalent to maximising the cosine of the angle between two unit vectors ($\boldsymbol{\xi}$ and $\boldsymbol{\omega}$), i.e. minimising the angle between the projections [Björck and Golub, 1973, Uurtio et al., (accepted)]. This means that the CCA problem can be re-written as:

$$\begin{aligned} &\underset{\boldsymbol{u},\boldsymbol{v}}{\text{minimise}} \ \|\boldsymbol{X}\boldsymbol{u} - \boldsymbol{Y}\boldsymbol{v}\|_2^2 \\ &\text{subject to} \\ &\|\boldsymbol{X}\boldsymbol{u}\|_2^2 = 1, \quad \|\boldsymbol{Y}\boldsymbol{v}\|_2^2 = 1 \end{aligned} \tag{7.1}$$

The problem expressed in Equation 7.1 can be solved by fixing $\boldsymbol{u}$ and solving it for $\boldsymbol{v}$, and *vice versa*, until it converges to a solution. This is known as the ALS, which is described in (Algorithm 7.1) [Golub and Zha, 1992].

The ALS algorithm is a straightforward approach, which consists of iteratively estimating each weight vector $\boldsymbol{u}$ and $\boldsymbol{v}$ by performing alternating least squares regression steps, using the projection of the opposite view as a target. The solutions are re-scaled in steps 6 and 9 in order to obey the l_2-norm constraints expressed in Equation 7.1.

As previously mentioned, the use of exploratory sparse methods may improve the interpretability of the results, by removing features which do not encode relevant information. This property can be included in CCA as well, which gives rise to a

Algorithm 7.1 Alternating Least Squares (ALS) algorithm to solve the optimisation problem expressed in (7.1).

1: Initialise $\boldsymbol{u} \in \mathbb{R}^{p\times 1}$ as a random vector.
2: Initialise $\boldsymbol{v} \in \mathbb{R}^{q\times 1}$ as a random vector.
3: **repeat**
4: $\quad \boldsymbol{\omega} \leftarrow \boldsymbol{Y}\boldsymbol{v}$
5: $\quad$ Solve the least-squares problem: $\boldsymbol{u} \leftarrow \arg\min_{\boldsymbol{u}} \|\boldsymbol{\omega} - \boldsymbol{X}\boldsymbol{u}\|_2^2$
6: $\quad \boldsymbol{u} \leftarrow \boldsymbol{u}/\|\boldsymbol{X}\boldsymbol{u}\|_2$ ▷ Normalise $\boldsymbol{u}$ so that it obeys the constraint
7: $\quad \boldsymbol{\xi} \leftarrow \boldsymbol{X}\boldsymbol{u}$
8: $\quad$ Solve the least-squares problem: $\boldsymbol{v} \leftarrow \arg\min_{\boldsymbol{v}} \|\boldsymbol{\xi} - \boldsymbol{Y}\boldsymbol{v}\|_2^2$
9: $\quad \boldsymbol{v} \leftarrow \boldsymbol{v}/\|\boldsymbol{Y}\boldsymbol{v}\|_2$ ▷ Normalise $\boldsymbol{v}$ so that it obeys the constraint
10: **until** convergence
11: **return** $\boldsymbol{u}$, $\boldsymbol{v}$

type of method known as Sparse CCA (SCCA). By using other regression functions in steps 5 and 8 of Algorithm 7.1, one can adapt ALS to solve different types of CCA, including sparse formulations.

The CCA problem described in Equation 7.1 was adapted to a SCCA optimisation problem:

$$\begin{gathered} \underset{\boldsymbol{u},\boldsymbol{v}}{\text{minimise}} \ \|\boldsymbol{X}\boldsymbol{u} - \boldsymbol{Y}\boldsymbol{v}\|_2^2 \\ \text{subject to} \\ \|\boldsymbol{X}\boldsymbol{u}\|_2^2 = 1, \quad \|\boldsymbol{Y}\boldsymbol{v}\|_2^2 = 1, \quad |I_u| \leqslant p_u, \quad |I_v| \leqslant q_v \end{gathered} \tag{7.2}$$

where $I_u := \{j \mid u_j \neq 0\}$, $I_v := \{j \mid v_j \neq 0\}$, $p_u \in \{1, 2, \ldots, p\}$ and $q_v \in \{1, 2, \ldots, q\}$. The inequalities in Equation 7.2 define an upper bound on the number of features included in the model, i.e. the number of features selected by $\boldsymbol{u}$ and $\boldsymbol{v}$.

The problem expressed in Equation 7.2 can be solved by substituting the non-penalised regressions (Steps 5 and 8) in Algorithm 7.1, by LASSO regression steps (Equation 2.9). However, since the solutions are sparse, each update in steps 5 and 8 causes features to be abruptly removed from the model, which in practice led to oscillations of the algorithm, not allowing it to converge. One of the ways to avoid these oscillations, is by gradually updating the weight vectors, taking several steps to shrink the weights towards zero, as opposed to abruptly removing the features from the model in a single step. In order to obtain a smoother convergence to the solution, the addition of an `update` step (Algorithm 7.2) to the ALS is proposed. Instead of directly updating the weights by the result of the latest regression step

$\boldsymbol{w}^{i+1}$, the `update` function takes the weight vector from the previous iteration $\boldsymbol{w}^i$ and updates it towards the new weight vector $\boldsymbol{w}^{i+1}$. Note that if an entry j of the latest regression step w_j^{i+1} is equal to 0, and the same entry from the previous weight vector $|w_j^i| < \epsilon$ (Step 3), where ϵ is a very small number, then the feature will be removed during the weight update (Step 6). The use of this function is inspired by gradient based algorithms for CCA [Fu et al., 2016], and constitutes the contribution of this section.

Algorithm 7.2 Proposed update weight function.

1: **function** UPDATE($\boldsymbol{w}^{(i+1)}, \boldsymbol{w}^{(i)}, \boldsymbol{M}, \delta$)
2: $\quad \boldsymbol{w}^{(i+1)} = \boldsymbol{w}^{(i+1)} / \|\boldsymbol{M}\boldsymbol{w}^{(i+1)}\|_2$
3: $\quad I_w := \left\{ j \mid w_j^{(i+1)} = 0 \bigwedge |w_j^{(i)}| < \epsilon \right\}$
4: $\quad \Delta\boldsymbol{w} = \delta \left(\boldsymbol{w}^{(i+1)} - \boldsymbol{w}^{(i)} \right)$
5: $\quad \boldsymbol{w}^{(i+1)} \leftarrow \boldsymbol{w}^{(i)} + \Delta\boldsymbol{w}$
6: $\quad w_j^{(i+1)} = 0,\ j \in I_w$
7: $\quad \boldsymbol{w}^{(i+1)} = \boldsymbol{w}^{(i+1)} / \|\boldsymbol{M}\boldsymbol{w}^{(i+1)}\|_2$
8: $\quad$ **return** $\boldsymbol{w}^{(i+1)}$
9: **end function**

The proposed ALS approach is described in Algorithm 7.3. The `glmnet` package was used to solve the LASSO regression steps [Friedman et al., 2010, Qian et al., 2013], and the convergence criteria was met when all the following conditions were true:

1. $\|\boldsymbol{u}^{(i+1)} - \boldsymbol{u}^{(i)}\|_2 < 10^{-5}$
2. $\|\boldsymbol{v}^{(i+1)} - \boldsymbol{v}^{(i)}\|_2 < 10^{-5}$
3. $|I_u| \leqslant p_u$
4. $|I_v| \leqslant q_v$

Moreover, the vectors $\boldsymbol{u}^{(0)}$ and $\boldsymbol{v}^{(0)}$ were initialised using the first singular vectors of $\boldsymbol{X}^\intercal\boldsymbol{Y}$, in order to provide a good first estimate of the weight vectors.

Our initial investigations performed with the proposed ALS approach revealed that the algorithm still tended to oscillate in certain conditions, even with the introduction of the `update` function (Algorithm 7.2). This was most common on the last iterations of ill-conditioned problems, i.e. when $\|\boldsymbol{u}^{(i+1)} - \boldsymbol{u}^{(i)}\|_2 \approx 10^{-4}$ and $\|\boldsymbol{v}^{(i+1)} - \boldsymbol{v}^{(i)}\|_2 \approx 10^{-4}$. This might be an indication that ALS tends to oscillate

around very similar solutions, this was indeed observed in several cases, e.g. the weight vectors would periodically change within a small set of very similar vectors. In order to avoid oscillations due to a very large step size δ in Algorithm 7.2, an option to update the step size was introduced (Step 11 in Algorithm 7.3). The ALS was considered to be oscillating if all the following conditions where met:

1. $\|\boldsymbol{u}^{(i+1)} - \boldsymbol{u}^{(i)}\|_2 < 10^{-3}$
2. $\|\boldsymbol{v}^{(i+1)} - \boldsymbol{v}^{(i)}\|_2 < 10^{-3}$
3. $|I_u| \leqslant p_u$
4. $|I_v| \leqslant q_v$
5. the correlation between the projections of the train data $\mathrm{Crr}(\boldsymbol{X}\boldsymbol{u}^{(i+1)}, \boldsymbol{Y}\boldsymbol{v}^{(i+1)})$ is approximately equal (the difference is less than 10^{-6}) to a correlation observed in past iterations.

Algorithm 7.3 SCCA using ALS. The `update` function is described in Algorithm 7.2.

1: Set $\boldsymbol{u}^{(0)}$ and $\boldsymbol{v}^{(0)}$ equal to the first singular vector pair of $\boldsymbol{X}^\intercal\boldsymbol{Y}$.
2: Set $\delta = 0.5$
3: **repeat**

4: $\quad\boldsymbol{b} \leftarrow \boldsymbol{Y}\boldsymbol{v}^{(i)}$
5: $\quad$Solve $\boldsymbol{u}^{(i+1)} \leftarrow \arg\min_{\boldsymbol{u}} \|\boldsymbol{b} - \boldsymbol{X}\boldsymbol{u}\|_2^2 + \gamma\|\boldsymbol{u}\|_1$, with γ such that $|I_u| \approx p_u$
6: $\quad\boldsymbol{u}^{(i+1)} \leftarrow \textsc{update}(\boldsymbol{u}^{(i+1)}, \boldsymbol{u}^{(i)}, \boldsymbol{X}, \delta)$

7: $\quad\boldsymbol{b} \leftarrow \boldsymbol{X}\boldsymbol{u}^{(i+1)}$
8: $\quad$Solve $\boldsymbol{v}^{(i+1)} \leftarrow \arg\min_{\boldsymbol{v}} \|\boldsymbol{b} - \boldsymbol{Y}\boldsymbol{v}\|_2^2 + \gamma\|\boldsymbol{v}\|_1$, with γ such that $|I_v| \approx q_v$
9: $\quad\boldsymbol{v}^{(i+1)} \leftarrow \textsc{update}(\boldsymbol{v}^{(i+1)}, \boldsymbol{v}^{(i)}, \boldsymbol{Y}, \delta)$

10: $\quad$**if** ALS is oscillating **then**
11: $\quad\quad\delta \leftarrow \delta/2$
12: $\quad$**end if**

13: $\quad i \leftarrow i + 1$
14: **until** convergence
15: **return** $\boldsymbol{u}$, $\boldsymbol{v}$

Note that there are other ALS formulations proposed in the literature, including methods which penalise the covariance matrices $\boldsymbol{X}^\intercal\boldsymbol{X}$ and $\boldsymbol{Y}^\intercal\boldsymbol{Y}$, such as the one proposed by Chi et al. [2013]. In this case, the update of $\boldsymbol{u}$ would be performed as

follows (an analogous update would be performed for $\boldsymbol{v}$):

$$\boldsymbol{u} \leftarrow \arg\min_{\boldsymbol{u}} \frac{1}{2}\|\widetilde{\boldsymbol{C}}_{xx}^{-1/2}\boldsymbol{X}^\intercal\boldsymbol{Y}\boldsymbol{v} - \widetilde{\boldsymbol{C}}_{xx}^{-1/2}\boldsymbol{u}\|_2^2 + \gamma\|\boldsymbol{u}\|_1$$

with

$$\widetilde{\boldsymbol{C}}_{xx} = \lambda_x m_x \boldsymbol{I} + (1-\lambda_x)\boldsymbol{C}_{xx}$$

where $\boldsymbol{C}_{xx} = \boldsymbol{X}^\intercal\boldsymbol{X}$, m_x is the average eigenvalue of $\boldsymbol{C}_{xx}$, and $\lambda_x \in [0,1]$ is a mixing coefficient which shrinks $\boldsymbol{C}_{xx}$ towards $m_x\boldsymbol{I}$. The latter appears to be analogous to the parameter τ_x in the classical regularised CCA formulation (Equation 3.8). However, this means that two extra hyper-parameters are included in the model (λ_x and λ_y), which would result in more hyper-parameters that would have to be optimised. Moreover, this formulation also requires the computation $\boldsymbol{X}^\intercal\boldsymbol{X}$, which might be expensive in some settings. Therefore, the ALS method proposed in this section (Algorithm 7.3) is based on simpler ALS approaches for SCCA previously proposed in the literature [Wilms and Croux, 2015, Polajnar, 2015].

In this section, three algorithms were compared: ALS in its simplest version (without the `update` steps); ALS using the `update` steps, but without updating δ; and ALS using both the `update` steps and updating the value of δ.

An upper limit on the number of iterations was set, i.e. if the algorithm did not converge after 1000 iterations, it was considered as "not converged".

7.1.1.1 Dataset

The first experiments performed with SCCA using ALS revealed that the `glmnet` package does not scale very well with the number of features. This means that it was not possible to run ALS on very high dimensional datasets (e.g. whole-brain voxel-wise neuroimaging data), as the computational time was too large. Other software packages where tested in early preliminary experiments (e.g. the `lars` [Efron et al., 2004]), however, it was clear that the computational time increased with the number of features as well. The best performing package was `glmnet`, this observation aligns with earlier publications suggesting that `glmnet` performs better in terms of computational time [Friedman et al., 2010]. Therefore, all the experiments in Chapters 7 and 8 were run on a lower dimensional dataset. Note that the focus of this section is on empirically assessing the convergence properties of the ALS

algorithm, its scalability is outside the scope of this section.

A subset of 701 subjects at baseline from the ADNI dataset was used for this chapter. The demographic information is expressed in Table 7.1.

Table 7.1: Demographic information of the dataset. The gender information represents the number of males/females.

	Healthy	**MCI**	**AD**
Gender	117/109	154/136	94/91
Age	75.86 ± 5.05	74.84 ± 7.44	75.20 ± 7.38

A total of 145 ROI volumes taken from the UPENN Section of Biomedical Image Analysis (SBIA) volumes were used as features in $\boldsymbol{X}$, a complete list can be found in Appendix C.1.3. A subset of 7 clinical/demographic variables was used in $\boldsymbol{Y}$: Age, Education, ADAS11 [Mohs, 1994], ADAS13 [Petersen et al., 2005], Mini-Mental State Examination (MMSE) [Folstein et al., 1975], RAVALT_immediate [Schmidt et al., 1996], and Functional Assessment Questionnaire (FAQ) [Pfeffer et al., 1982]. ADAS11 and ADAS13 denote two variants of the Alzheimer's Disease Assessment Scale (ADAS) [Rosen et al., 1984], and RAVALT_immediate denotes the sum of the scores from the 5 first trials of the Rey Auditory Verbal Learning Test (RAVALT) [Schmidt et al., 1996, Moradi et al., 2016]. For more information, please refer to the ADNI manual [adn, 2006]. These clinical scores were chosen due to the fact that they provided a fair amount of information regarding the cognitive ability of the subjects, and did not have any missing data.

In this section, the performances of the ALS algorithms were investigated using two settings: a well-conditioned problem ($p, q < n$), and an ill-conditioned problem ($p, q > n$). For the ill-conditioned problem, a dataset $\{\boldsymbol{X}', \boldsymbol{Y}'\}$ was created by adding 1000 columns of uncorrelated Gaussian noise to each data matrix:

$$\boldsymbol{X}' = [\boldsymbol{X}, \boldsymbol{\mathcal{E}}_x] \quad \text{and} \quad \boldsymbol{Y}' = [\boldsymbol{Y}, \boldsymbol{\mathcal{E}}_y]$$

where $\boldsymbol{\mathcal{E}}_x \in \mathbb{R}^{n \times 1000}$ and $\boldsymbol{\mathcal{E}}_y \in \mathbb{R}^{n \times 1000}$ are matrices where each column contains randomly generated numbers following a Gaussian distribution, and $\boldsymbol{\mathcal{E}}_x \neq \boldsymbol{\mathcal{E}}_y$.

The $\{\boldsymbol{X}', \boldsymbol{Y}'\}$ dataset allowed not only to test ALS in an ill-conditioned setting, where the majority of the features are comprised of noise, but also to check whether the weight vectors computed by SCCA contain the "true" features, i.e. features in

$\{\boldsymbol{X}, \boldsymbol{Y}\}$.

All the tests were performed using the multiple hold-out framework described in Chapter 6. Since the aim of the section is to assess the convergence properties of the proposed ALS approach, all the experiments were restricted to the first weight vector pair.

A small change was introduced to the framework. On the original publication [Monteiro et al., 2016], the metric used throughout the framework was the absolute correlation (Equation 3.4.1), this was done to be coherent with previous publications, such as the paper by Parkhomenko et al. [2009]. However, it may actually be unnecessary, due to the fact that SCCA tries to maximise the correlation between the projections, therefore, the train correlation will always be positive. This means that if the test correlation is negative, the method is not generalising well, which usually happens during permutations. Thus, by using the absolute correlation as a metric, one is in fact overestimating the amount of times the method using permuted data outperforms the original correlation, which may lead to a decrease in statistical power. Therefore, all the tests performed in this chapter used the correlation between the projections as the performance metric (instead of the absolute correlation).

The hyper-parameter range explored for the optimisation was determined based on the number of features in each view. More specifically, if the number of features in $\boldsymbol{X}$ is given by p, then the hyper-parameter range was set to: $p_u \in \{0.1p, 0.2p, \ldots, p\}$. An analogous range was applied for $\boldsymbol{Y}$. However, for the $\{\boldsymbol{X}, \boldsymbol{Y}\}$ dataset, $\boldsymbol{Y}$ only has 7 features, thus, in this case the hyper-parameter range was set to: $q_v \in \{1, 2, \ldots, 7\}$.

In Chapter 6, the data were mean-centered and normalised as a pre-processing step, due to the fact that mean-centering and normalising such large data matrices for each data split during validation would increase the computational time. Moreover, our initial investigations revealed that this did not have a noticeable impact on the hyper-parameter optimisation. Since high dimensionality is no longer an issue, in this chapter and in Chapter 8, the data were mean-centered and normalised based on the train data.

7.1.2 Results and Discussion

The results will be presented in three sections. The first (Section 7.1.2.1) will show the performance obtained with the proposed ALS method for both datasets ($\{\boldsymbol{X}, \boldsymbol{Y}\}$ and $\{\boldsymbol{X}', \boldsymbol{Y}'\}$), i.e. how consistent were the correlations obtained during the hyper-parameter selection step across the 10 different hold-out splits. The second section (Section 7.1.2.2) will present the weight vector pairs obtained, and how these compare with the results obtained in Chapter 6. Finally, the last section (Section 7.1.2.3) will compare the performance of the proposed ALS formulation, with two simpler ALS formulations, both in terms of the obtained test correlations, and how often the algorithms converged.

7.1.2.1 Test correlations

SCCA using ALS (Algorithm 7.3) was able to find statistically significant effects ($p < 0.005$ for all the hold-out datasets), for both $\{\boldsymbol{X}, \boldsymbol{Y}\}$ and $\{\boldsymbol{X}', \boldsymbol{Y}'\}$. The correlations on the hold-out datasets, with the corresponding p-values, are presented in Appendix C.1.1.

Figure 7.1 shows the correlation values on the test datasets (used for hyper-parameter selection) and on the hold-out datasets. As one can see, the introduction of noise decreased the average correlations, however, these were still quite high for both cases.

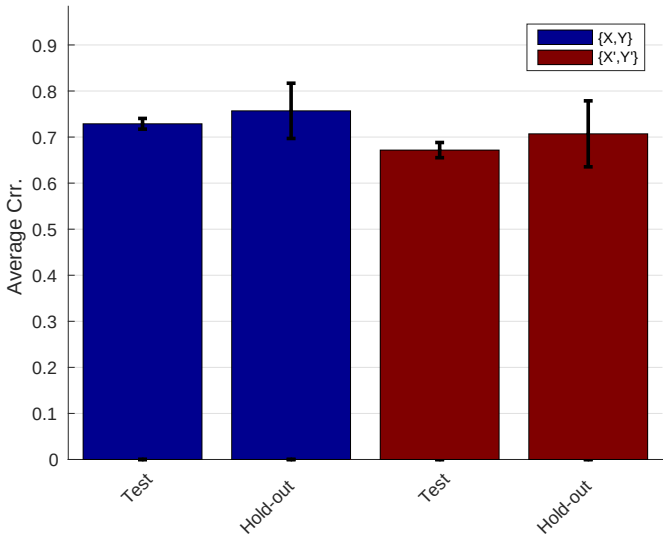


Figure 7.1: Average correlations for both datasets.

Figure 7.2 shows the average test correlations obtained during hyper-parameter

selection, for all hyper-parameter combinations across the 10 hold-out splits. As one can see, the hyper-parameter estimation was fairly consistent across the 10 splits, which indicates that SCCA was robust to perturbations in the data. The optimal hyper-parameters selected also indicate that all the clinical/demographic features ($\boldsymbol{Y}$) should be included in the model, while only approximately half the ROI features ($\boldsymbol{X}$) should be included.

In the case of the ill-conditioned dataset $\{\boldsymbol{X}', \boldsymbol{Y}'\}$, the optimal hyper-parameter combination was the one corresponding to the sparsest solution (Figure 7.3). This result was expected, since most of the features in the data matrices were comprised of noise. Note that one might obtain an even better solution for $p_u < 0.1p$ and $q_v < 0.1q$, but these were the smallest hyper-parameters tested.

The plots in Figure 7.3 show that the correlations did not change much when the hyper-parameter is low in one view, and high in the other view, i.e. low p_u with high q_v, and *vice versa*. This is due to the fact that the solutions obtained with these hyper-parameter combinations sometimes contain far less features than the ones defined by the constraints $\{p_u, q_v\}$ (Equation 7.2). When the LASSO problem is solved using the `glmnet`, the package sometimes outputs a solution which is sparser than the specified upper bound on the number of features (p_u or q_v). In other words, even with a generous upper bound (p_u or q_v), `glmnet` removed some of the noisy features. This resulted in a difference between the number of features included by the weight vectors $\{\boldsymbol{u}, \boldsymbol{v}\}$ and the model constraints $\{p_u, q_v\}$. For more information regarding this difference, please refer to the results presented in Appendix C.1.2.

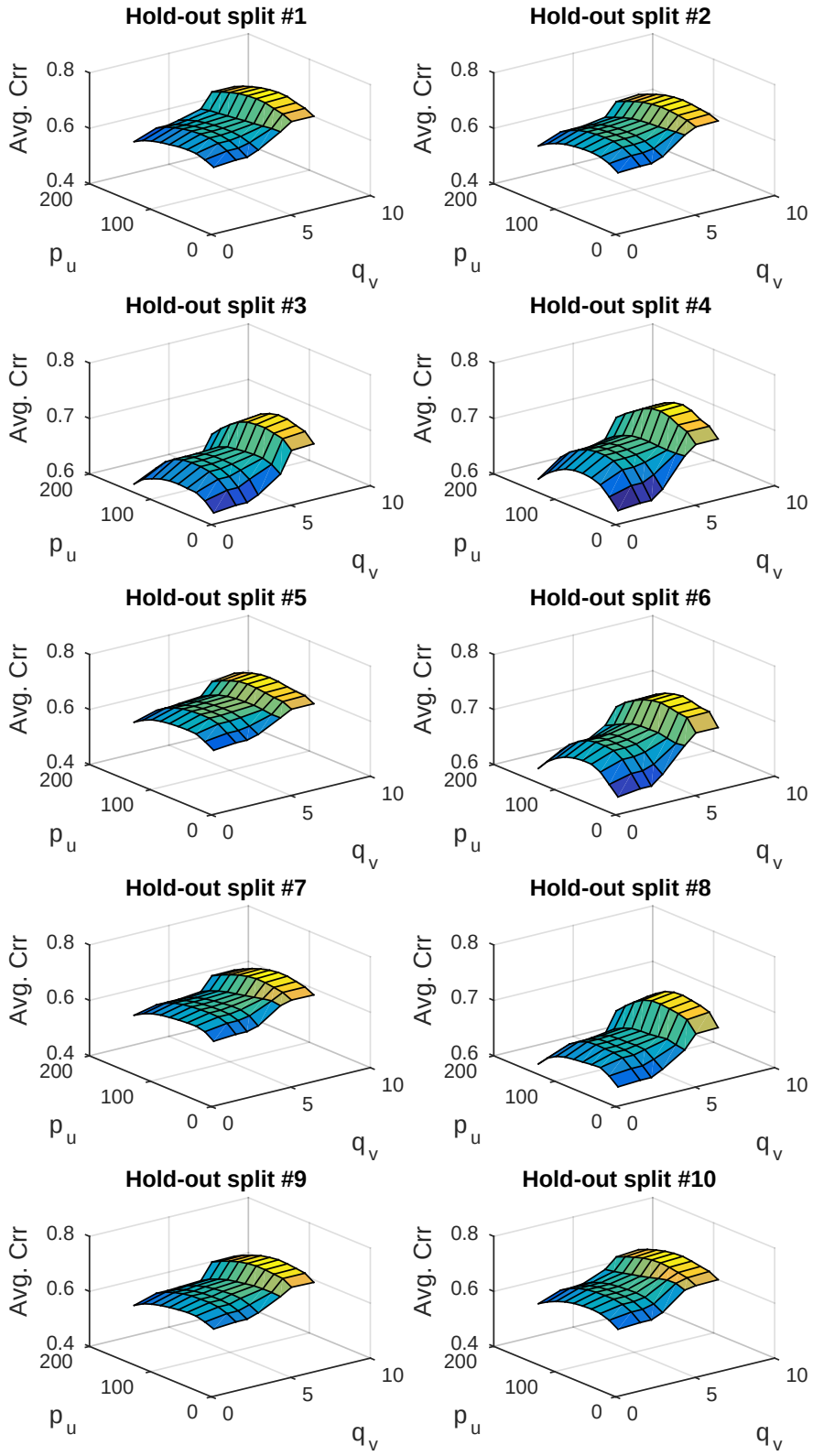


Figure 7.2: Average test correlations for each hyper-parameter optimisation step. Original dataset ($\{\boldsymbol{X}, \boldsymbol{Y}\}$).

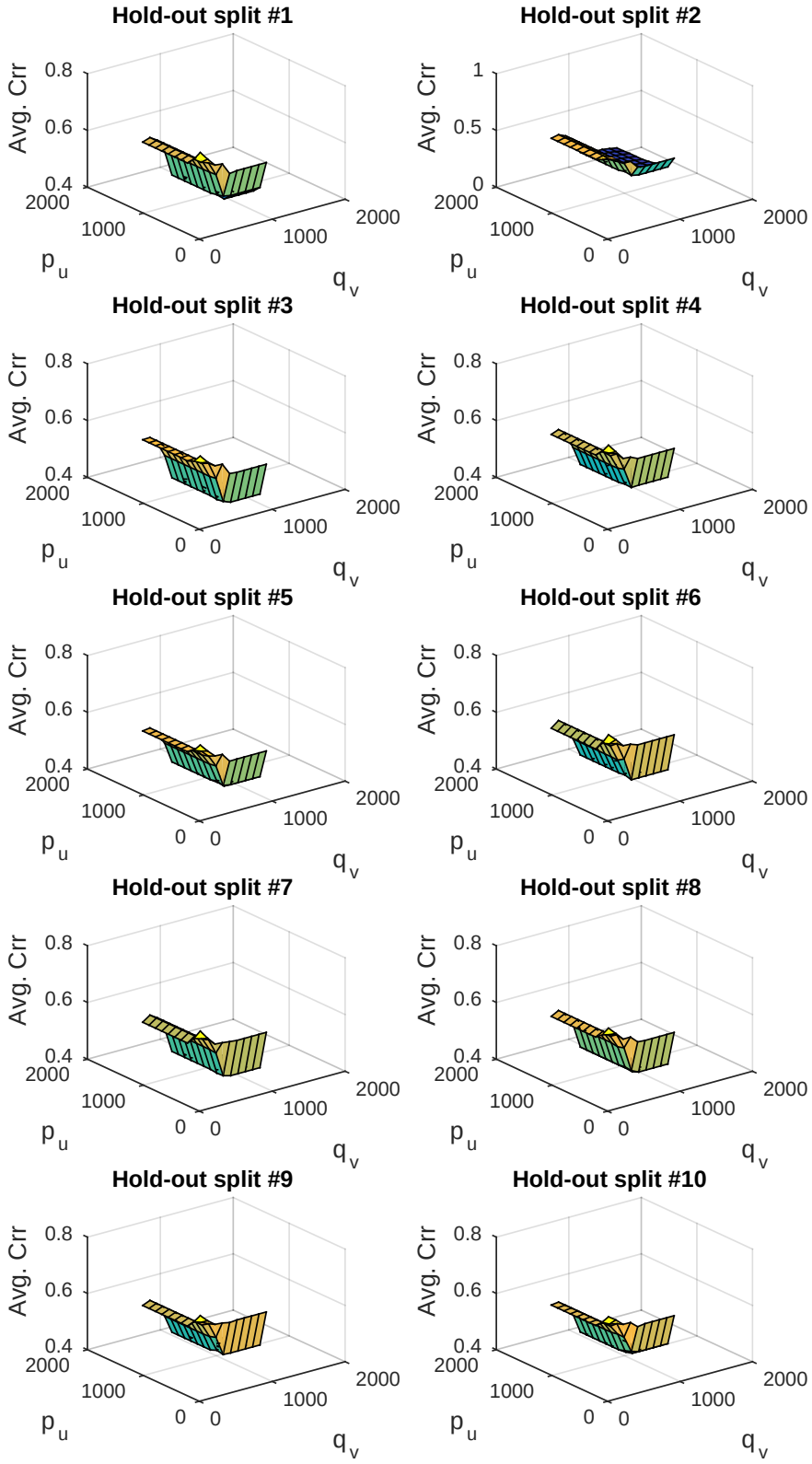
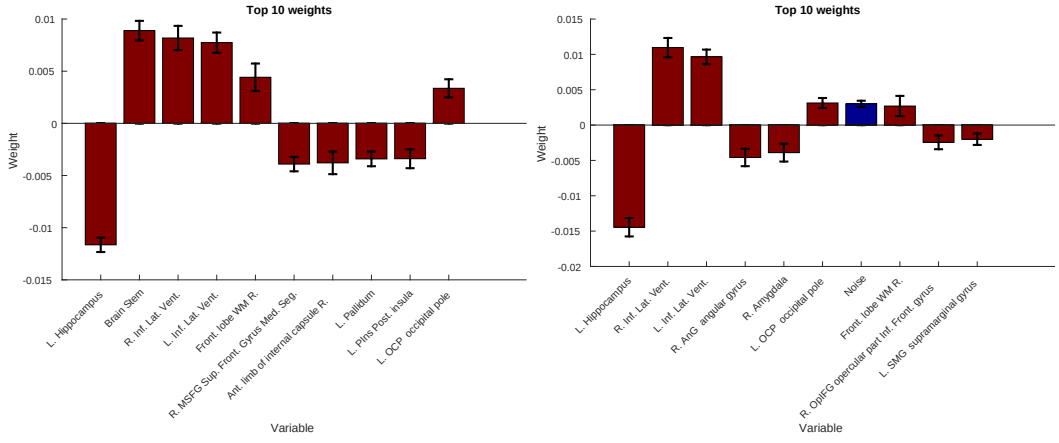


Figure 7.3: Average test correlations for each hyper-parameter optimisation step. Noisy dataset ($\{\boldsymbol{X}', \boldsymbol{Y}'\}$).

7.1.2.2 Weight vectors

Figure 7.4 shows the average of the 10 ROI weight vectors for both datasets used. As one can see, the weight with the largest magnitude corresponded to the hippocampus, which is consistent with previous findings in the AD literature [Jack et al., 2000] and with the results presented in Chapter 6. However, it is interesting to note that only the left hippocampus was selected with such a large weight, the right hippocampus had a much smaller weight, which was not observed in Chapter 6, where both hippocampi had similar weights. This may be due to the fact that the ALS method uses LASSO regression steps, which are known to down-weight/exclude features which correlate with other features already included in the model [Zou and Hastie, 2005].

The addition of noise to the dataset ($\{\boldsymbol{X}', \boldsymbol{Y}'\}$) caused the weights to change (Figure 7.4(b)). However, one can see that the top weight was still the same as in the original dataset (left hippocampus). Other top weights are also reasonable from a biological point of view, e.g. the hippocampus and the amygdala have weights with opposite signs to the ones in the ventricles, which is consistent with how their volumes tend to change with dementia: hippocampus and amygdala decrease in volume, while ventricles increase.

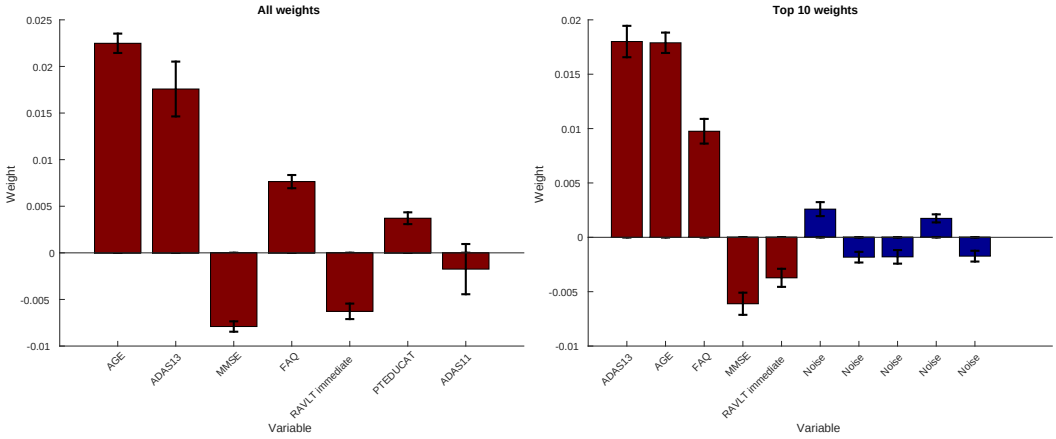


(a) $\{\boldsymbol{X}, \boldsymbol{Y}\}$ – Mean $\boldsymbol{u}$. Top 10 weights. **(b)** $\{\boldsymbol{X}', \boldsymbol{Y}'\}$ – Mean $\boldsymbol{u}$. Top 10 weights.

Figure 7.4: Mean ROI weight vectors across the 10 hold-out splits. Red bars correspond to features coming from the original dataset $\boldsymbol{X}$ and blue bars correspond to noisy features from $\boldsymbol{\mathcal{E}}_x$.

The average clinical weights are presented in Figure 7.5. As one can see, the top weights on the noisy dataset were all associated with variables in the original dataset

$\{\boldsymbol{X}, \boldsymbol{Y}\}$ (Figure 7.5(b)). However, two of the clinical features were not included in the top 10: ADAS11 and Education. The former was not selected in any of the 10 data splits, which may be due to the fact that it is very correlated with ADAS13, which was included in the model. All the "true" features presented in Figure 7.5(b) had the same sign as the ones in Figure 7.5(a), i.e. if a feature had a positive weight when using $\boldsymbol{Y}$, it also had a positive weight when using $\boldsymbol{Y}'$, and *vice versa*.



(a) $\{\boldsymbol{X}, \boldsymbol{Y}\}$ – Mean $\boldsymbol{v}$ **(b)** $\{\boldsymbol{X}', \boldsymbol{Y}'\}$ – Mean $\boldsymbol{v}$. Top 10 weights.

Figure 7.5: Mean clinical weight vectors across the 10 hold-out splits. Red bars correspond to features coming from the original dataset $\boldsymbol{Y}$ and blue bars correspond to noisy features from $\boldsymbol{\mathcal{E}}_y$.

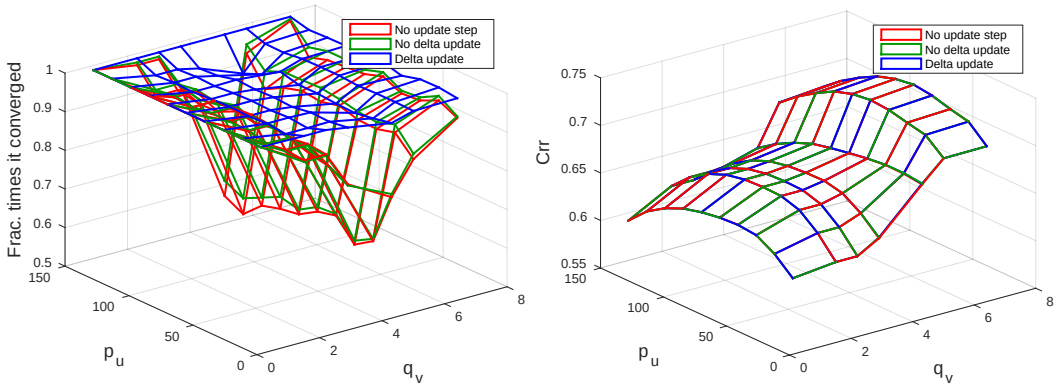
7.1.2.3 Comparison of ALS algorithms

Three algorithms are compared in this section: ALS without the `update` step, i.e. Algorithm 7.3 where the `update` steps are substituted by vector rescaling steps ($\boldsymbol{u} \leftarrow \boldsymbol{u}/\|\boldsymbol{X}\boldsymbol{u}\|_2$ and $\boldsymbol{v} \leftarrow \boldsymbol{v}/\|\boldsymbol{Y}\boldsymbol{v}\|_2$); ALS with the `update` step, but with a fixed value of δ; and the final proposed algorithm with the `update` step and with the update on the value of δ, as described in Algorithm 7.3.

The results presented below summarise the performance during the hyper-parameter selection steps for all the 10 hold-out data splits, i.e. each hyper-parameter combination was evaluated using 1000 data splits.

Figure 7.6 shows the results obtained when using the original dataset without the addition of noisy features ($\{\boldsymbol{X}, \boldsymbol{Y}\}$). By comparing how ALS performed before (red) and after (green) the addition of the `update` step, one can see that there was a slight improvement in the fraction of times it converged (Figure 7.6(a)). The update of the value of δ provided even better results (blue). Despite this improvement, one

can see that the impact on the test correlations was negligible (Figure 7.6(b)), which seems to support our initial hypothesis that, by not forcing the algorithm to converge to a result using the proposed ALS method, there is the risk that it will oscillate between a set of very similar solutions.



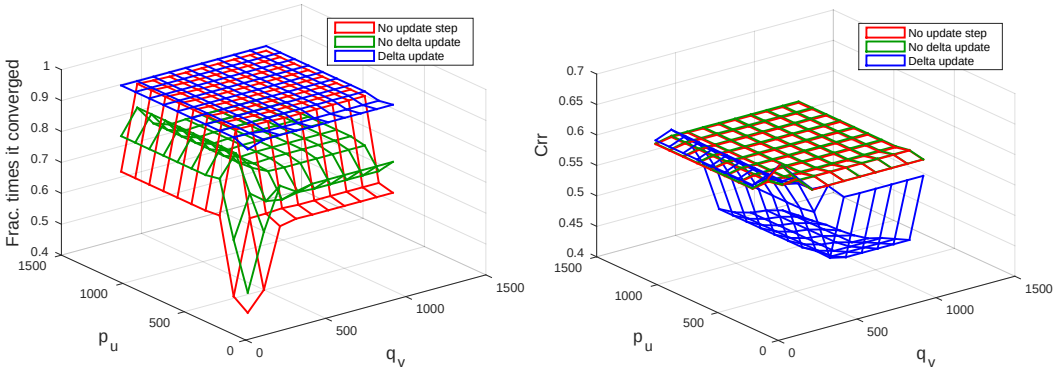
(a) Fraction of times algorithm converged **(b)** Average test correlation

Figure 7.6: Comparison of ALS algorithms using original dataset $\{\boldsymbol{X}, \boldsymbol{Y}\}$. Both plots show results for hyper-parameter selection across the 10 hold-out splits. **Red** – ALS without the `update` step; **Green** – ALS with the `update` step, but with a fixed δ; **Blue** – proposed algorithm, i.e. ALS with `update` step and update on the value of δ.

Figure 7.7 shows the same results as Figure 7.6, but using the dataset with added noisy features ($\{\boldsymbol{X}', \boldsymbol{Y}'\}$). In this case, one can see that the algorithms behave quite differently. Figure 7.7(a) shows that for the optimal hyper-parameter combination, the addition of the `update` step (green) improved the fraction of times the algorithm converged from 0.4530 to 0.5180, which was further improved by the update on the value of δ (blue) from 0.5180 to 0.9820. However, this was not the case for most hyper-parameter combinations, where the original ALS formulation (red) converged as often as the proposed formulation (blue).

Figure 7.7(b) shows that, unlike the results with the original dataset (Figure 7.6(b)), the correlation obtained on the test sets actually changed with the ALS algorithm. For most of the hyper-parameter combinations, the proposed method actually performed worse (blue) than the original ALS (red), or the ALS with the `update` step but without the update on the value of δ (green). This result may be due an implementation issue, where the proposed ALS formulation assumes that the algorithm is oscillating too soon, and prematurely decreases the value of δ, forcing ALS to converge to a sub-optimal solution.

Despite these results, the difference between the correlation obtained by the three different ALS algorithms for the optimum hyper-parameter combination was negligible: there was a small increase from 0.6714 (red) to 0.6717 (blue). Therefore, the proposed ALS algorithm will still choose the same hyper-parameter combination as the other two ALS algorithms. This result leads us to believe that using the proposed ALS algorithm is the best option, as it will provide comparable solutions, while converging more often. This last property is especially attractive, since it will decrease the computational time, by preventing the algorithm from spending time oscillating between very similar solutions.



(a) Fraction of times algorithm converged **(b)** Average test correlation

Figure 7.7: Comparison of ALS algorithms using dataset with added noise $\{\boldsymbol{X}', \boldsymbol{Y}'\}$. Both plots show results for hyper-parameter selection across the 10 hold-out splits. **Red** – ALS without the `update` step; **Green** – ALS with the `update` step, but with a fixed δ; **Blue** – proposed algorithm, i.e. ALS with `update` step and update on the value of δ.

7.1.3 Conclusion

This section has served mainly as an introduction to the remaining work presented in this thesis (Section 7.2 and Chapter 8). However, it also proposes a different formulation of the ALS algorithm [Golub and Zha, 1992]. This differs from earlier ALS formulations [Lykou and Whittaker, 2010, Chi et al., 2013, Wilms and Croux, 2015, Polajnar, 2015] by introducing an extra update step on the weight vectors, in order to prevent the algorithm for oscillating between similar solutions.

The algorithm was tested on a dementia dataset containing ROI information and clinical/demographic information ($\{\boldsymbol{X}, \boldsymbol{Y}\}$), and on a higher dimensional dataset where $p, q > n$ and most features were comprised of noise ($\{\boldsymbol{X}', \boldsymbol{Y}'\}$). This was done to assess whether the ALS algorithm could be used for SCCA when the problem was

ill-conditioned. The results have shown that the ALS was indeed able to provide statistically significant results, which were robust to perturbations in the data.

There were still a few cases in which ALS did not converge, due to the fact that the constraints could not be obeyed. This was especially common in cases where the constraints on the number of features were very strict, i.e. only one or two features were allowed. In some of these cases, the `glmnet` package was not able to provide a solution which obeyed the constraints. Nevertheless, SCCA was able to compute weight vector pairs which were able to generalise well for test data.

One of the possible limitations of the ALS is the fact that its computational time did not scale well with the number of features. This scalability will be dependent on the algorithm used to solve each LASSO regression step. As mentioned in Section 7.1.1.1, `glmnet` was chosen due to the fact that it was the best performing algorithm in early preliminary experiments. However, the computational time of ALS can be further reduced if the LASSO regression steps are solved using a faster algorithm.

Despite the encouraging results, there are still questions that should be investigated, more specifically: how do these results compare with SPLS; is it possible to use other types of penalties besides the LASSO; and is it possible to solve SPLS using ALS. These shall be addressed in Section 7.2.

7.2 SCCA vs. SPLS

7.2.1 Introduction

Section 7.1 discussed some of the properties of the ALS approach when used with a LASSO penalty to solve a SCCA problem. However, no comparisons have been made between this method, and other potential candidates, such as: SCCA using an elastic net penalty, SPLS, CCA, and PLS.

In this section, a comparison between these different approaches is made. However, a direct comparison between SCCA in its current ALS formulation and SPLS is not completely fair, as the latter is solved using a different algorithm (the power method). This section tries to address this issue by solving a SPLS problem using ALS.

As mentioned in the introduction of this chapter, one of the advantages of using an ALS framework is that the SCCA penalties can be easily changed, provided that there

are functions available to solve the corresponding regression steps. By substituting the LASSO regression step by an elastic net regression step, the properties of SCCA should become more similar with the properties of an elastic net. Thus, by changing these regression steps, the following methods were compared using the ALS approach: SCCA using LASSO (SCCA-ALS-L1), SPLS using LASSO (SPLS-ALS-L1), SCCA using elastic net (SCCA-ALS-EN), SPLS using elastic net (SPLS-ALS-EN), and non-penalised CCA. These were also compared with two PLS approaches solved using the power method: PLS (Algorithm 3.2) and SPLS (Algorithm 3.4).

Note that the use of ALS to solve SPLS has been previously proposed in the literature [Chi et al., 2013]. However, as far as we are aware, no comparisons have been made between the performance of SPLS using ALS, and SPLS using the power method. By making these comparisons, this section aims not only to determine which method provides the best test correlations, but also how consistent are the methods across the different hold-out splits, which features are selected by the different methods, and how different are the projections of the data onto the computed weight vector pairs.

7.2.2 Materials and Methods

7.2.2.1 SPLS using ALS

In order to make the comparisons between SCCA and SPLS consistent, an adaptation of the ALS algorithm presented in Section 7.1 is proposed to solve SPLS.

By looking at the ALS algorithm for SCCA (Algorithm 7.3), one can see that the essential steps are the regression operations performed to estimate $\boldsymbol{u}$ (Step 5) and $\boldsymbol{v}$ (Step 10). These steps can be re-written to solve a SPLS problem.

Consider the update steps for each weight vector in the PLS-SVD algorithm (Algorithm 3.2), i.e. the power method applied to the covariance matrix $\boldsymbol{X}^\intercal\boldsymbol{Y}$:

$$\boldsymbol{u} \leftarrow \boldsymbol{X}^\intercal\boldsymbol{Y}\boldsymbol{v} \quad \text{and} \quad \boldsymbol{v} \leftarrow (\boldsymbol{X}^\intercal\boldsymbol{Y})^\intercal\boldsymbol{u} \tag{7.3}$$

The steps performed by the power method (Equation 7.3), can be re-written as regression steps:

$$\boldsymbol{u} \leftarrow \arg\min_{\boldsymbol{u}} \|\boldsymbol{X}^\intercal\boldsymbol{Y}\boldsymbol{v} - \boldsymbol{u}\|_2^2 \quad \text{and} \quad \boldsymbol{v} \leftarrow \arg\min_{\boldsymbol{v}} \|(\boldsymbol{X}^\intercal\boldsymbol{Y})^\intercal\boldsymbol{u} - \boldsymbol{v}\|_2^2 \tag{7.4}$$

If the regression steps described in Equation 7.4 are performed in an ALS framework, and the weight vectors are re-scaled by their l_2-norms (instead of the norms of their projections), then one obtains an algorithm to solve PLS using ALS.

In order to use ALS to solve a SPLS problem, one has to add sparse penalies. The regression steps described in Equation 7.4 are then re-written as LASSO regression steps:

$$\begin{gathered} \boldsymbol{u} \leftarrow \arg\min_{\boldsymbol{u}} \|\boldsymbol{X}^\intercal \boldsymbol{Y}\boldsymbol{v} - \boldsymbol{u}\|_2^2 + \gamma\|\boldsymbol{u}\|_1 \\ \text{and} \\ \boldsymbol{v} \leftarrow \arg\min_{\boldsymbol{v}} \|(\boldsymbol{X}^\intercal \boldsymbol{Y})^\intercal \boldsymbol{u} - \boldsymbol{v}\|_2^2 + \gamma\|\boldsymbol{v}\|_1 \end{gathered} \tag{7.5}$$

In the papers by Chi et al. [2013] and Grellmann et al. [2015], the authors describe a SCCA algorithm whose steps look very similar to the ones expressed in Equation 7.5. Where each weight is fixed alternately and a regression problem is solved. In other words, for a fixed $\boldsymbol{v}$, $\boldsymbol{u}$ is computed using Algorithm 7.4. Which is an indication that the problem the authors solved was a SPLS problem, and not a SCCA problem.

Algorithm 7.4 Regression step described by Chi et al. [2013] and Grellmann et al. [2015].

1: $\boldsymbol{u} \leftarrow \arg\min_{\|\boldsymbol{u}\|_2=1} \frac{1}{2}\|\boldsymbol{X}^\intercal \boldsymbol{Y}\boldsymbol{v} - \boldsymbol{u}\|_2^2 + \gamma\|\boldsymbol{u}\|_1$
2: $\boldsymbol{u} \leftarrow \boldsymbol{u}/\|\boldsymbol{u}\|_2$

As previously mentioned in Section 7.1.1, Chi et al. [2013] proposed an extension of this formulation which may be closer to SCCA, however, the connection between SCCA and SPLS is not explicitly made.

Note that the expressions in Equation 7.5 are equivalent to:

$$\boldsymbol{u} \leftarrow \arg\min_{\boldsymbol{u}} \|\boldsymbol{b}_v - \boldsymbol{I}_u\boldsymbol{u}\|_2^2 + \gamma\|\boldsymbol{u}\|_1 \quad \text{and} \quad \boldsymbol{v} \leftarrow \arg\min_{\boldsymbol{v}} \|\boldsymbol{b}_u - \boldsymbol{I}_v\boldsymbol{v}\|_2^2 + \gamma\|\boldsymbol{v}\|_1$$

where $\boldsymbol{b}_v = \boldsymbol{X}^\intercal \boldsymbol{Y}\boldsymbol{v}$, $\boldsymbol{b}_u = (\boldsymbol{X}^\intercal \boldsymbol{Y})^\intercal \boldsymbol{u}$, and $\boldsymbol{I}_u \in \mathbb{R}^{p\times p}$ and $\boldsymbol{I}_v \in \mathbb{R}^{q\times q}$ are identity matrices. The use of these identity matrices allows one to use several popular regression packages (e.g. `glmnet`), whose functions typically require a data matrix as an input argument.

Based on the ALS algorithm for SCCA (Algorithm 7.3), a method is proposed to solve a SPLS problem using ALS (Algorithm 7.5).

Algorithm 7.5 SPLS using ALS. The `update` function is described in Algorithm 7.6.

Set $\boldsymbol{C} = \boldsymbol{X}^\intercal \boldsymbol{Y}$
Set $\boldsymbol{u}^{(0)}$ and $\boldsymbol{v}^{(0)}$ equal to the first singular vector pair of $\boldsymbol{C}$.
Set $\delta = 0.5$
Define the identity matrices $\boldsymbol{I}_u \in \mathbb{R}^{p \times p}$ and $\boldsymbol{I}_v \in \mathbb{R}^{q \times q}$
repeat

  $\boldsymbol{b}_v = \boldsymbol{C}\boldsymbol{v}^{(i)}$
  Solve $\boldsymbol{u}^{(i+1)} \leftarrow \arg\min_{\boldsymbol{u}} \|\boldsymbol{b}_v - \boldsymbol{I}_u \boldsymbol{u}\|_2^2 + \gamma \|\boldsymbol{u}\|_1$, with γ such that $\|\boldsymbol{u}\|_1 \leqslant c_u$
  $\boldsymbol{u}^{(i+1)} \leftarrow \text{UPDATE}(\boldsymbol{u}^{(i+1)}, \boldsymbol{u}^{(i)}, \delta)$

  $\boldsymbol{b}_u \leftarrow \boldsymbol{C}^\intercal \boldsymbol{u}^{(i+1)}$
  Solve $\boldsymbol{v}^{(i+1)} \leftarrow \arg\min_{\boldsymbol{v}} \|\boldsymbol{b}_u - \boldsymbol{I}_v \boldsymbol{v}\|_2^2 + \gamma \|\boldsymbol{v}\|_1$, with γ such that $\|\boldsymbol{v}\|_1 \leqslant c_v$
  $\boldsymbol{v}^{(i+1)} \leftarrow \text{UPDATE}(\boldsymbol{v}^{(i+1)}, \boldsymbol{v}^{(i)}, \delta)$

  if Algorithm is oscillating **then**
    $\delta \leftarrow \delta / 2$
  end if

  $i \leftarrow i + 1$
until convergence
return $\boldsymbol{u}$, $\boldsymbol{v}$

Algorithm 7.6 Update weight function.

function UPDATE($\boldsymbol{w}^{(i+1)}, \boldsymbol{w}^{(i)}, \delta$)
  $\boldsymbol{w}^{(i+1)} = \boldsymbol{w}^{(i+1)} / \|\boldsymbol{w}^{(i+1)}\|_2$
  $I_w := \left\{ j \mid w_j^{(i+1)} = 0 \bigwedge |w_j^{(i)}| < \epsilon \right\}$
  $\Delta \boldsymbol{w} = \delta \left(\boldsymbol{w}^{(i+1)} - \boldsymbol{w}^{(i)} \right)$
  $\boldsymbol{w}^{(i+1)} \leftarrow \boldsymbol{w}^{(i)} + \Delta \boldsymbol{w}$
  $w_j^{(i+1)} = 0$, $j \in I_w$
  $\boldsymbol{w}^{(i+1)} = \boldsymbol{w}^{(i+1)} / \|\boldsymbol{w}^{(i+1)}\|_2$
  return $\boldsymbol{w}^{(i+1)}$
end function

The differences between Algorithm 7.3 and Algorithm 7.5 are: the projection steps, the weight vector re-scaling in Algorithm 7.6 (so that the SPLS l_2-norm constraints are obeyed), and the setting of the γ penalty terms in the LASSO regression steps (so that the SPLS l_1-norm constraints $\{c_u, c_v\}$ are obeyed).

As mentioned in Section 2.1.1, one of the disadvantages of the LASSO [Tibshirani, 1996] is that it tends to exclude variables that are correlated with other variables already included in the model. Zou and Hastie [2005] proposed an approach known as the elastic net, which consists in adding a l_2-norm penalty to the LASSO optimisation

problem, which results in a method that provides sparse solutions while maintain a grouping effect:

$$\underset{\boldsymbol{w}}{\text{minimise}} \ \|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{w}\|_2^2+\gamma_2\|\boldsymbol{w}\|_2^2+\gamma_1\|\boldsymbol{w}\|_1 \tag{7.6}$$

The problem expressed in Equation 7.6 can be re-written as:

$$\underset{\boldsymbol{w}}{\text{minimise}} \ \frac{1}{2n}\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{w}\|_2^2+\gamma\left((1-\alpha)\frac{1}{2}\|\boldsymbol{w}\|_2^2+\alpha\|\boldsymbol{w}\|_1\right) \tag{7.7}$$

which is the formulation that is used in the `glmnet` package [Friedman et al., 2010]. Note that $1/(2n)$ and $1/2$ are scaling terms. These are included to simplify the derivatives of the squared norms.

The α term in Equation 7.7 (not to be confused with the $\boldsymbol{\alpha}$ commonly used to refer to dual weight vectors) is a hyper-parameter between 0 and 1, which controls the trade-off between the l_1-norm penalty and the l_2-norm penalty. Thus, instead of directly controlling the penalty hyper-parameters $\{\gamma_1,\gamma_2\}$ (Equation 7.6), this alternative formulation controls how much the penalty influences the result (γ) and how much that penalty is influenced by the l_1-norm or the l_2-norm of $\boldsymbol{w}$ (which is controlled by α). This means that if $\alpha=1$ the optimisation problem is equivalent to the LASSO (Equation 2.9), and if $\alpha=0$, the problem becomes a ridge regression (Equation 2.8). Values between 0 and 1 provide an interpolation between the two optimisation problems, i.e. the elastic net optimisation problem.

Note that, if the sparsity penalties are not included, PLS solved using the classical ALS approach and PLS solved using the power method are mathematically equivalent (proof in Appendix A.2). However, this is not the case for SPLS. Therefore, only the results for PLS solved using the power method will be presented in this section, whereas the SPLS results will be presented using both the ALS and the power method.

7.2.2.2 Experiments

All the experiments were performed using the $\{\boldsymbol{X},\boldsymbol{Y}\}$ dataset from Section 7.1 with the multiple hold-out framework. The following seven methods were tested:

- Non-sparse methods:
  - **CCA**: non-penalised CCA solved using ALS;

  - **PLS**: non-penalised PLS solved using the power method;

- SCCA methods:

  - **SCCA-ALS-L1**: SCCA solved using ALS with the LASSO;
  - **SCCA-ALS-EN**: SCCA solved using ALS with the elastic net;

- SPLS methods:

  - **SPLS-PM**: SPLS solved using the power method (Algorithm 3.4);
  - **SPLS-ALS-L1**: SPLS solved using ALS with the LASSO;
  - **SPLS-ALS-EN**: SPLS solved using ALS with the elastic net;

As previously mentioned, the elastic net (Equation 7.7) has two hyper-parameters to tune $\{\gamma, \alpha\}$. However, the value of α was fixed *a priori* to 0.5, otherwise, both SCCA-ALS-EN and SPLS-ALS-EN would have four hyper-parameters to tune, which would drastically increase the computational time of the hyper-parameter selection steps.

The hyper-parameter values used for hyper-parameter selection were comprised of 40 equidistant points for $p_u \in [0.01p, p]$ and $q_v \in \{1, \ldots, 7\}$ for both SCCA-ALS-L1 and SCCA-ALS-EN. For all the SPLS methods, 40 equidistant points were used between $c_u \in [1, \sqrt{p}]$ and $c_v \in [1, \sqrt{q}]$.

7.2.3 Results and Discussion

7.2.3.1 Test correlations

All the results were statistically significant, for a complete list of the hold-out correlations and corresponding p-values, please refer to Appendix C.2.1.

Figure 7.8 shows the averages of the best correlations obtained during the hyper-parameter selection step. These are plotted instead of the correlations obtained in the hold-out datasets (Appendix C.2.1) due to the fact that each one is based on the average of 100 fits, which should make them more robust estimates for model comparison.

The results show that SCCA-ALS-L1 performed the best, however, the correlations were very similar to the ones obtained with SCCA-ALS-EN. The SPLS methods performed worse than the SCCA methods, with a difference in the average

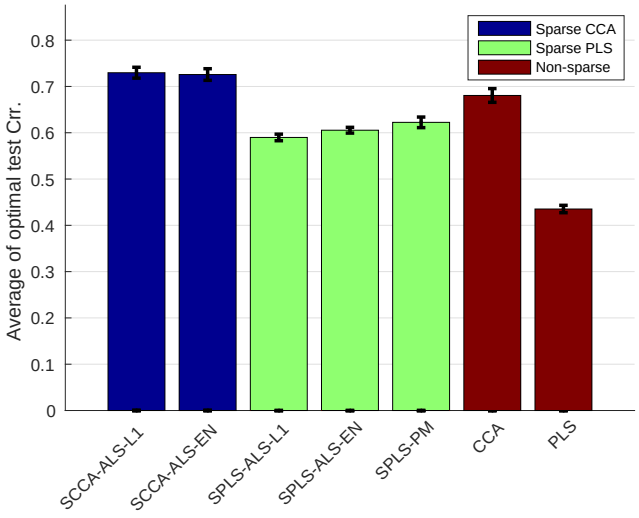


Figure 7.8: Average of the optimal test correlations for all the seven methods tested.

test correlation larger than between SCCA-ALS-L1 and SCCA-ALS-EN. This was expected, since SCCA optimises the performance metric, i.e. correlation between the projections, whereas SPLS optimises the covariance between the projections.

Among the non-sparse methods, CCA performed better than PLS, in fact, CCA performed even better than all the SPLS methods. This was expected, since CCA is maximising correlation instead of covariance, and the dataset is fairly low dimensional, which means that sparsity is less likely to provide a relatively large improvement.

Figure 7.9 shows the average test correlation obtained during the hyper-parameter optimisation step for all 10 hold-out splits using SPLS-PM. As one can see, the hyper-parameter selection was quite consistent across the 10 splits. Note that the x and y axes are different than the ones from previous figures in this chapter (Figures 7.2 and 7.3). The axes in Figures 7.2 and 7.3 represent the upper bound on the number of features (p_u and q_v), whereas the axes in Figure 7.9 represent the l_1-norm constraint on the SPLS weight vectors (just as in Chapter 6).

Figure 7.10 shows the same information as Figure 7.9, but for SPLS-ALS-EN. As one can see, these results are different from the ones obtained with SPLS-PM (Figure 7.9), this suggests that SPLS solved with the power method leads to different results from SPLS solved using ALS. The results for SPLS-ALS-L1 were similar to the ones shown in Figure 7.10, these are presented in Appendix C.2.2.

The SCCA-ALS-L1 and SCCA-ALS-EN provided similar results to the ones shown in the previous section (Figure 7.2), these can be found in Appendix C.2.2.

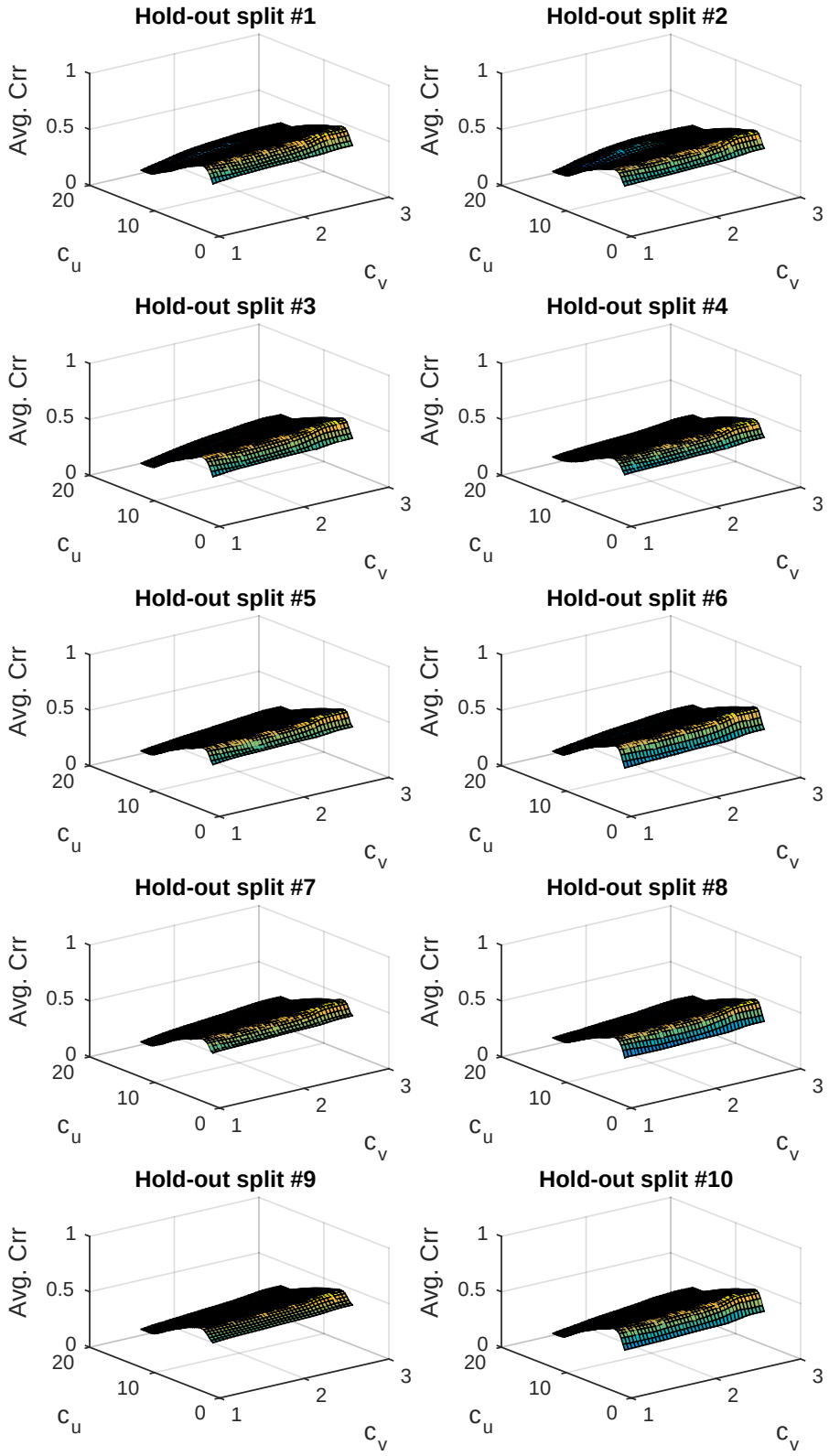


Figure 7.9: Average test correlations for each hyper-parameter optimisation step, using SPLS-PM.

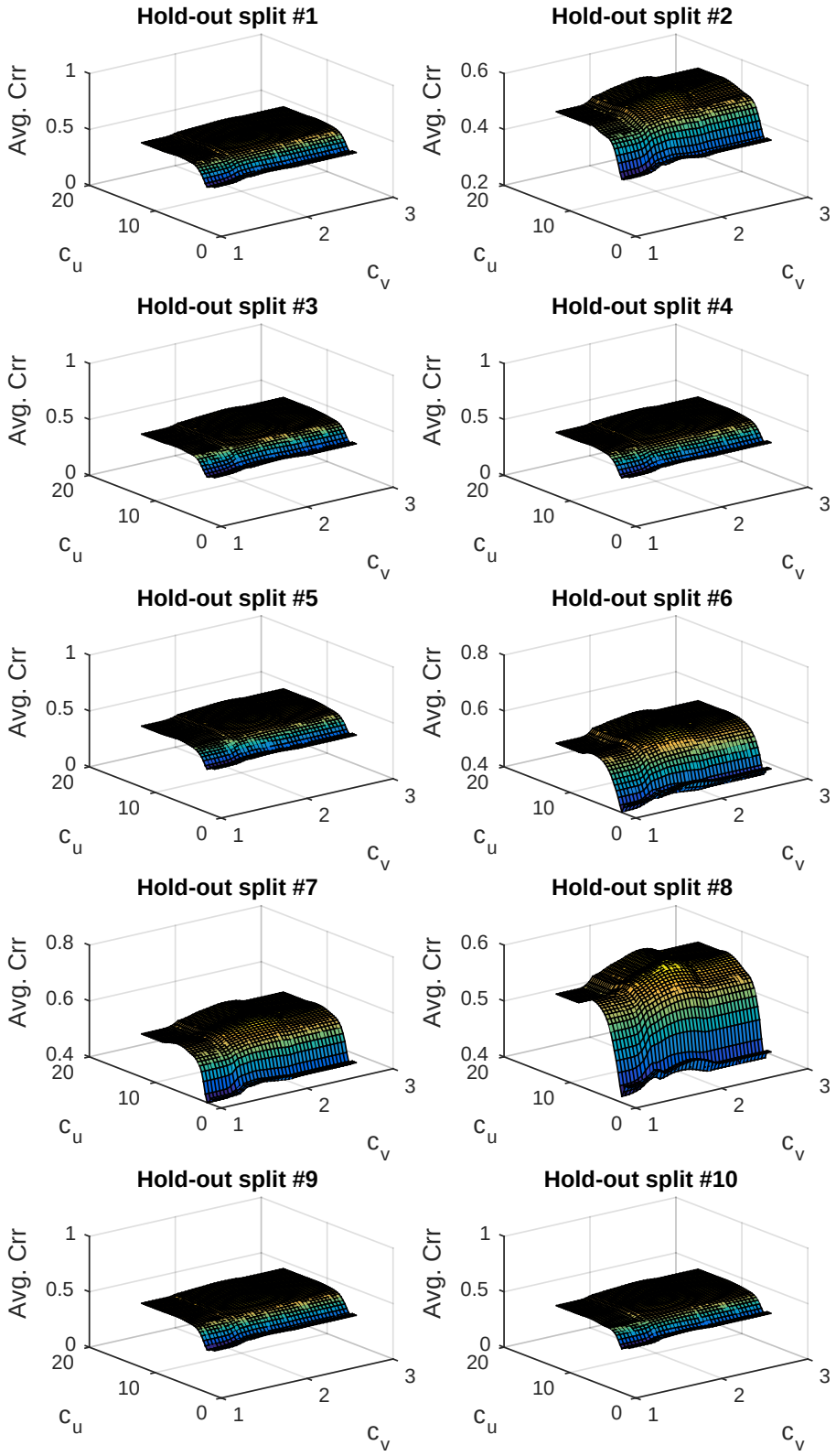


Figure 7.10: Average test correlations for each hyper-parameter optimisation step, using SPLS-ALS-EN.

The similarity between the results acquired by both methods was expected, since they have also shown very similar correlations on the test sets (Figure 7.8).

7.2.3.2 Weight vectors

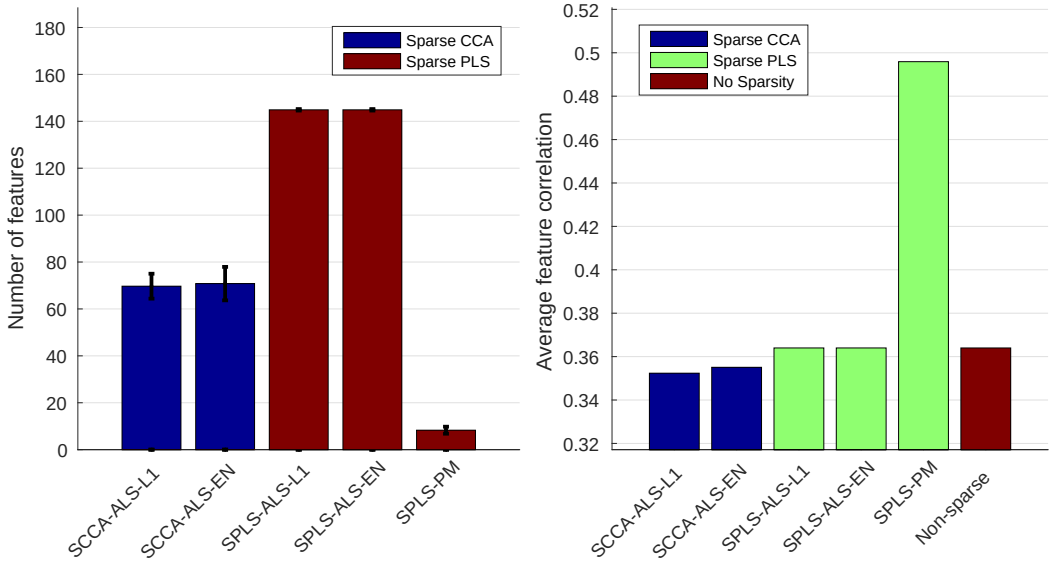
The weights in the clinical/demographic view ($\boldsymbol{Y}$), were not very sparse. In fact, the SCCA methods selected 6 features (ADAS11 was excluded), while the SPLS methods selected all 7 features. Due to the low dimensionality of this view, this section will focus mainly on the results on the view containing the ROI data ($\boldsymbol{X}$).

A summary of the ROI weights ($\boldsymbol{u}$) selected by the models is presented in Figure 7.11. As one can see, the SCCA methods computed solutions with a similar number of features (Figure 7.11(a)). This was not the case for the SPLS methods, where SPLS-ALS-L1 and SPLS-ALS-EN computed solutions selecting all the ROI features, while SPLS-PM computed sparse solutions. This result was expected, since Figures 7.9 and 7.10 show that the optimal hyper-parameter combinations for each SPLS method were very different.

Another interesting aspect to explore is how correlated were the features that were selected by each method. In order to answer this question, the correlation matrix between the variables of $\boldsymbol{X}$ was computed, and the average absolute correlation for the subset of all the features selected by each method was calculated. Figure 7.11(b) shows how correlated the selected features were with each other. The baseline correlation is shown in red, i.e. the average absolute feature correlation obtained using non-sparse methods. As one can see, both SCCA methods selected a less correlated set of features than the baseline provided by non-sparse methods. This was not the case for SPLS-ALS-L1 and SPLS-ALS-EN, which provided the same average absolute feature correlation as the baseline correlation. This result was expected, since all the ROI features were included when using these methods. Interestingly, SPLS-PM provided sparse solutions whose features tended to be more correlated with each other than in all the other methods.

Figure 7.11 provides some information regarding the overall properties of the weight vectors, however, it does not contain any information regarding which weights were attributed to which features. In order to further analyse these weights, the following procedure was performed:

1. Compute the average weight vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ across the 10 hold-out data splits,



(a) Number of features selected by $\boldsymbol{u}$ **(b)** Average |Crr.| of features selected by $\boldsymbol{u}$

Figure 7.11: Properties of features selected in $\boldsymbol{X}$ by each one of the methods tested.

for all the 7 methods tested;

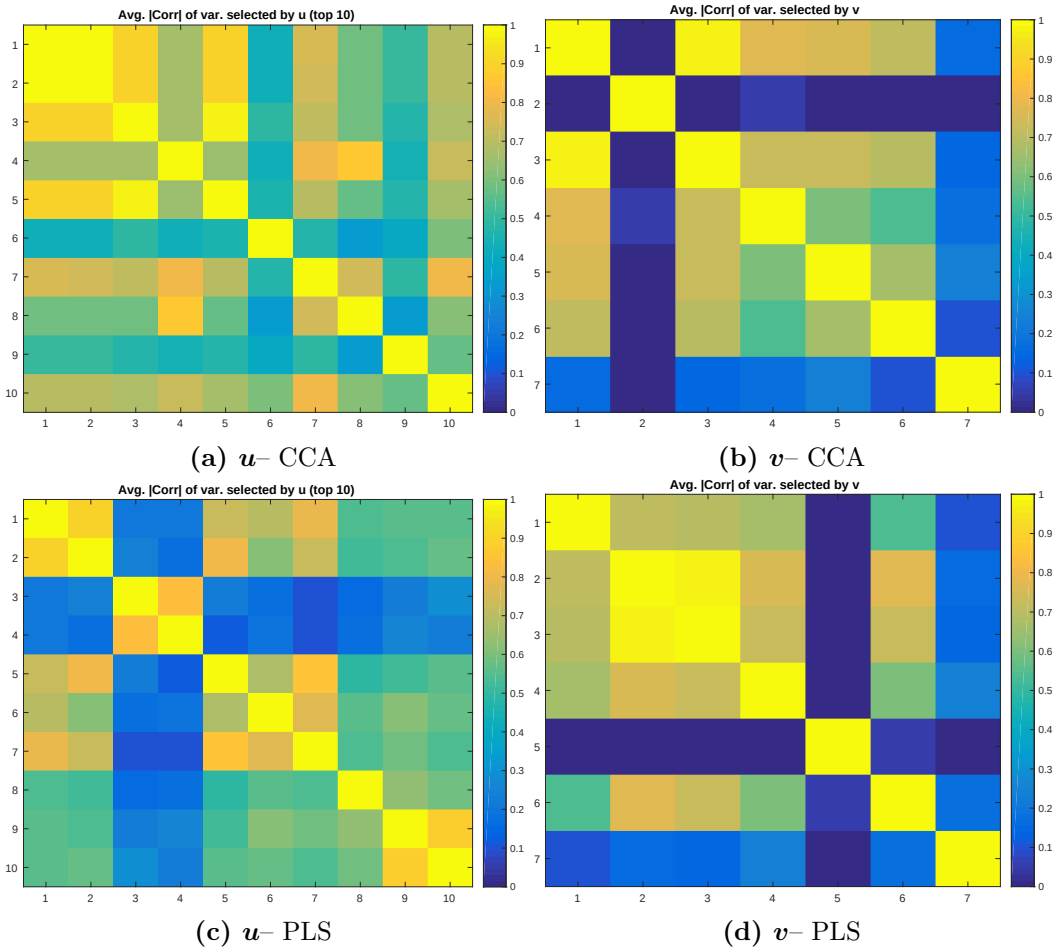
2. For each average weight vector $\boldsymbol{u}$ and $\boldsymbol{v}$ from each method, perform the following:

   (a) Re-order the features in $\boldsymbol{X}$ and $\boldsymbol{Y}$ according to the absolute weight values, e.g. for the average SPLS-PM weight vector $\boldsymbol{u}$, reorder the columns of $\boldsymbol{X}$, such that the first columns correspond to the ones with the largest $|u_j|$ while the last columns correspond to the ones with the smallest $|u_j|$;

   (b) Compute the correlation matrices using the reordered $\boldsymbol{X}$ and $\boldsymbol{Y}$ data matrices;

These matrices will contain information regarding the correlations between the features with the largest weights, which should provide some insights into the properties of the (S)CCA and (S)PLS methods.

Figure 7.12 shows the absolute correlation between the variables selected by each one of the non-sparse methods. One can see that, for the view containing the ROI data (Figure 7.12(a)), CCA tended to attribute weights with large absolute values to several highly correlated variables. While PLS has shown a similar effect (Figure 7.12(c)), the method also attributed high absolute weights to two ROIs which were very correlated between themselves (left and right inferior lateral ventricles),



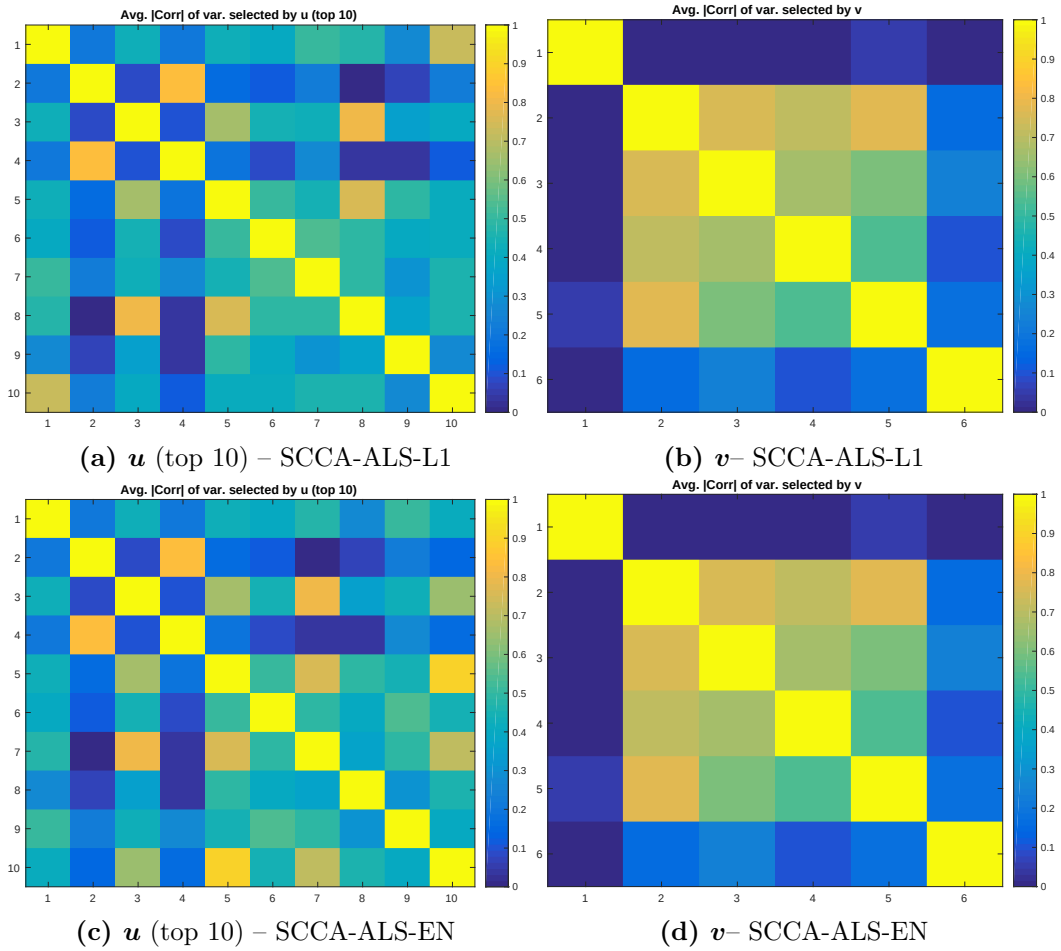
(a) $\boldsymbol{u}$– CCA **(b)** $\boldsymbol{v}$– CCA

(c) $\boldsymbol{u}$– PLS **(d)** $\boldsymbol{v}$– PLS

Figure 7.12: Correlation matrices between the variables selected by the average $\boldsymbol{u}$ and $\boldsymbol{v}$ for the non-sparse methods. The rows/columns were ordered by absolute weight value. The absolute correlation values are shown.

but had a low absolute correlation with the remaining ones.

Figure 7.13 shows the absolute correlation between the features selected by each one of the SCCA methods. One can see that the order of the rows/columns in the correlation matrices corresponding to the features selected by $\boldsymbol{v}$ (Figures 7.13(b) and 7.13(d)) did not change, which means that changing the penalty from a LASSO penalty (SCCA-ALS-L1) to an elastic net penalty (SCCA-ALS-EN), did not have a large effect on the relative weight of each feature in $\boldsymbol{Y}$. However, there were some top features selected by the SCCA-ALS-EN $\boldsymbol{u}$ vector (Figure 7.13(c)) which were more correlated between themselves than the top features in the SCCA-ALS-L1 $\boldsymbol{u}$ vector (Figure 7.13(a)). Although the effect was not very pronounced, it is consistent with what one would expect from an elastic net penalty, which tends to keep correlated features in the model. Unlike a LASSO penalty, which tends to down-weight/exclude



(a) $\boldsymbol{u}$ (top 10) – SCCA-ALS-L1 **(b)** $\boldsymbol{v}$– SCCA-ALS-L1

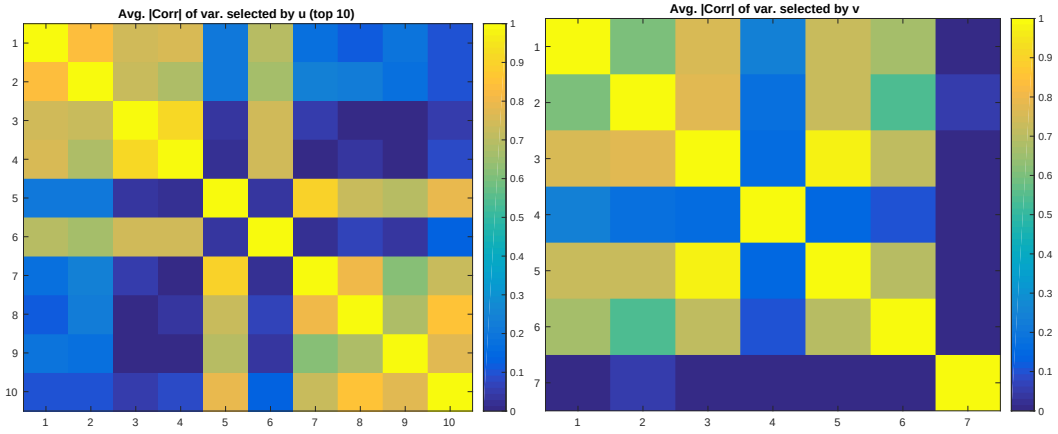
(c) $\boldsymbol{u}$ (top 10) – SCCA-ALS-EN **(d)** $\boldsymbol{v}$– SCCA-ALS-EN

Figure 7.13: Correlation matrices between the variables selected by the average $\boldsymbol{u}$ and $\boldsymbol{v}$ for the SCCA methods. The rows/columns were ordered by absolute weight value. The absolute correlation values are shown.

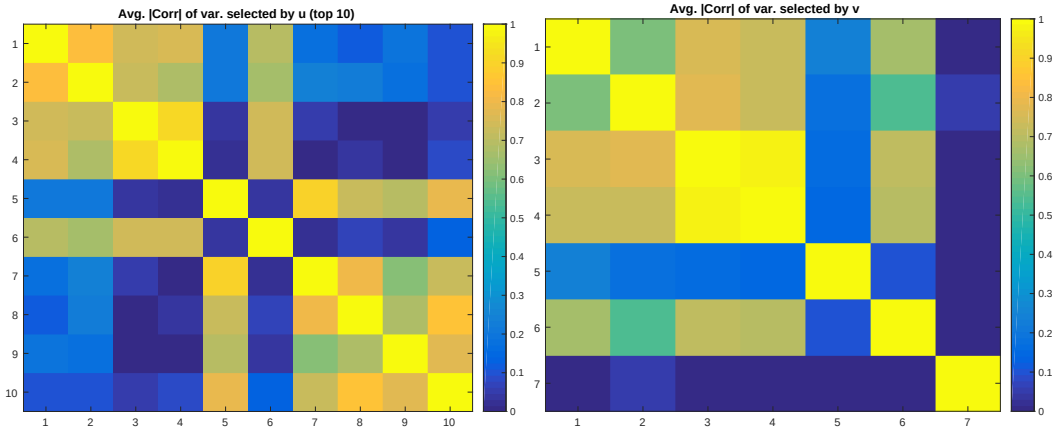
correlated features.

Figure 7.14 shows the same information as Figure 7.13, but for the SPLS methods. As one can see, the results were very different between the different SPLS approaches, more specifically, between the SPLS methods solved using ALS, and SPLS solved using the power method. By looking at the features selected by $\boldsymbol{u}$ (Figures 7.14(a), 7.14(c), and 7.14(e)), one can see that the methods based on ALS selected two groups of highly correlated features among the top 10. The features in each group were very correlated between other features in the same group, but had a very low correlation with features from the other group. This was not the case for SPLS-PM (Figure 7.14(e)), whose pattern was more similar to the one obtained by the SCCA methods (Figures 7.13(a), and 7.13(c)). However, the features selected by SPLS-PM (Figure 7.14(e)) were more correlated with each other than the ones selected by

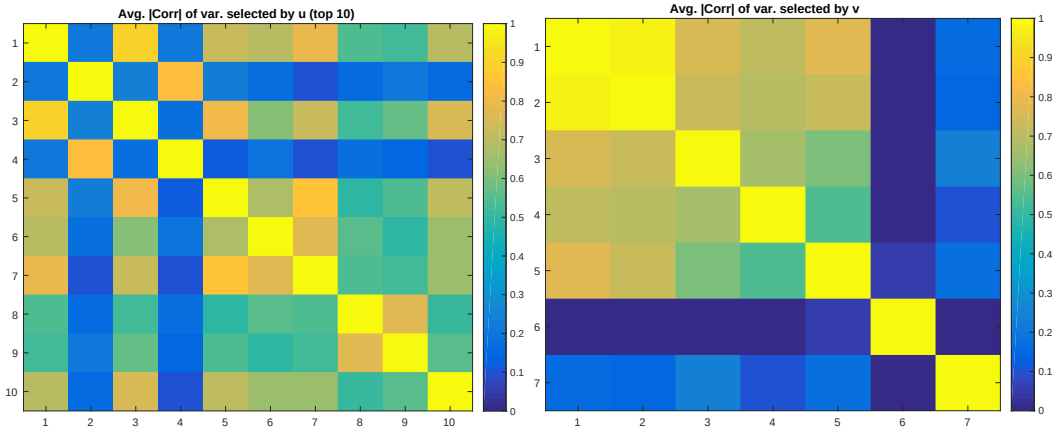
the SCCA methods (Figures 7.13(a) and 7.13(c)). This result was expected, since the features selected by SPLS were, on average, more correlated with each other (Figure 7.11).



(a) $\boldsymbol{u}$ (top 10) – SPLS-ALS-L1 **(b)** $\boldsymbol{v}$– SPLS-ALS-L1



(c) $\boldsymbol{u}$ (top 10) – SPLS-ALS-EN **(d)** $\boldsymbol{v}$– SPLS-ALS-EN

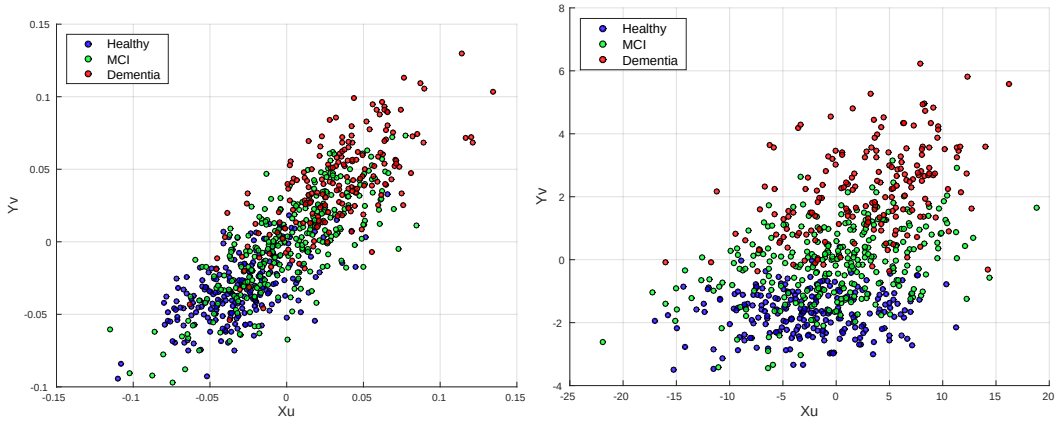


(e) $\boldsymbol{u}$ (top 10) – SPLS-PM **(f)** $\boldsymbol{v}$– SPLS-PM

Figure 7.14: Correlation matrices between the variables selected by the average $\boldsymbol{u}$ and $\boldsymbol{v}$ for the SPLS methods. The rows/columns were ordered by absolute weight value. The absolute correlation values are shown.

7.2.3.3 Projections

Figure 7.15 shows the projections of the subjects onto the computed weight vector pairs for each non-sparse method. These projections seem to form a continuous distribution from smaller to greater degrees of neuro-degeneration, which is consistent with the results shown in Chapter 6. Moreover, one can see that the projections provided by CCA (Figure 7.15(a)) seem to be more elongated, when compared with the projections provided by PLS (Figure 7.15(b)), i.e. they seem to be more aligned along a line. This is consistent with the results which showed that CCA was able to provide higher test correlations (Figure 7.8).

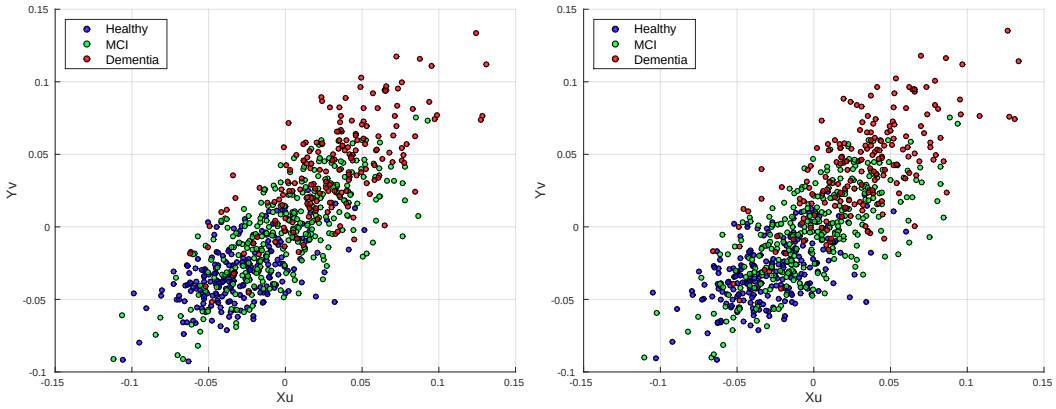


(a) CCA **(b)** PLS

Figure 7.15: Data projected onto the best weight vector pair for each one of the non-sparse methods tested.

Figure 7.16 shows the projections of the data onto the weight vector pairs computed by the SCCA methods. Just as in previous results, they seem to form a continuous distribution from smaller to greater degrees of neuro-degeneration. One can see that the projections are quite similar, which was expected, since all the results presented so far have shown that these two methods had a similar behaviour.

Figure 7.17 shows the projections obtained using the weight vector pairs computed by the SPLS methods. All the projections appeared to be more spread out than the ones given by the SCCA methods (Figure 7.16). This was expected, since the average correlations on the test sets for the SPLS methods were lower (Figure 7.8), which means that the projections onto the latent space given by $(\boldsymbol{Xu}, \boldsymbol{Yv})$ should align less along a line. Among the SPLS projections, SPLS-PM seemed to provide projections which aligned better along a line. Again, this was expected, due to the

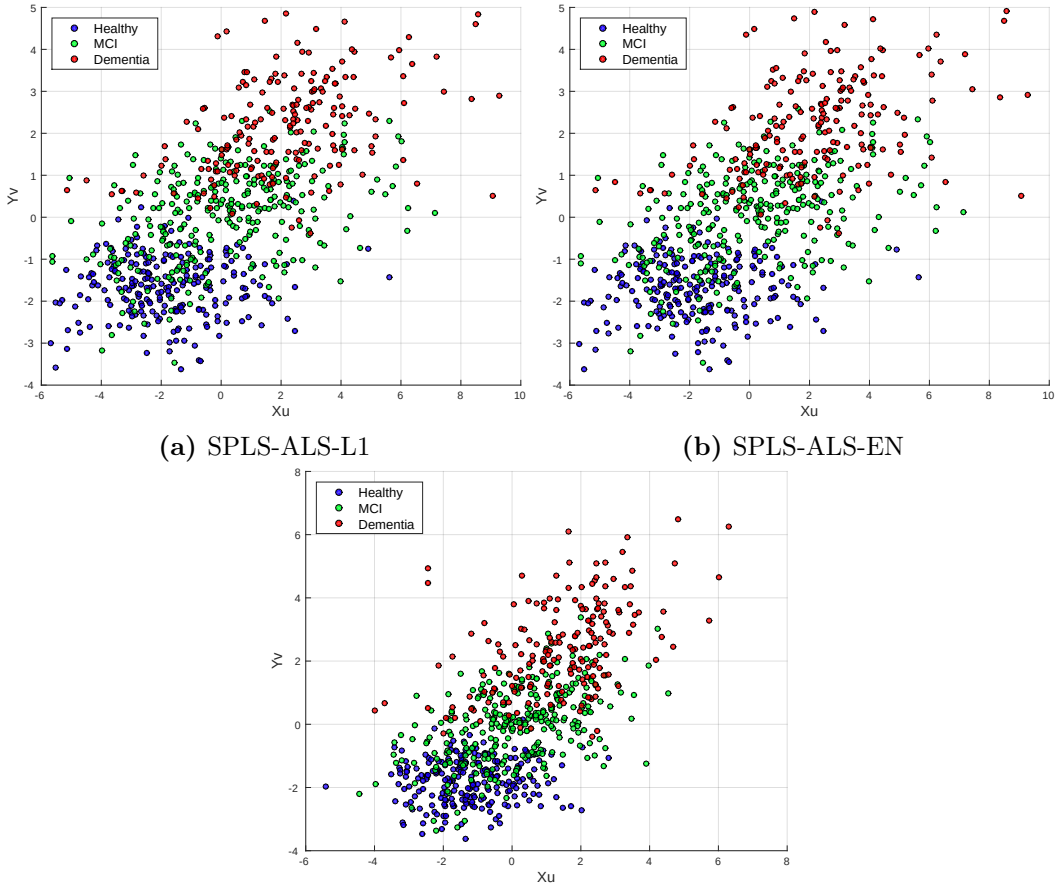


(a) SCCA-ALS-L1 **(b)** SCCA-ALS-EN

Figure 7.16: Data projected onto the best weight vector pair for each one of the SCCA methods tested.

fact that this method provided an average higher correlation in the test sets during the hyper-parameter selection steps (Figure 7.8).

One interesting property of the SPLS projections is that the separation between healthy controls and patients with dementia seems to be clearer along the $\boldsymbol{Yv}$ axis (Figure 7.17), when compared with the SCCA methods (Figure 7.16). This may be due to the fact that, for the SPLS methods, the top weights of $\boldsymbol{v}$ were associated with clinical scores, while age had the smallest absolute weight. In the case of the SCCA methods, age had the largest absolute weight. The greater contribution of the clinical variables (relative to the others) in the SPLS methods may be the driving force behind the separation along the $\boldsymbol{Yv}$ projection.



(a) SPLS-ALS-L1

(b) SPLS-ALS-EN

(c) SPLS-PM

Figure 7.17: Data projected onto the best weight vector pair for each one of the SPLS methods tested.

7.2.4 Conclusion

In this section, seven eigen-decomposition methods have been implemented and compared. By making these comparisons, this section aimed not only to determine which method provided the best test correlation, but also how consistent were the methods across the different hold-out splits, which features were selected by the different methods, and how different were the projections of the data onto the computed weight vector pairs.

All the methods provided solutions which were robust to perturbations in the data, and projections which where consistent with previous results in dementia datasets. From the seven methods tested, the SCCA methods have shown the best performance, by providing higher average correlations between the projections of the test data.

Although the use of ALS to solve SPLS has been previously proposed in the literature [Chi et al., 2013, Grellmann et al., 2015], its performance relative to SPLS using the power method had not been assessed. The experiments performed in this section have shown that by solving SPLS using ALS, one might not be solving the same optimisation problem as the one solved by SPLS using the power method. However, when compared with non-sparse PLS, all the SPLS methods were still able to provide a higher average correlation between the projections of the test data.

7.3 Chapter conclusion

This chapter has explored the use of ALS approaches to solve CCA and PLS problems.

The first section (Section 7.1) proposed a modification of the ALS algorithm for SCCA. The modification allowed it to converge more often, while providing comparable results for the optimal hyper-parameter combinations. The algorithm was tested using a dementia dataset, both in its original form and with added noisy features (i.e. in an ill-conditioned setting). The results have shown that ALS was indeed able to provide statistically significant results, which were robust to perturbations in the data, and consistent with previous results from the AD literature.

The second section (Section 7.2) compared seven eigen-decomposition methods. Our investigations showed that solving SPLS using an ALS approach, or solving SPLS using the power method, provided different results, which has not been previously shown. Unfortunately, the reasons for this behaviour were not very clear. In the future, further investigations into the different properties of these algorithms should be performed.

Despite the contributions presented in this chapter, its aim was mainly to serve as an introduction to Chapter 8, where ALS is used to solve a novel SCCA method.

Chapter 8

Primal-dual SCCA

8.1 Introduction

Although linear methods have been the focus of this thesis so far, it is important to acknowledge that, in some situations, they might not provide the best possible fit. Sometimes, the patterns of interest may be non-linear, which means that linear models will not have enough complexity to fit the data properly, i.e. they will underfit. One of the ways of addressing this issue is by implicitly mapping the data onto a higher dimensional space using kernel methods (Section 2.1.2). These allow one to fit a better model, by leveraging the non-linear relationships in the data. However, this will also decrease the interpretability of the model, which will now make use of all the features and its non-linear transformations at the same time. Moreover, if non-linear kernels are used, it is not straightforward to recover the weights in the original input space.

In order to use CCA with non-linear fits, a kernel version of CCA (KCCA) has been proposed in the literature (Section 3.2.2) [Lai and Fyfe, 2000]. However, the first implementations of KCCA were dual-dual formulations, i.e. the data on both views are transformed into kernel space $\{\boldsymbol{K}_x, \boldsymbol{K}_y\}$. There are situations in which one may wish to maintain one of the views in a primal formulation, e.g. to model the correlation between a linear combination of the features in one view with a non-linear combination of features in the other view. This version of KCCA, where one of the views is in the primal formulation and the other in the dual formulation, is referred to as *primal-dual KCCA* [Zheng et al., 2006, Van Vaerenbergh et al., 2008]. Note that an equivalent formulation of the primal-dual KCCA can be performed with a dual-dual KCCA with a linear and a non-linear kernel, however, it might be

computationally inefficient to perform if $n > p$ for the view with the linear kernel.

In addition to the potential computational advantages of primal-dual versions of KCCA, these may also be adapted to explore both sparse and non-linear properties of the data. Hardoon and Shawe-Taylor [2011] proposed a primal-dual version of SCCA, where sparsity was applied to both views, which means that correlations were found between a subset of features in the primal view, and a subset of kernel entries in the dual view (i.e. subset of samples).

Despite the popularity of this approach [Nybo et al., Rousu et al., 2013, Uurtio et al., 2015], the method proposed by Hardoon and Shawe-Taylor [2011] was designed for the specific type of problem that the authors were trying to address. The authors were interested in document retrieval, i.e. finding a sparse set of words in a language (primal view) correlated with a sparse set of documents in another language (dual view) [Hardoon and Shawe-Taylor, 2011]. However, in a clinical/neuroscience context, one is usually not interested in enforcing sparsity in the dual view. In this context, the aim is usually to detect an effect in a dataset containing a relatively small number of samples, therefore, one would usually want to use all the samples in the population, while still excluding features which do not contain relevant information.

The use of non-linear kernels is not as common in neuroimaging, although there are a few examples in the literature for specific low dimensional applications [Dong et al., 2015]. One of the main reasons is the fact that the dimensionality of the data is usually very high, therefore, providing extra complexity is usually not a good strategy, as this will tend to overfit the data. However, in a CCA setting, the dimensionality of one view may be much smaller than the dimensionality of the other view, e.g. if one view is comprised of a small number of clinical/demographic features, while the other contains a larger number of neuroimaging derived features. In these cases, one may be interested in modeling each view with very different degrees of complexity, e.g. by enforcing sparsity in the high dimensional view, while exploring non-linear relationships in the low dimensional view.

In this section, a novel primal-dual SCCA method is proposed, which enforces sparsity in the primal view, while including all the samples in the dual view, with the possibility of modeling them using a non-linear kernel. This was solved using an ALS approach. The method allows one to take advantage of the flexibility provided

by the use of non-linear kernels in one view, while still taking advantage of the interpretability provided by sparsity constraints in the other view. This may prove to be very useful when one is only interested on the interpretability of one view. By moving away from strictly linear CCA models, one might be able to find novel relationships between brain and behaviour, which may only be found by taking into account possible non-linear relationships in one of the views.

The proposed primal-dual SCCA method was tested using three types of kernels (linear, polynomial, and Gaussian), in order to determine if it is possible to find statistically significant correlations between neuroimaging data and clinical/demographic data, using both sparsity constraints and non-linear transformations. The following experimental setups were tested: using kernels on both views ($\{\boldsymbol{K}_x, \boldsymbol{K}_y\}$), using a kernel on the clinical/demographic view only ($\{\boldsymbol{X}, \boldsymbol{K}_y\}$), and using a kernel on the neuroimaging view only ($\{\boldsymbol{K}_x, \boldsymbol{Y}\}$).

8.2 Materials and Methods

8.2.1 Primal-dual SCCA

Let $\boldsymbol{M} \in \mathbb{R}^{n \times p}$ be a data matrix for the primal view, and $\boldsymbol{K} \in \mathbb{R}^{n \times n}$ be a kernel matrix for the dual view. The original primal-dual SCCA optimisation problem proposed by Hardoon and Shawe-Taylor [2011] was the following:

$$\begin{aligned} \underset{\boldsymbol{w},\boldsymbol{\alpha}}{\text{minimise}} \quad & \|\boldsymbol{M}\boldsymbol{w} - \boldsymbol{K}\boldsymbol{\alpha}\|_2^2 + \gamma_w \|\boldsymbol{w}\|_1 + \gamma_\alpha \|\widetilde{\boldsymbol{\alpha}}\|_1 \\ & \text{subject to} \\ & \|\boldsymbol{\alpha}\|_\infty = 1 \end{aligned} \tag{8.1}$$

where $\widetilde{\boldsymbol{\alpha}} = [\alpha_1, \ldots, \alpha_{j-1}, \alpha_{j+1}, \alpha_n]^\intercal$. In other words, there will be one entry α_j for witch $\alpha_j = 1$, and this value will be the maximum value in all the entries of $\boldsymbol{\alpha}$ (i.e. $\|\boldsymbol{\alpha}\|_\infty = 1$). The remaining entries ($\widetilde{\boldsymbol{\alpha}}$) will be subjected to a l_1-norm penalty.

The optimisation problem expressed in Equation 8.1 was solved using an algorithm specifically designed for it. Since the method proposed in this chapter solves a different optimisation problem, the description of the algorithm proposed by Hardoon and Shawe-Taylor [2011] is beyond the scope of this thesis.

The SCCA problem described in Equation 8.1 was originally proposed for mate-retrieval, where one tries to match a document written in one language to a paired

document in another language. In a clinical/neuroscience setting, this would result in finding a subset of features in $\boldsymbol{M}$ that would maximally correlate with a subgroup of subjects in $\boldsymbol{K}$. This problem is quite different from the ones commonly found in neuroscience and clinical applications, where one would usually be interested in including all the subjects in the model. Therefore, it does not make sense to enforce sparsity on $\boldsymbol{\alpha}$, as this means that only a subset of subjects will be selected. However, it is desirable to enforce sparsity on $\boldsymbol{w}$, as this will select a subset of features in $\boldsymbol{M}$. Therefore, a novel primal-dual SCCA formulation is proposed:

$$\begin{aligned} &\underset{\boldsymbol{w},\boldsymbol{\alpha}}{\text{minimise}} \ \|\boldsymbol{M}\boldsymbol{w} - \boldsymbol{K}\boldsymbol{\alpha}\|_2^2 + \gamma\boldsymbol{\alpha}^\intercal\boldsymbol{K}\boldsymbol{\alpha} \\ &\qquad\qquad \text{subject to} \\ &\|\boldsymbol{M}\boldsymbol{w}\|_2^2 = 1, \quad \|\boldsymbol{K}\boldsymbol{\alpha}\|_2^2 = 1, \quad |I_w| \leqslant p_w \end{aligned} \tag{8.2}$$

where $\boldsymbol{M} \in \mathbb{R}^{n\times p}$ is a data matrix; $\boldsymbol{K} \in \mathbb{R}^{n\times n}$ is a kernel matrix; $\boldsymbol{w} \in \mathbb{R}^{p\times 1}$ and $\boldsymbol{\alpha} \in \mathbb{R}^{n\times 1}$ are weight vectors; $I_w := \{i \mid w_j \neq 0\}$, and $p_w \in \{1, 2, \ldots, p\}$. Note that the term $\gamma\boldsymbol{\alpha}^\intercal\boldsymbol{K}\boldsymbol{\alpha}$ corresponds to a ridge penalty in kernel space (Equation 2.15), where γ is the hyper-parameter which controls the ridge penalty.

The optimisation problem described in Equation 8.2 can be solved using Algorithm 8.1.

Since the solutions will no longer be sparse in $\boldsymbol{\alpha}$, the `update` step is only performed on the sparse weight vector $\boldsymbol{w}$ (Step 9 of Algorithm 8.1).

In order to prevent overfitting, especially when $\boldsymbol{K}$ corresponds to a non-linear kernel, Step 5 of Algorithm 8.1 is performed using a kernel ridge regression (Equation 2.14).

Note that the optimisation problem will be quite different from the one proposed by Hardoon and Shawe-Taylor [2011], as sparsity is only enforced in one view, no l_∞-norm constraint is used, and a ridge penalty is applied to $\boldsymbol{\alpha}$.

8.2.2 Experiments

Three types of kernel functions (Section 2.1.2.1) were tested: linear kernel (LK), second degree polynomial kernel (PK), and Gaussian kernel (GK). These were tested in both views of the data. In order to distinguish between these methods, the following notation will be used: SCCA-X-Y, where X and Y correspond to the

Algorithm 8.1 Primal-dual SCCA using ALS. The `update` function is described in Algorithm 7.2.

Set $\boldsymbol{w}^{(0)}$ and $\boldsymbol{\alpha}^{(0)}$ equal to the first singular vector pair of $\boldsymbol{M}^\intercal\boldsymbol{K}$.
Set $\delta = 0.5$
repeat

  $\boldsymbol{b} \leftarrow \boldsymbol{M}\boldsymbol{w}^{(i)}$
  Solve $\boldsymbol{\alpha}^{(i+1)} \leftarrow \arg\min_{\boldsymbol{\alpha}} \|\boldsymbol{b} - \boldsymbol{K}\boldsymbol{\alpha}\|_2^2 + \gamma\boldsymbol{\alpha}^\intercal\boldsymbol{K}\boldsymbol{\alpha}$
  $\boldsymbol{\alpha}^{(i+1)} \leftarrow \boldsymbol{\alpha}^{(i+1)} / \|\boldsymbol{K}\boldsymbol{\alpha}^{(i+1)}\|_2$

  $\boldsymbol{b} \leftarrow \boldsymbol{K}\boldsymbol{\alpha}^{(i+1)}$
  Solve $\boldsymbol{w}^{(i+1)} \leftarrow \arg\min_{\boldsymbol{w}} \|\boldsymbol{b} - \boldsymbol{M}\boldsymbol{w}\|_2^2 + \gamma_1 \|\boldsymbol{w}\|_1$, with γ_1 such that $|I_w| \approx p_w$
  $\boldsymbol{w}^{(i+1)} \leftarrow \text{UPDATE}(\boldsymbol{w}^{(i+1)}, \boldsymbol{w}^{(i)}, \boldsymbol{M}, \delta)$

  if ALS is oscillating **then**
    $\delta \leftarrow \delta/2$
  end if

  $i \leftarrow i + 1$
until convergence
return $\boldsymbol{w}$, $\boldsymbol{\alpha}$

type of regression that was applied to the corresponding view, e.g. "SCCA-PK-L1" denotes a primal-dual SCCA with a polynomial kernel on $\boldsymbol{X}$ and a LASSO on $\boldsymbol{Y}$, where $\boldsymbol{X}$ denotes the view containing the neuroimaging derived features (ROIs) and $\boldsymbol{Y}$ denotes the view containing the clinical/demographic data.

Note that the CCA methods using only kernels (KCCA-LK-LK, KCCA-PK-PK, and KCCA-GK-GK) are not exactly the same penalised KCCA methods described in Equation 3.11. The KCCA versions used in this chapter are equivalent to solving a CCA problem with a penalty on the l_2-norm of $\boldsymbol{u}$ and $\boldsymbol{v}$, while the classical regularised KCCA method described in Equation 3.11 penalises the solution by solving an optimisation problem which is an interpolation between CCA and PLS.

All the experiments were performed using the same dataset and the same validation procedure as the ones in Chapter 7. The hyper-parameters tested for the LASSO regression step were the same as in the experiments from Section 7.2 (40 equidistant points $p_w \in [0.01p, p]$), and the ridge penalty hyper-parameters tested were: $\gamma \in \{10^{-3}, 10^{-2}, \ldots, 10^{8}\}$. The σ hyper-parameter of the Gaussian kernel (Equation 2.11) was set using the "median trick", which consists of setting the value of σ equal to the median of the squared distance between the training points [Song

et al., 2010]. For the polynomial kernel, the R hyper-parameter was set to be equal to the number of features (i.e. p or q, depending on the view in which the kernel was applied). As mentioned in Section 2.1.2.1, R controls the influence of the linear terms in the polynomial kernel in order to prevent overfitting (i.e. the larger it is, the more influence the linear terms have). When performing a feature mapping using all the polynomial terms of degree d, the number of terms may grow very large (as shown in Equation 2.13), therefore, one may want to scale R with the number of features in the data matrix, so that the influence of the linear terms is balanced with the introduction of a large amount of new non-linear terms.

The aim of this chapter is not to demonstrate a clear performance improvement by using primal-dual SCCA in this particular dataset, but to provide a proof of concept that the proposed method is able to compute solutions which are both robust to perturbations in the data, and provide results which align with previous findings in the AD literature.

8.3 Results and Discussion

8.3.1 Correlations

All the methods provided statistically significant results ($p < 0.005$). For the complete list of correlations on the hold-out datasets and corresponding p-values, please refer to Appendix D.1.

Figure 8.1 shows the average optimal correlation obtained for each one of the 10 hyper-parameter selection steps. With the exception of KCCA-GK-GK and SCCA-GK-L1, all the methods provided similar performances, which were also close to the ones obtained in Section 7.1. These results show that primal-dual SCCA using non-linear kernels was able not only to obtain statistically significant results, but also to provide correlations on test data comparable to primal-primal SCCA approaches (Chapter 7).

The non-sparse methods (i.e. using KCCA) provided correlations above 0.7, with the exception of KCCA-GK-GK, which was one of the worst performing methods (above 0.5), the other one being SCCA-GK-L1. Interestingly, this was not the case for SCCA-L1-GK, which suggests that using a Gaussian kernel to model the relationships in the clinical/demographic data ($\boldsymbol{Y}$) is a better approach than using it to model the ROI data in $\boldsymbol{X}$ (SCCA-GK-L1). This result supports the idea that mapping the

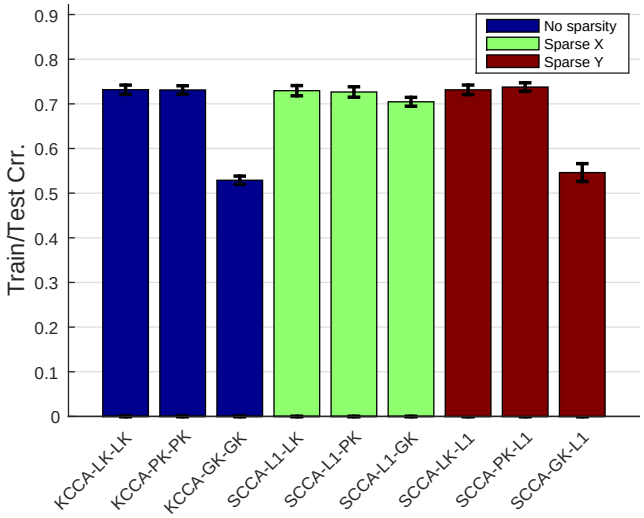


Figure 8.1: Average optimal test correlation across the 10 different data splits.

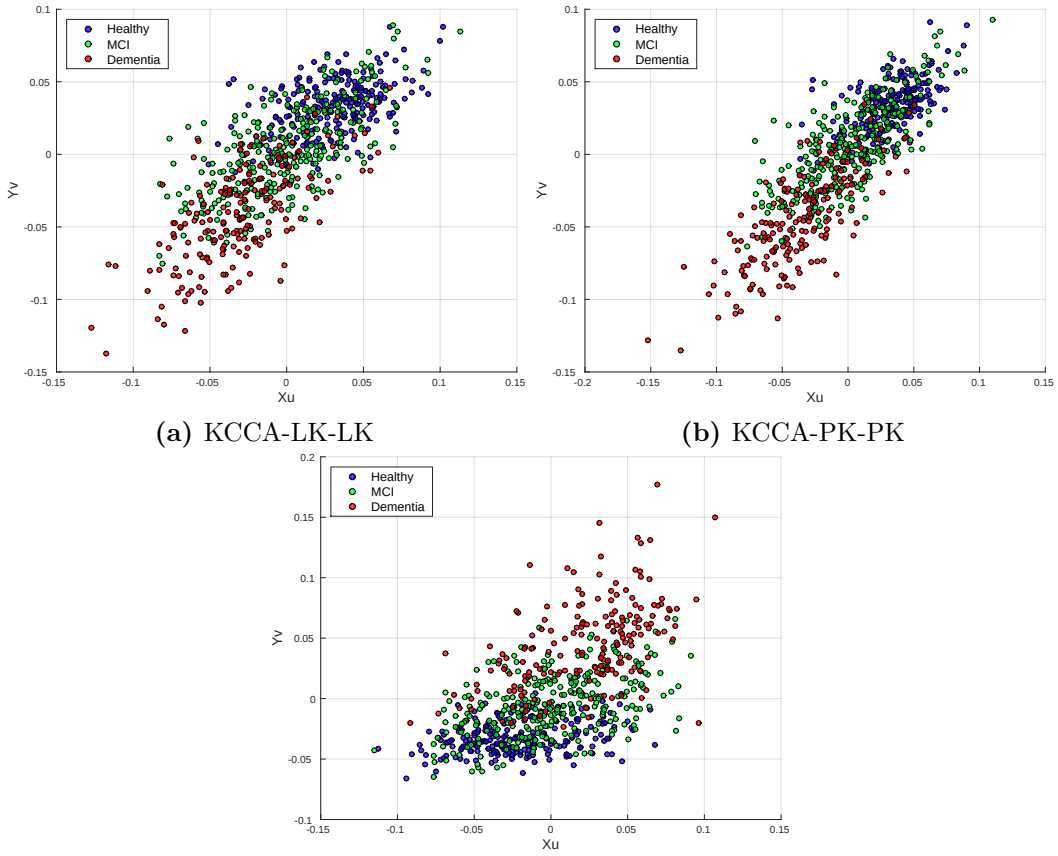
view with the higher dimensionality ($\boldsymbol{X}$) into a even higher dimensional space, might not be a good approach, however, mapping the view with the lower dimensionality ($\boldsymbol{Y}$) may provide good results.

The plots showing the average test correlation for the different hyper-parameter combinations (i.e. "hyper-parameter space") for all the methods tested in this chapter are provided in Appendix D.2.

8.3.2 Projections

Just as in previous chapters, the data were projected onto the best weight vector pairs computed by each method. Figure 8.2 shows the projections of the data onto the weight vectors computed using the non-sparse methods: KCCA-LK-LK, KCCA-PK-PK, and KCCA-GK-GK. As one can see, these show a distribution from individuals with higher to lower degrees of neuro-degeneration, which is consistent with the results from Chapters 6 and 7. The results presented in Figure 8.2 show that the projections provided by KCCA-GK-GK were less aligned along a line (when compared with the other KCCA approaches). These results are consistent with Figure 8.1, which shows that KCCA-GK-GK provided lower average correlations between the projections of test data.

Figure 8.3 shows the projections of the data onto the weight vector pairs computed by the primal-dual SCCA methods: SCCA-L1-LK, SCCA-L1-PK, SCCA-L1-GK, SCCA-LK-L1, SCCA-PK-L1, and SCCA-GK-L1. Just as in previous results, the projections show a distribution from individuals with higher to lower degrees



(a) KCCA-LK-LK

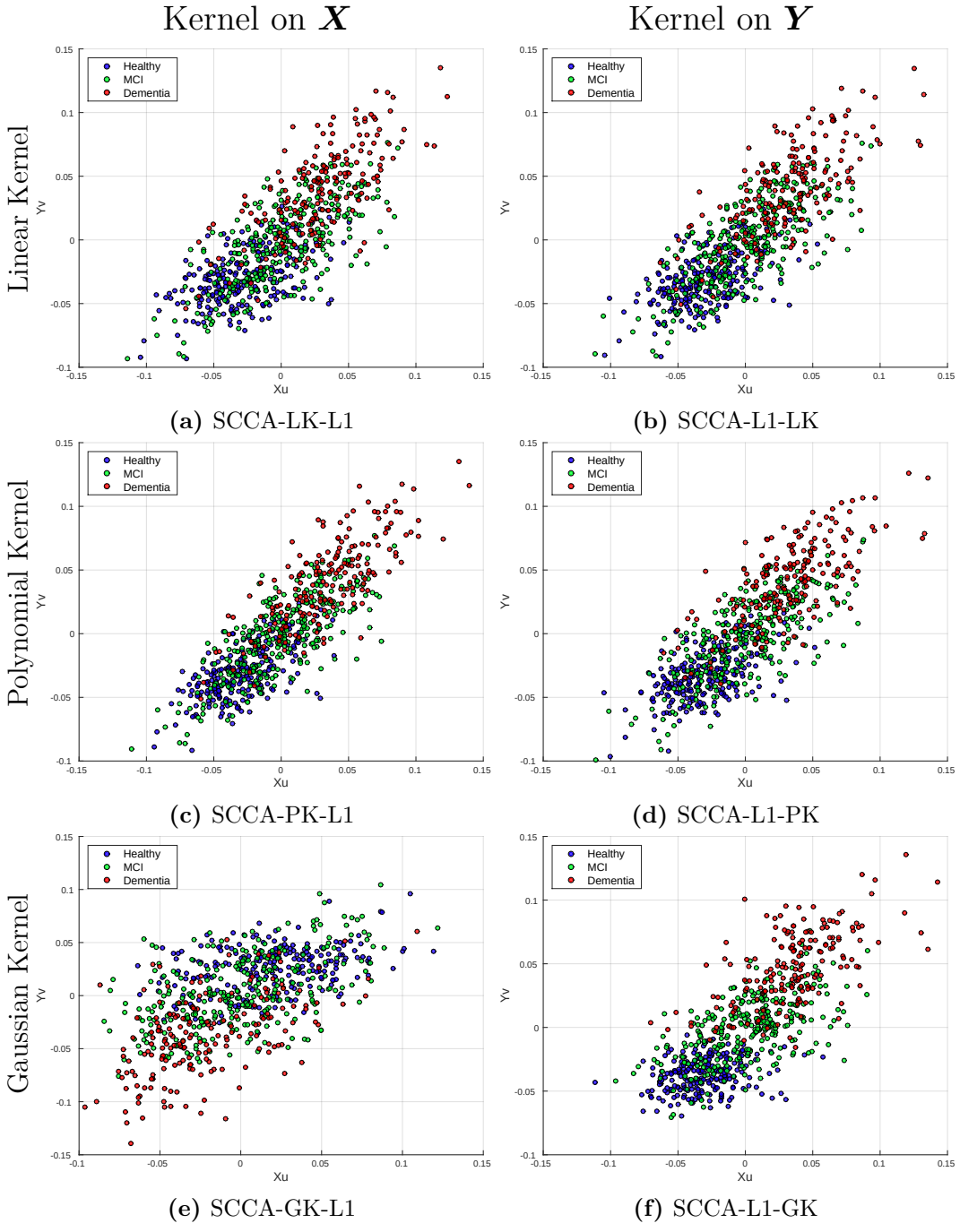
(b) KCCA-PK-PK

(c) KCCA-GK-GK

Figure 8.2: Projections of the subjects onto the weight vectors computed using non-sparse methods.

of neuro-degeneration, but no projection provided a clear separation between the groups. Moreover, as expected by the results presented in Figure 8.1, the projections provided by SCCA-GK-L1 were less aligned along a line.

One of the possible applications of these methods is to try to predict which subjects will convert from MCI to AD. Figure 8.4 shows the projections of the MCI subjects onto the computed weight vector pairs, encoding which subjects converted from MCI to AD after six months. As one can see, there was no clear separation between these two groups for all methods tested. The plots for the non-sparse methods are presented in Appendix D.3.

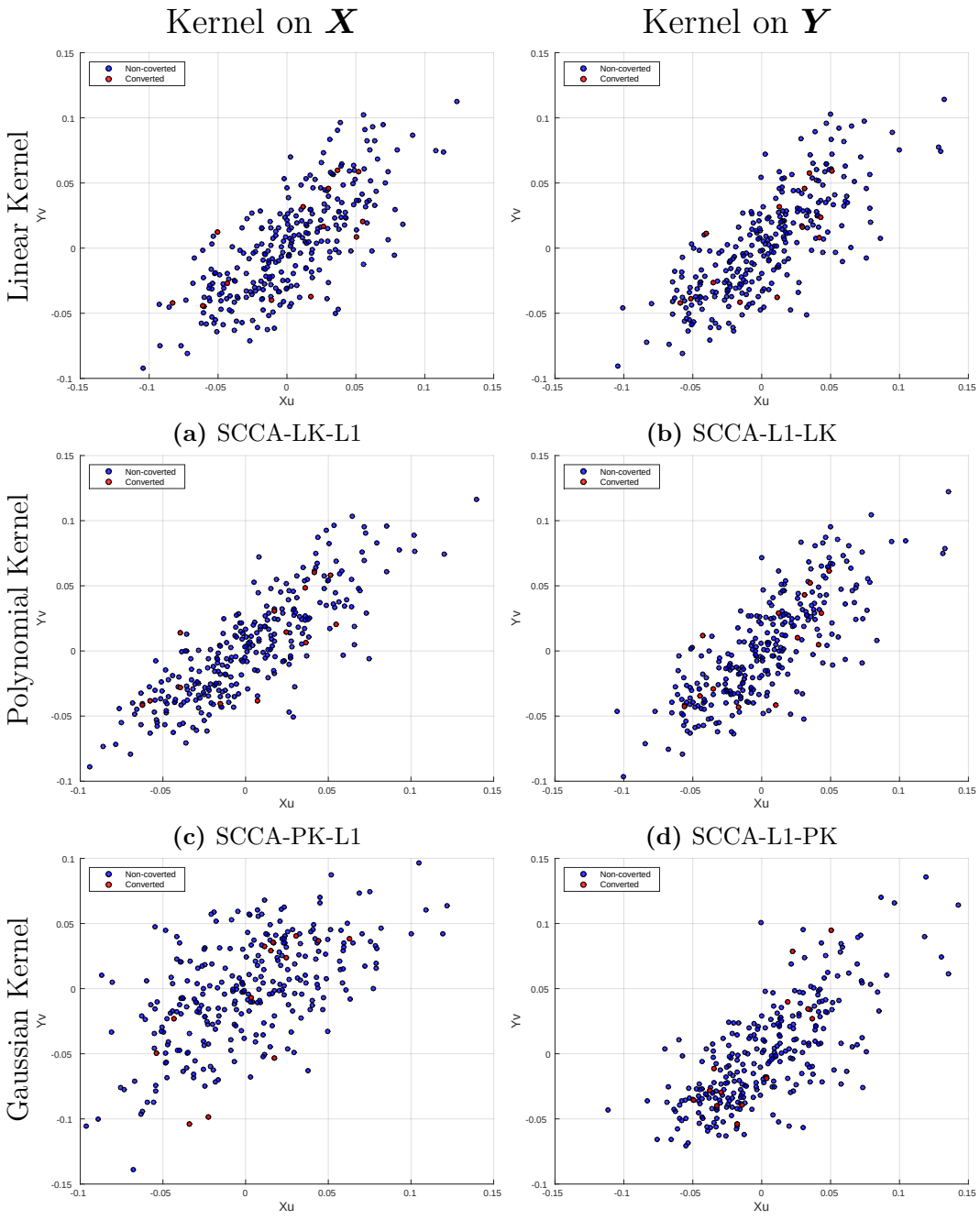


(a) SCCA-LK-L1 **(b)** SCCA-L1-LK

(c) SCCA-PK-L1 **(d)** SCCA-L1-PK

(e) SCCA-GK-L1 **(f)** SCCA-L1-GK

Figure 8.3: Projections of the subjects onto the primal-dual SCCA weight vectors. Each column corresponds to applying a kernel on either $\boldsymbol{X}$ (ROI data) or $\boldsymbol{Y}$ (clinical/demographic data), i.e. the left column corresponds to using $\boldsymbol{K}_x$ and the right column corresponds to using $\boldsymbol{K}_y$. Each row corresponds to a different type of kernel.



(a) SCCA-LK-L1 **(b)** SCCA-L1-LK

(c) SCCA-PK-L1 **(d)** SCCA-L1-PK

(e) SCCA-GK-L1 **(f)** SCCA-L1-GK

Figure 8.4: Subject conversion from MCI to AD after 6 months. Each column corresponds to applying a kernel on either $\boldsymbol{X}$ (ROI data) or $\boldsymbol{Y}$ (clinical/demographic data), i.e. the left column corresponds to using $\boldsymbol{K}_x$ and the right column corresponds to using $\boldsymbol{K}_y$. Each row corresponds to a different type of kernel.

8.4 Conclusion

In this chapter, a novel primal-dual SCCA method was proposed and tested using different types of kernels for each view.

Our experiments have shown that primal-dual SCCA was able to provide statistically significant results whose performance on test data was comparable with primal-primal SCCA (Chapter 7). Moreover, the projections provided by the method showed that the subjects lied along a continuous distribution, from healthy subjects to subjects suffering from greater degrees of neuro-degeneration, which is consistent with previous results from Chapters 6 and 7.

Although the proposed primal-dual SCCA method did not out-perform other CCA approaches for this particular dataset, this chapter still provided a proof of concept that the proposed method was able to compute weight vector pairs which were both robust to perturbations in the data, and provided results which align with previous findings.

Future work should explore the use of primal-dual SCCA for other low dimensional neuroimaging datasets, especially in problems which linear approaches have failed to find strong associations between the views. These may also include problems with other types of data, which may provide better features to be used with non-linear kernels, including: brain ROI features based on shape, brain connectivity data [Smith et al., 2015], Cerebrospinal Fluid (CSF) biomarkers [Young et al., 2014], and genetics [Altmann et al., 2014].

Chapter 9

General Conclusions

9.1 Summary of the main contributions

The use of computational approaches to study neuroimaging data has been increasing for several years, starting with mass-univariate statistical models [Friston et al., 1994], to more recent applications using machine learning [Ecker et al., 2010, Mourão-Miranda et al., 2005, Nouretdinov et al., 2011, Orrù et al., 2012, Rao et al., 2011, Klöppel et al., 2008]. However, most machine learning studies still rely on the labeling of the subjects, constraining the study of several brain diseases within a paradigm of pre-defined clinical labels, which have shown to be unreliable in some cases [Insel et al., 2010].

The lack of understanding regarding the associations between brain and behaviour presents itself as an interesting challenge for more exploratory machine learning approaches, which could potentially help in the study of diseases whose clinical labels have limitations. The aim of this project was to explore the possibility of using eigen-decomposition approaches to find multivariate associative effects between brain structure and behaviour in an exploratory way.

The first contribution of this thesis (Chapter 4) was to show the advantages of using an alternative matrix deflation approach with sparse eigen-decomposition methods, more specifically, Sparse Partial Least Squares (SPLS) [Monteiro et al., 2014]. The proposed deflation approach was able to provided sparse weight vectors which were approximately orthogonal, providing a much better approximation than the deflation strategy proposed in the original publication [Witten et al., 2009].

After demonstrating the advantages of the alternative deflation strategy, the second contribution was to use SPLS with the modified deflation step to model

the association between clinical/demographic features and brain structure using sparsity in both views, i.e. without relying on the *a priori* assumption that the multivariate associative effects were not sparse in the clinical/demographic view (Chapter 5) [Monteiro et al., 2015].

The type of analysis performed in Chapter 5 was taken a step further in Chapter 6. In this chapter, a multiple hold-out framework was proposed, which allowed for the detection of robust multivariate associative effects between brain structure and individual questionnaire items, while being computationally less intensive than nested-CV [Monteiro et al., 2016].

Even though the correlation was used as a metric to evaluate the model in Chapter 6, its use with SPLS does present some issues. Therefore, one should look into alternative sparse eigen-decomposition methods to be used with this metric, such as, Sparse Canonical Correlation Analysis (SCCA). Chapter 7 proposed an adaptation of the ALS method to solve several sparse eigen-decomposition problems, presenting comparisons between seven variants of CCA and PLS, which have not been previously shown in the literature.

Despite the popularity of strictly linear models in neuroscience, one of their limitations is their assumption that the patterns of interest are linear. However, this may not be the case, which means that these methods will not have enough flexibility to detect non-linear associations between brain and behaviour. Nevertheless, the choice of whether non-linear associations should be taken into account has to be based, among other things, on the dimensionality of the data. Therefore, a novel primal-dual SCCA method was proposed in Chapter 8, which allowed one to enforce sparsity in views where the dimensionality is high, while simultaneously exploring non-linear relationships in views where the dimensionality is lower.

9.2 Limitations and directions for future research

Despite the encouraging results provided by exploratory eigen-decomposition methods, one of the disadvantages of these approaches is that they may require large amounts of data, when compared with commonly used supervised classification approaches. However, the current trend in neuroscience is to fund projects which aim to collect increasingly larger datasets, and making them available to the scientific community [Thompson et al., 2014, Sudlow et al., 2015, nsp], therefore, exploratory

methods may start to gain more visibility in future neuroscience/clinical applications.

Some of these future applications could be built on some of the contributions of this thesis, namely, one could apply the framework proposed in Chapter 6 to datasets in which stratification is the main challenge. These include not only psychiatric datasets, but also datasets from the general population. One of the possible datasets to explore in future work is the one acquired by the NeuroScience in Psychiatry Network (NSPN). This consists of a joint venture between the University of Cambridge and University College London to study the development of the adolescent brain, containing data from over 2000 subjects [nsp].

As the datasets available to the scientific community start to grow, so should the complexity of methods used to analyse them. By adapting these methods to explore non-linearities in the data, one could potentially extract more information from the increasingly larger datasets available. Some contributions in this direction have been made in Chapter 8. Although the proposed non-linear methods did not show an improvement relative to the linear methods, the proposed approach was able to provide robust results. These methods could potentially enable the identification of non-linear associations using other types of data, including: brain ROI features based on shape, brain connectivity data [Smith et al., 2015, Rosa et al., (in preparation)], CSF biomarkers [Young et al., 2014], and genetics [Altmann et al., 2014].

By applying the strategies proposed in this thesis, future work could explore the potential for stratifying patients based on their associations between brain and behaviour, which will hopefully help shift the paradigm used to study several brain disorders.

Appendix A

Proofs

A.1 Projection deflation vs. PLS Mode-A deflation

Projection deflation (Chapter 4):

$$\boldsymbol{X}_{h+1} = \boldsymbol{X}_h \left(\boldsymbol{I} - \boldsymbol{u}_h \boldsymbol{u}_h^\intercal\right) = \boldsymbol{X}_h - \boldsymbol{X}_h \boldsymbol{u}_h \boldsymbol{u}_h^\intercal \tag{A.1}$$

PLS Mode-A deflation (Section 3.3):

$$\boldsymbol{X}_{h+1} = \boldsymbol{X}_h - \boldsymbol{\xi}_h \boldsymbol{\gamma}_h^\intercal$$

where:

$$\boldsymbol{\xi}_h = \boldsymbol{X}_h \boldsymbol{u}_h \quad \text{and} \quad \boldsymbol{\gamma}_h = \boldsymbol{X}_h^\intercal \frac{\boldsymbol{\xi}_h}{\boldsymbol{\xi}_h^\intercal \boldsymbol{\xi}_h}$$

$\boldsymbol{\gamma}_h$ can be re-written as:

$$\boldsymbol{\gamma}_h = \boldsymbol{X}_h^\intercal \frac{\boldsymbol{\xi}_h}{\boldsymbol{\xi}_h^\intercal \boldsymbol{\xi}_h} = \boldsymbol{X}_h^\intercal \frac{\boldsymbol{X}_h \boldsymbol{u}_h}{\boldsymbol{u}_h^\intercal \boldsymbol{X}_h^\intercal \boldsymbol{X}_h \boldsymbol{u}_h}$$

Thus, PLS Mode-A deflation can be re-written as:

$$\boldsymbol{X}_{h+1} = \boldsymbol{X}_h - \boldsymbol{X}_h \boldsymbol{u}_h \left(\frac{\boldsymbol{X}_h^\intercal \boldsymbol{X}_h \boldsymbol{u}_h}{\boldsymbol{u}_h^\intercal \boldsymbol{X}_h^\intercal \boldsymbol{X}_h \boldsymbol{u}_h} \right)^\intercal \tag{A.2}$$

As one can see, projection deflation (A.1) and PLS Mode-A deflation (A.2) are equivalent iff $\boldsymbol{X}_h^\intercal \boldsymbol{X}_h = \boldsymbol{I}$. As noted by Wegelin [2000], the vectors obtained by PLS-SVD are the singular vectors of $\boldsymbol{X}^\intercal \boldsymbol{Y}$. However, this is not the case for all types of PLS, including PLS Mode-A.

A.2 PLS vs. PLS-ALS

Each update step in the PLS-ALS method is performed by solving the regression problems expressed in Equation 7.4, which can be re-written as follows:

$$\boldsymbol{u} \leftarrow \arg\min_{\boldsymbol{u}} \|\boldsymbol{b}_v - \boldsymbol{I}_u \boldsymbol{u}\|_2^2 \quad \text{and} \quad \boldsymbol{v} \leftarrow \arg\min_{\boldsymbol{v}} \|\boldsymbol{b}_u - \boldsymbol{I}_v \boldsymbol{v}\|_2^2$$

where $\boldsymbol{b}_v = \boldsymbol{X}^\intercal \boldsymbol{Y} \boldsymbol{v}$, $\boldsymbol{b}_u = (\boldsymbol{X}^\intercal \boldsymbol{Y})^\intercal \boldsymbol{u}$, and $\boldsymbol{I}_u \in \mathbb{R}^{p \times p}$ and $\boldsymbol{I}_v \in \mathbb{R}^{q \times q}$ are identity matrices.

The solution for a non-penalised least squares regression is expressed in Equation 2.7, which can be re-written for this case as:

$$\begin{aligned} \boldsymbol{u} &\leftarrow (\boldsymbol{I}_u^\intercal \boldsymbol{I}_u)^{-1} \boldsymbol{I}_u \boldsymbol{b}_v \quad \text{and} \quad \boldsymbol{v} \leftarrow (\boldsymbol{I}_v^\intercal \boldsymbol{I}_v)^{-1} \boldsymbol{I}_v \boldsymbol{b}_u \\ \boldsymbol{u} &\leftarrow \boldsymbol{X}^\intercal \boldsymbol{Y} \boldsymbol{v} \quad \text{and} \quad \boldsymbol{v} \leftarrow (\boldsymbol{X}^\intercal \boldsymbol{Y})^\intercal \boldsymbol{u} \end{aligned}$$

which are the exact same update steps used in PLS-SVD (Algorithm 3.2). Thus, both methods are equivalent.

Appendix B

Chapter 6

B.1 Mini-Mental State Examination

Table B.1 gives a brief description of the questions/tasks performed during the MMSE [Folstein et al., 1975].

Table B.1: MMSE questions/tasks.

Domain	Question/Task
Orientation	1. What is today's date? 2. What year is it? 3. What month is it? 4. What day of the week is today? 5. What season is it? 6. What is the name of this hospital? 7. What floor are we on? 8. What town or city are we in? 9. What county (district) are we in? 10. What state are we in?
Registration	11. Name object (ball) 12. Name object (flag) 13. Name object (tree) 13a. Number of trials
Att. & Calc.	14. D 15. L 16. R 17. O 18. W
Recall	19. Recall Ball 20. Recall Flag 21. Recall Tree
Language	22. Show a wrist watch and ask "What is this?" 23. Show a pencil and ask "What is this?" 24. Repeat a sentence. 25. Takes paper in right hand. 26. Folds paper in half. 27. Puts paper on floor. 28. Read and obey a command ("Close your eyes"). 29. Write a sentence. 30. Copy design.

B.2 Hyper-parameter optimisation

When performing the hyper-parameter optimisation step, the average absolute correlation was computed for each hyper-parameter combination. Figure B.1 shows the values of the mean absolute correlation obtained for the first weight vector pair, in all 10 random splits of the data. As one can see, the surfaces are smooth and fairly consistent across splits, which further supports the reliability of the model. The corresponding results for the second weight vector pair can be seen in Figure B.2.

In both multivariate associative effects, the relaxation of the sparsity constraint in the neuroimaging view (c_u) decreased the mean absolute correlation between the projections, which means that both effects are better expressed by a relatively small subset of the image voxels. On the other hand, the optimal constraint on the clinical view (c_v) was in the middle of the hyper-parameter range on the first effect, but more relaxed on the second effect (closer to the upper limit of the hyper-parameter range). Which means that the first effect will be described by a smaller subset of clinical variables than the second.

This behaviour was not observed if PLS Mode-A deflation was used instead of projection deflation (Figure B.3). In this case, the surfaces were not smooth, the average absolute correlations were lower, and the maximum hyper-parameter combination changes quite a bit when a different split of the data is used. This is consistent with the results that showed a lower average absolute correlation on the hold-out datasets, and lack of statistical significance associated with these weight vectors.

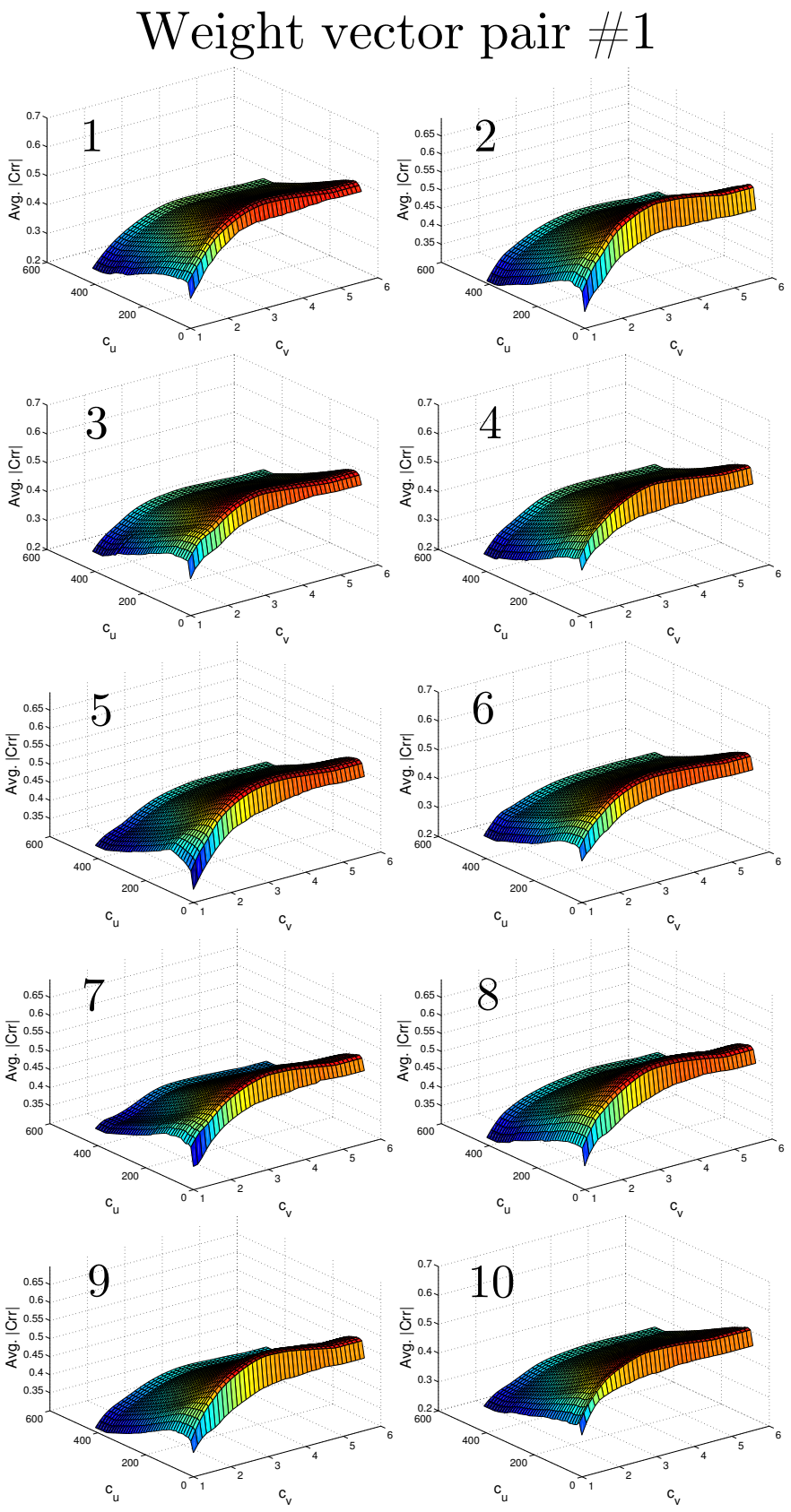


Figure B.1: Mean absolute correlation value computed for each split of the data (indicated by the numbers on the top left corners) for the first weight vector pair during the hyper-parameter optimisation step.

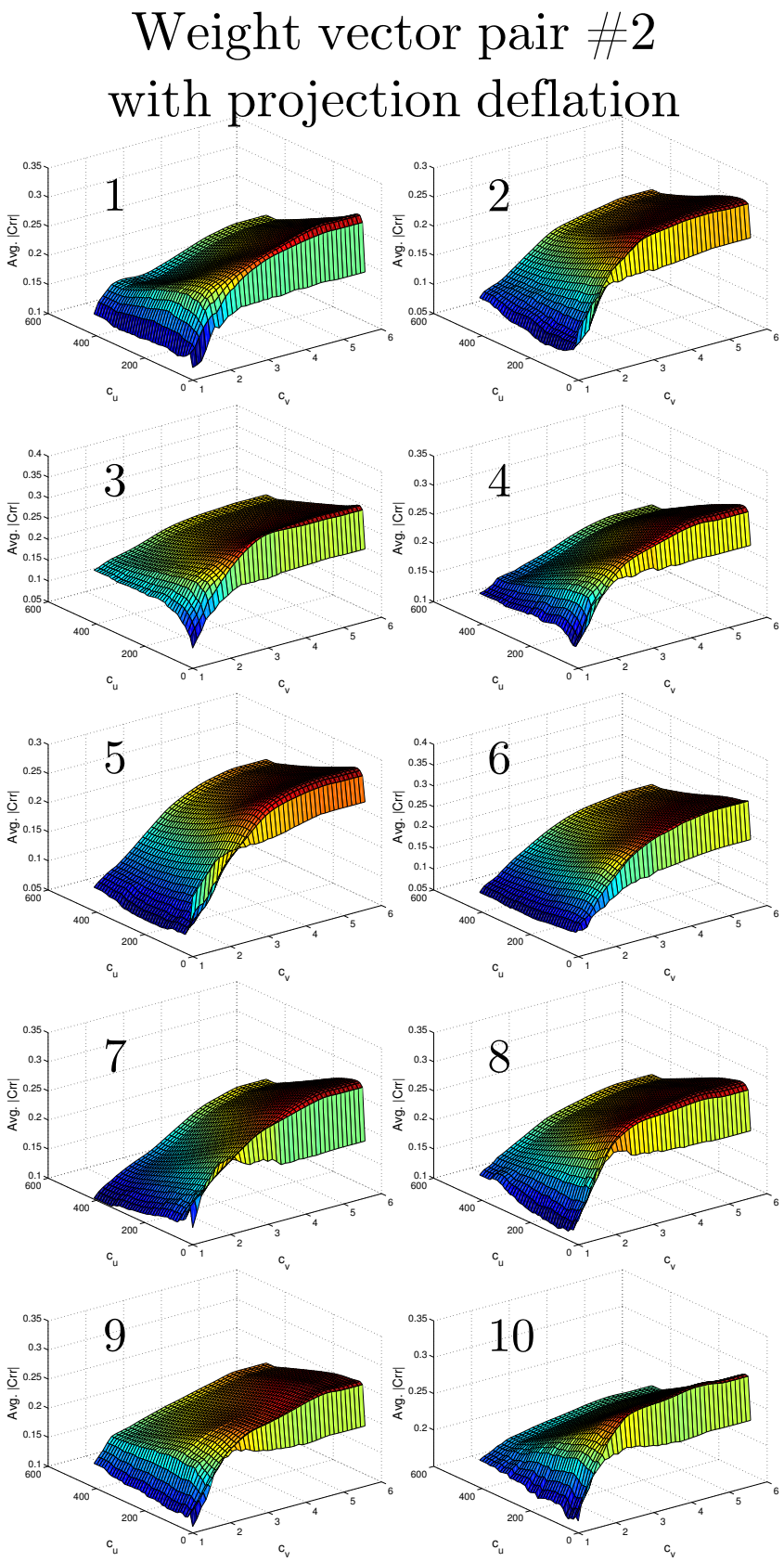


Figure B.2: Mean absolute correlation value computed for each split of the data (indicated by the numbers on the top left corners) for the second weight vector pair during the hyper-parameter optimisation step, using projection deflation.

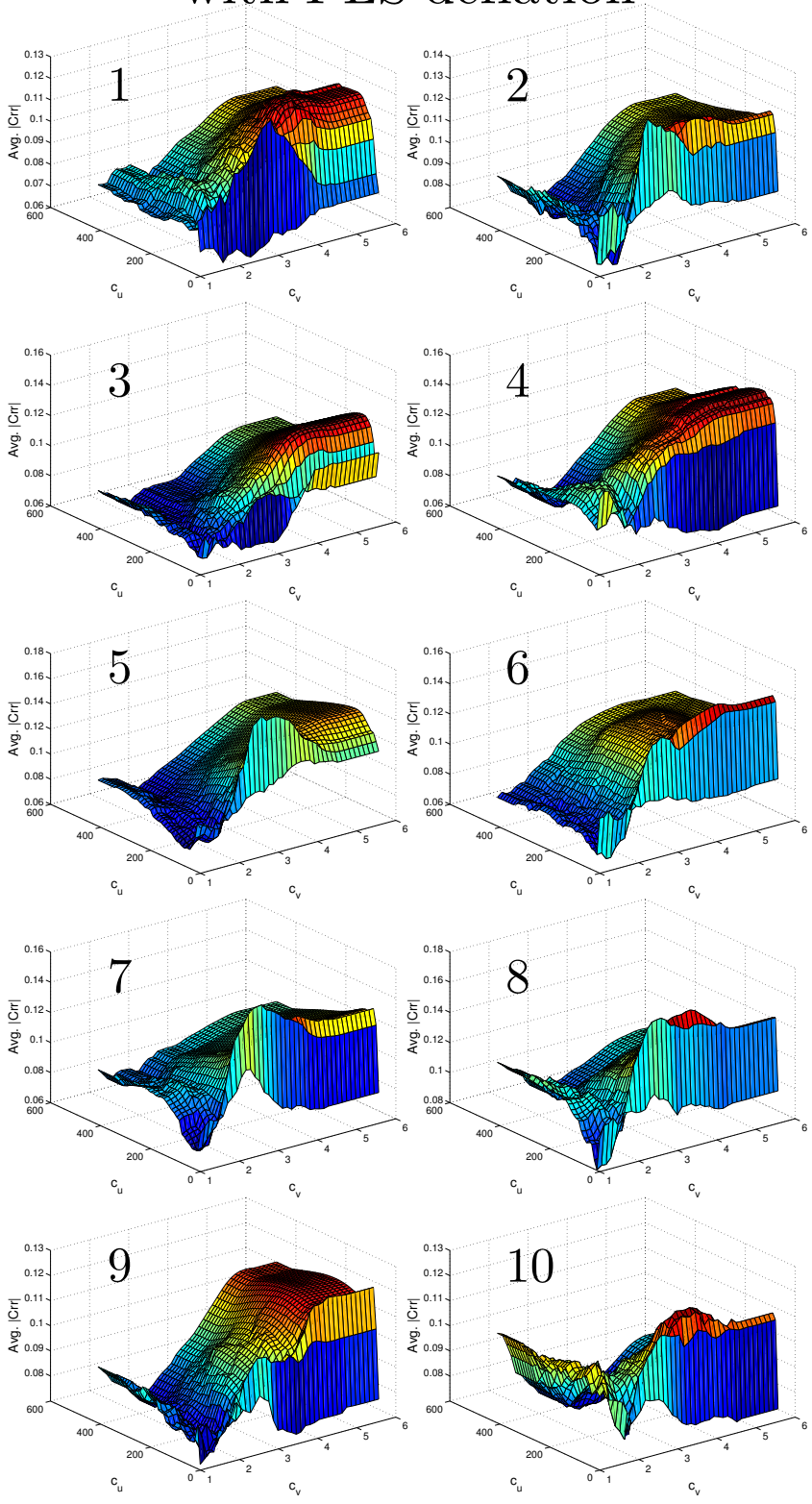


Figure B.3: Mean absolute correlation value computed for each split of the data (indicated by the numbers on the top left corners) for the second weight vector pair during the hyper-parameter optimisation step, using PLS Mode-A deflation..

B.3 Weight vectors or associative effects

B.3.1 PLS

The average of the clinical weight vectors is presented in Figure B.4, and the corresponding image weight vector can be seen in Figure B.5. As expected, these are not sparse, all the available clinical variables and image voxels are included in the model. As one can see, the weights are higher in the hippocampus and amygdala regions. However, since these are weights and not p-values (as the ones obtained in an mass-univariate statistical test), they cannot be thresholded.

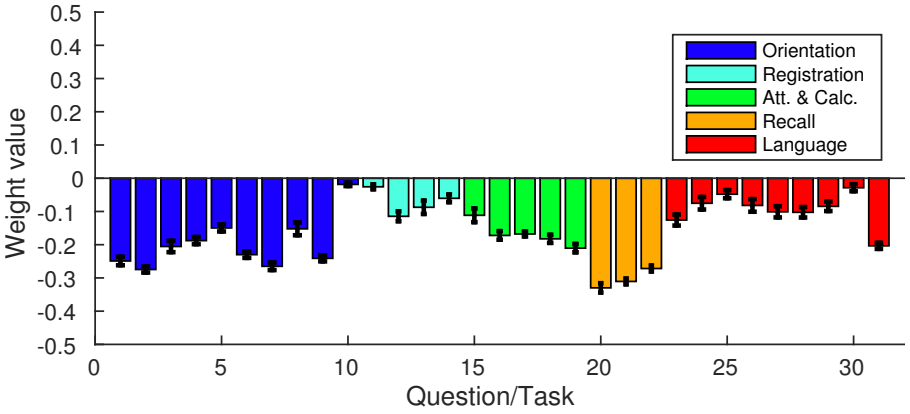


Figure B.4: Mean of clinical weight vector using PLS.

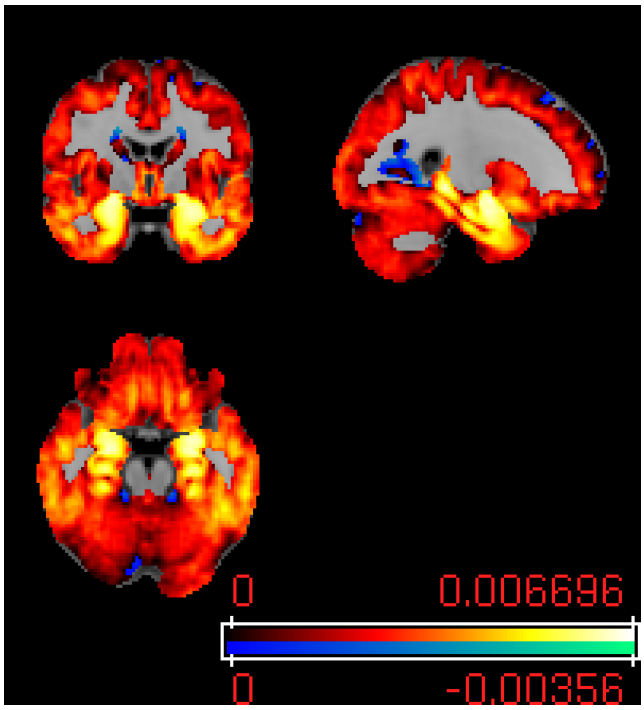


Figure B.5: Mean of image weight vectors using PLS.

B.3.2 SPLS with PLS deflation

This section shows the results obtained for the second weight vector pair when the common PLS deflation (Equation 3.13) was performed, instead of the projection deflation (Equation 4.1), which was used in this study.

Figure B.6(b) shows the mean clinical weight vector obtained using a PLS deflation. The weight vectors are less reliable than the ones obtained with SPLS using projection deflation (Figure B.6(a)), which can be observed by the larger standard deviation bars. Also, the weight vectors in Figure B.6(b) had to be forced to have a consistent direction before averaging, i.e. the direction of the weight vectors flipped from one split to another, which was not observed when using projection deflation (the direction was consistent). If this was not done, the standard deviation bars for the PLS deflation would be even larger. Thus, it is interesting to see that the non-statistical significance of the weight vectors is consistent with their lack of reliability.

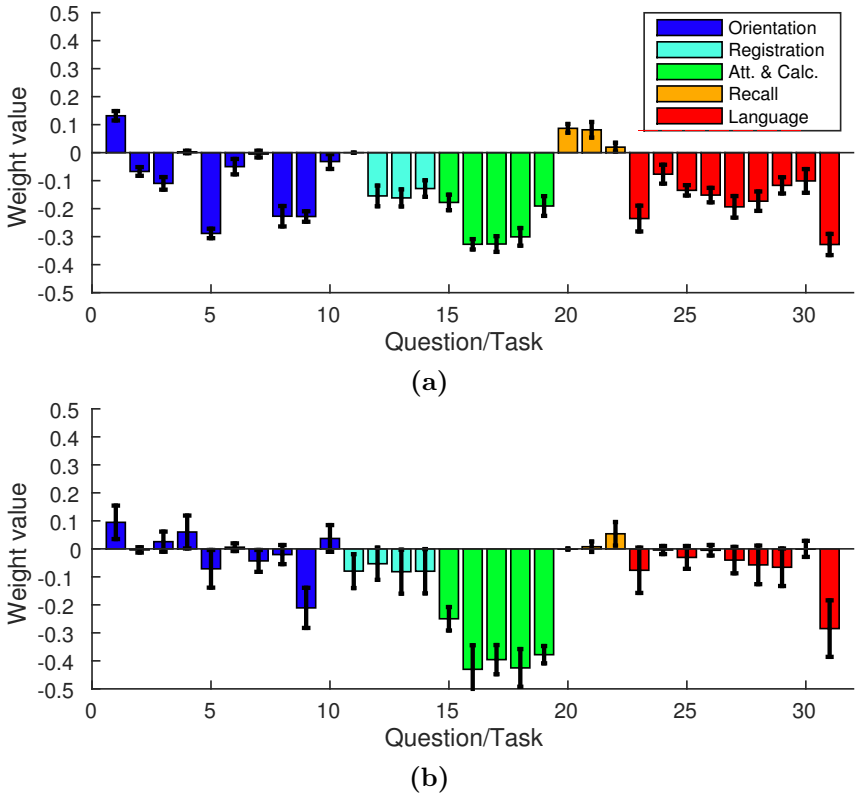
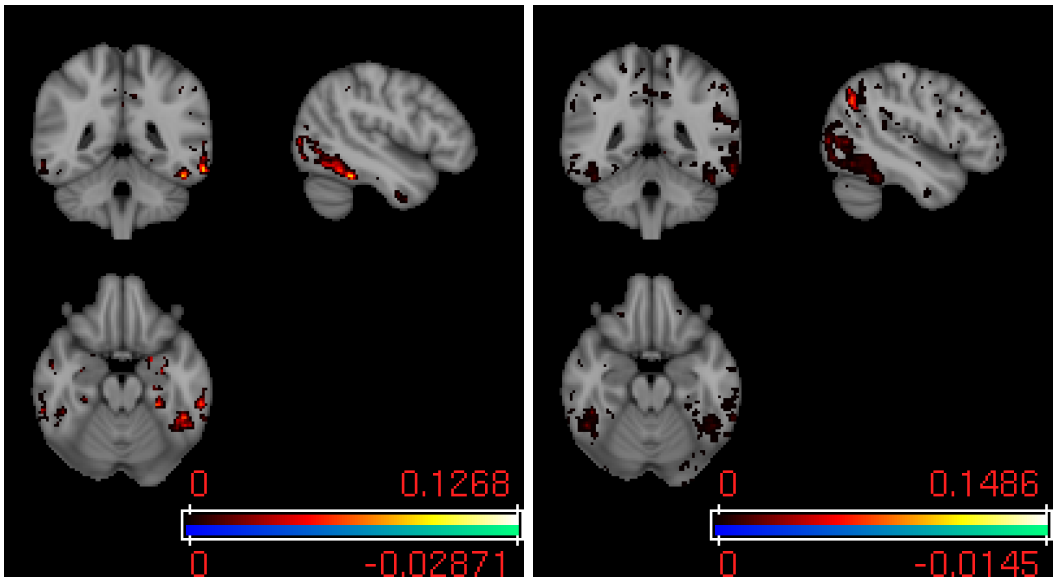


Figure B.6: **(a)** Mean of second clinical weight vectors using SPLS with projection deflation; **(b)** Mean of second clinical weight vectors using SPLS with PLS deflation. The weights vectors in (b) were forced to have a consistent direction before averaging.



(a) **(b)**

Figure B.7: **(a)** Mean of second image weight vectors using SPLS with projection deflation; **(b)** Mean of second image weight vectors using SPLS with PLS deflation.

The mean second image weight vector computed using the projection deflation is shown in Figure B.7(a), this appears to be sparser than the one obtained when using SPLS with a PLS deflation (Figure B.7(b)).

B.4 Atlas regions for each SPLS image weight vector

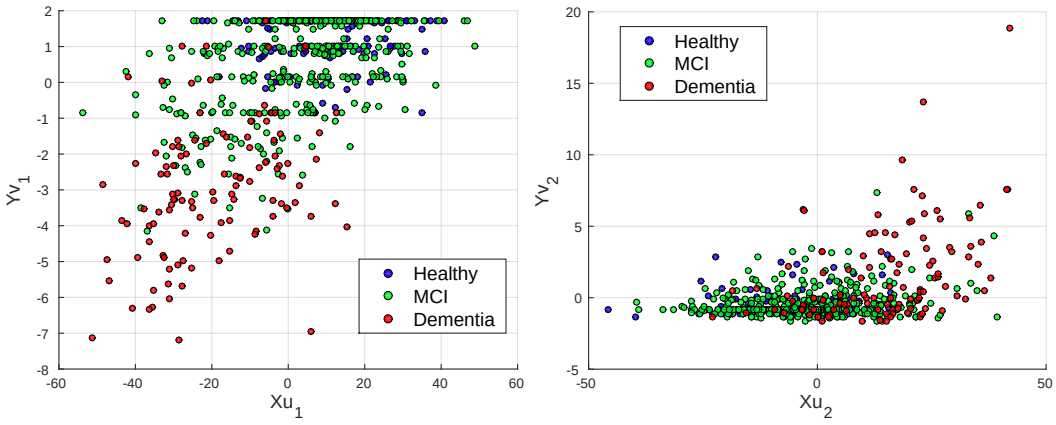
Table B.2: Atlas regions for the first image weight map. Only regions with selected voxels are shown.

Atlas Region	**# voxels found**
Amygdala_L	98
Amygdala_R	90
Hippocampus_R	175
Hippocampus_L	152
ParaHippocampal_R	92
ParaHippocampal_L	44
Lingual_L	9
Precuneus_L	2
Precuneus_R	1
Temporal_Pole_Sup_L	1
Fusiform_R	2

Table B.3: Atlas regions for the second image weight map. Only regions with selected voxels are shown.

Atlas Region	# voxels found
Amygdala_L	36
Temporal_Inf_L	292
Hippocampus_L	88
Amygdala_R	11
ParaHippocampal_L	53
Fusiform_L	78
Temporal_Inf_R	64
Hippocampus_R	22
Occipital_Inf_L	12
Temporal_Mid_L	76
Temporal_Mid_R	36
Heschl_R	1
Precuneus_L	17
Angular_R	4
Occipital_Mid_L	14
Temporal_Pole_Mid_L	1
Occipital_Mid_R	3
ParaHippocampal_R	4
Cingulum_Mid_L	5
Angular_L	2
Temporal_Pole_Sup_R	1
Insula_R	2
Precentral_R	2
Insula_L	4
Parietal_Inf_L	1
Lingual_L	2
Parietal_Inf_R	1
Caudate_L	1
Thalamus_L	1

B.5 Projections



(a) Projection of the image data onto the first weight vector pair $\{\boldsymbol{u}_1, \boldsymbol{v}_1\}$.

(b) Projection of the image data onto the second weight vector pair $\{\boldsymbol{u}_2, \boldsymbol{v}_2\}$.

Figure B.8: Projection of the data onto the SPLS weight vector pairs.

B.6 Number of SPLS computations

In order to statistically evaluate one weight vector pair, the number of times SPLS has to be performed in a nested cross-validation is given by:

$$N_{\text{SPLS}} = (k_o(n_p k_i) + k_o) + B(k_o(n_p k_i) + k_o)$$

where k_o is the number of outer folds, k_i is the number of inner folds, n_p is the number of hyper-parameter combinations, and B is the number of permutations.

The proposed framework requires a number of SPLS computations given by:

$$N_{\text{SPLS}} = S(Kn_p + 1) + SB$$

where S is the number of hold-out splits, and K is the number of subsample splits.

Appendix C

Chapter 7

C.1 SCCA using ALS

C.1.1 p-values

Table C.1: Correlations on the 10 hold-out sets, with the corresponding p-values in parenthesis. Statistically significant p-values are shown in bold.

	$\{\boldsymbol{X}, \boldsymbol{Y}\}$			$\{\boldsymbol{X}', \boldsymbol{Y}'\}$		
	No update	No δ update	δ update	No update	No δ update	δ update
1	0.6898 (**0.0001**)	0.6899 (**0.0001**)	0.6901 (**0.0001**)	0.6862 (**0.0001**)	0.6853 (**0.0001**)	0.6853 (**0.0001**)
2	0.8000 (**0.0001**)	0.8001 (**0.0001**)	0.8005 (**0.0001**)	0.7592 (**0.0001**)	0.7598 (**0.0001**)	0.7598 (**0.0001**)
3	0.8409 (**0.0001**)	0.8409 (**0.0001**)	0.8409 (**0.0001**)	0.8244 (**0.0001**)	0.8283 (**0.0001**)	0.8252 (**0.0001**)
4	0.6880 (**0.0001**)	0.6879 (**0.0001**)	0.6866 (**0.0001**)	0.6371 (**0.0001**)	0.6376 (**0.0001**)	0.6376 (**0.0001**)
5	0.8009 (**0.0001**)	0.8009 (**0.0001**)	0.7970 (**0.0001**)	0.7959 (**0.0001**)	0.7959 (**0.0001**)	0.7963 (**0.0001**)
6	0.7436 (**0.0001**)	0.7435 (**0.0001**)	0.7434 (**0.0001**)	0.6269 (**0.0001**)	0.6277 (**0.0001**)	0.6264 (**0.0001**)
7	0.7867 (**0.0001**)	0.7868 (**0.0001**)	0.7868 (**0.0001**)	0.7395 (**0.0001**)	0.7392 (**0.0001**)	0.7392 (**0.0001**)
8	0.7886 (**0.0001**)	0.7885 (**0.0001**)	0.7885 (**0.0001**)	0.6811 (**0.0001**)	0.6790 (**0.0001**)	0.6786 (**0.0001**)
9	0.7780 (**0.0001**)	0.7780 (**0.0001**)	0.7780 (**0.0001**)	0.7035 (**0.0001**)	0.7032 (**0.0001**)	0.7032 (**0.0001**)
10	0.6577 (**0.0001**)	0.6576 (**0.0001**)	0.6575 (**0.0001**)	0.6175 (**0.0001**)	0.6193 (**0.0001**)	0.6187 (**0.0001**)
Rej. H_{omni}	Yes	Yes	Yes	Yes	Yes	Yes

C.1.2 Distance to constraint

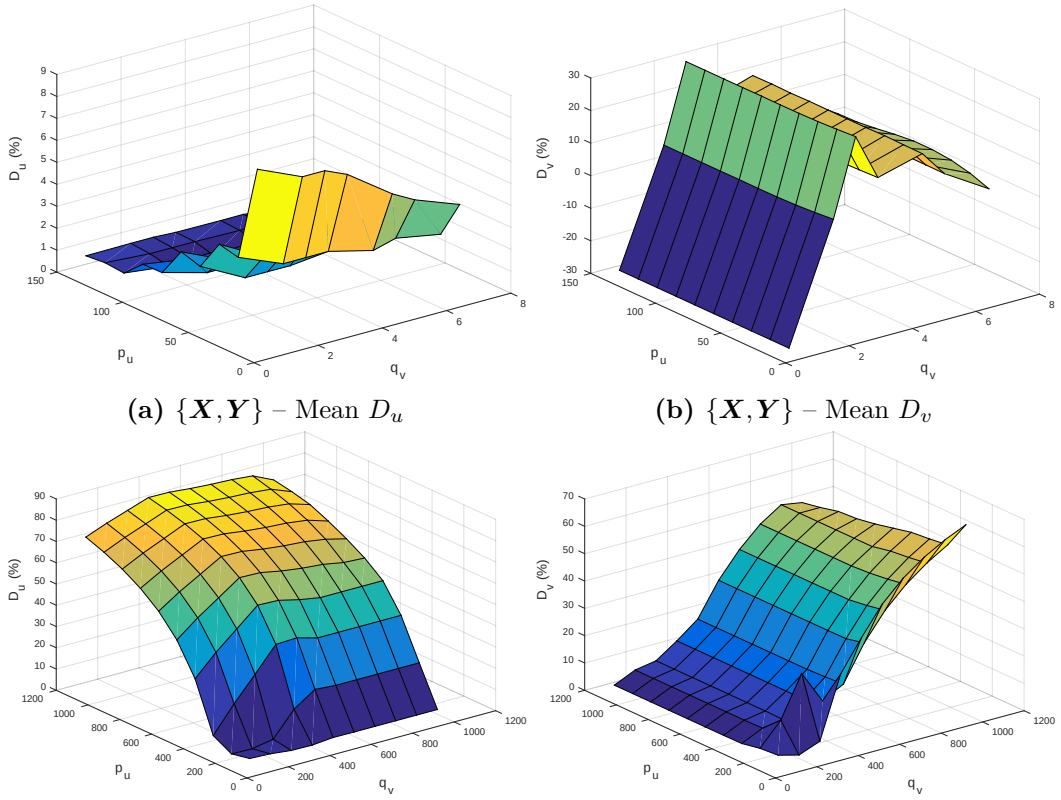
In order to quantify how far were the number of features of the final solutions (I_u, I_v) from the corresponding constraints (p_u, q_v), the "relative distance to the constraint" expressed as a percentage was computed in the following way:

$$D_u = \frac{I_u - p_u}{\max\{I_u, p_u\}} \times 100 \quad \text{and} \quad D_v = \frac{I_v - q_v}{\max\{I_v, q_v\}} \times 100$$

Please note that positive values of D indicate that the final solution was sparser than the one specified by the constraint, and negative values of D indicate that the algorithm was not able to converge to a point where the constraints expressed in Equation 7.2 were obeyed, i.e. the number of features exceeded the constraint.

The results are expressed in Figure C.1. Although not entirely consistent, there seems to be a general tendency for the relative distance to the constraint to increase,

the further away the hyper-parameter combination is from the optimal.



(a) $\{\boldsymbol{X}, \boldsymbol{Y}\}$ – Mean D_u

(b) $\{\boldsymbol{X}, \boldsymbol{Y}\}$ – Mean D_v

(c) $\{\boldsymbol{X}', \boldsymbol{Y}'\}$ – Mean D_u

(d) $\{\boldsymbol{X}', \boldsymbol{Y}'\}$ – Mean D_v

Figure C.1: Relative distances to the constraints (D_u and D_v), for both datasets (with and without noise).

C.1.3 ROI variables

Table C.2: Complete list of ROI volumes used as features in $\boldsymbol{X}$.

$\boldsymbol{X}$	$\boldsymbol{X}$ (cont.)
Right ACgG anterior cingulate gyrus	Right PP planum polare
Left ACgG anterior cingulate gyrus	Left PP planum polare
Right AIns anterior insula	Right PrG precentral gyrus
Left AIns anterior insula	Left PrG precentral gyrus
Right AOrG anterior orbital gyrus	Right PT planum temporale
Left AOrG anterior orbital gyrus	Left PT planum temporale
Right AnG angular gyrus	Right SCA subcallosal area
Left AnG angular gyrus	Left SCA subcallosal area
Right Calc calcarine cortex	Right SFG superior frontal gyrus
Left Calc calcarine cortex	Left SFG superior frontal gyrus
4th Ventricle	Right SMC supplementary motor cortex
Right CO central operculum	Left SMC supplementary motor cortex
Left CO central operculum	Right SMG supramarginal gyrus
Right Cun cuneus	Left SMG supramarginal gyrus
Left Cun cuneus	Right SOG superior occipital gyrus
Right Ent entorhinal area	Left SOG superior occipital gyrus
Left Ent entorhinal area	Right SPL superior parietal lobule
Right FO frontal operculum	Left SPL superior parietal lobule
Left FO frontal operculum	Right STG superior temporal gyrus
Right FRP frontal pole	Left STG superior temporal gyrus
Left FRP frontal pole	Right TMP temporal pole
Right FuG fusiform gyrus	Left TMP temporal pole
Left FuG fusiform gyrus	Right TrIFG triangular part of the inferior frontal gyrus
Right GRe gyrus rectus	Left TrIFG triangular part of the inferior frontal gyrus
Left GRe gyrus rectus	Right TTG transverse temporal gyrus
Right IOG inferior occipital gyrus	Left TTG transverse temporal gyrus
Left IOG inferior occipital gyrus	Right Accumbens Area
Right ITG inferior temporal gyrus	Left Accumbens Area
Left ITG inferior temporal gyrus	Right Amygdala
Right LiG lingual gyrus	Left Amygdala
Left LiG lingual gyrus	Brain Stem
Right LOrG lateral orbital gyrus	Right Caudate
Left LOrG lateral orbital gyrus	Left Caudate
Right MCgG middle cingulate gyrus	Right Cerebellum Exterior
Left MCgG middle cingulate gyrus	Left Cerebellum Exterior
Right MFC medial frontal cortex	3rd Ventricle
Left MFC medial frontal cortex	Right Cerebellum White Matter
Right MFG middle frontal gyrus	Left Cerebellum White Matter
Left MFG middle frontal gyrus	Right Hippocampus
Right MOG middle occipital gyrus	Left Hippocampus
Left MOG middle occipital gyrus	Right Inf Lat Vent
Right MOrG medial orbital gyrus	Left Inf Lat Vent
Left MOrG medial orbital gyrus	Right Lateral Ventricle
Right MPoG postcentral gyrus medial segment	Left Lateral Ventricle
Left MPoG postcentral gyrus medial segment	Right Pallidum
Right MPrG precentral gyrus medial segment	Left Pallidum
Left MPrG precentral gyrus medial segment	Right Putamen
Right MSFG superior frontal gyrus medial segment	Left Putamen
Left MSFG superior frontal gyrus medial segment	Right Thalamus Proper
Right MTG middle temporal gyrus	Left Thalamus Proper
Left MTG middle temporal gyrus	Right Ventral DC
Right OCP occipital pole	Left Ventral DC
Left OCP occipital pole	Cerebellar Vermal Lobules I-V
Right OFuG occipital fusiform gyrus	Cerebellar Vermal Lobules VI-VII
Left OFuG occipital fusiform gyrus	Cerebellar Vermal Lobules VIII-X
Right OpIFG opercular part of the inferior frontal gyrus	Left Basal Forebrain
Left OpIFG opercular part of the inferior frontal gyrus	Right Basal Forebrain
Right OrIFG orbital part of the inferior frontal gyrus	frontal lobe WM right
Left OrIFG orbital part of the inferior frontal gyrus	frontal lobe WM left
Right PCgG posterior cingulate gyrus	occipital lobe WM right
Left PCgG posterior cingulate gyrus	occipital lobe WM left
Right PCu precuneus	parietal lobe WM right
Left PCu precuneus	parietal lobe WM left
Right PHG parahippocampal gyrus	temporal lobe WM right
Left PHG parahippocampal gyrus	temporal lobe WM left
Right PIns posterior insula	fornix right
Left PIns posterior insula	fornix left
Right PO parietal operculum	anterior limb of internal capsule right
Left PO parietal operculum	anterior limb of internal capsule left
Right PoG postcentral gyrus	posterior limb of internal capsule inc. cerebral peduncle right
Left PoG postcentral gyrus	posterior limb of internal capsule inc. cerebral peduncle left
Right POrG posterior orbital gyrus	corpus callosum
Left POrG posterior orbital gyrus	

C.2 SCCA vs. SPLS

C.2.1 p-values

Table C.3: Correlations on the 10 hold-out sets for the non-sparse methods, with the corresponding p-values in parenthesis. Statistically significant p-values are shown in bold.

	CCA	PLS
1	0.6634 (**0.0001**)	0.4407 (0.0120)
2	0.7443 (**0.0001**)	0.6808 (**0.0034**)
3	0.8180 (**0.0001**)	0.5542 (0.0126)
4	0.6298 (**0.0001**)	0.3172 (0.0343)
5	0.7358 (**0.0001**)	0.5238 (0.0096)
6	0.6742 (**0.0001**)	0.3656 (0.0170)
7	0.7748 (**0.0001**)	0.4332 (0.0102)
8	0.7391 (**0.0001**)	0.5817 (**0.0014**)
9	0.7559 (**0.0001**)	0.3266 (0.0322)
10	0.6158 (**0.0001**)	0.4842 (0.0070)
Rej. H_{omni}	Yes	Yes

Table C.4: Correlations on the 10 hold-out sets for the sparse methods, with the corresponding p-values in parenthesis. Statistically significant p-values are shown in bold.

	SCCA-ALS-L1	SCCA-ALS-EN	SPLS-ALS-L1	SPLS-ALS-EN	SPLS-PM
1	0.6901 (**0.0001**)	0.6900 (**0.0001**)	0.6324 (**0.0013**)	0.6343 (**0.0012**)	0.5850 (**0.0005**)
2	0.8003 (**0.0001**)	0.7975 (**0.0001**)	0.7347 (**0.0004**)	0.7349 (**0.0005**)	0.8019 (**0.0001**)
3	0.8390 (**0.0001**)	0.8298 (**0.0001**)	0.6536 (**0.0036**)	0.6544 (**0.0031**)	0.7412 (**0.0001**)
4	0.6873 (**0.0001**)	0.6861 (**0.0001**)	0.6076 (**0.0014**)	0.6087 (**0.0013**)	0.5743 (**0.0004**)
5	0.8027 (**0.0001**)	0.8027 (**0.0001**)	0.6819 (**0.0008**)	0.6837 (**0.0006**)	0.6838 (**0.0001**)
6	0.7426 (**0.0001**)	0.7426 (**0.0001**)	0.5749 (**0.0036**)	0.5767 (**0.0035**)	0.6989 (**0.0001**)
7	0.7756 (**0.0001**)	0.7756 (**0.0001**)	0.5697 (**0.0041**)	0.5717 (**0.0041**)	0.6156 (**0.0002**)
8	0.7885 (**0.0001**)	0.7863 (**0.0001**)	0.5331 (0.0054)	0.5374 (**0.0042**)	0.6471 (**0.0002**)
9	0.7806 (**0.0001**)	0.7823 (**0.0001**)	0.5358 (0.0118)	0.5421 (0.0102)	0.4805 (0.0098)
10	0.6590 (**0.0001**)	0.6590 (**0.0001**)	0.5625 (**0.0040**)	0.5647 (**0.0035**)	0.6216 (**0.0001**)
Rej. H_{omni}	Yes	Yes	Yes	Yes	Yes

C.2.2 Hyper-parameter optimisation

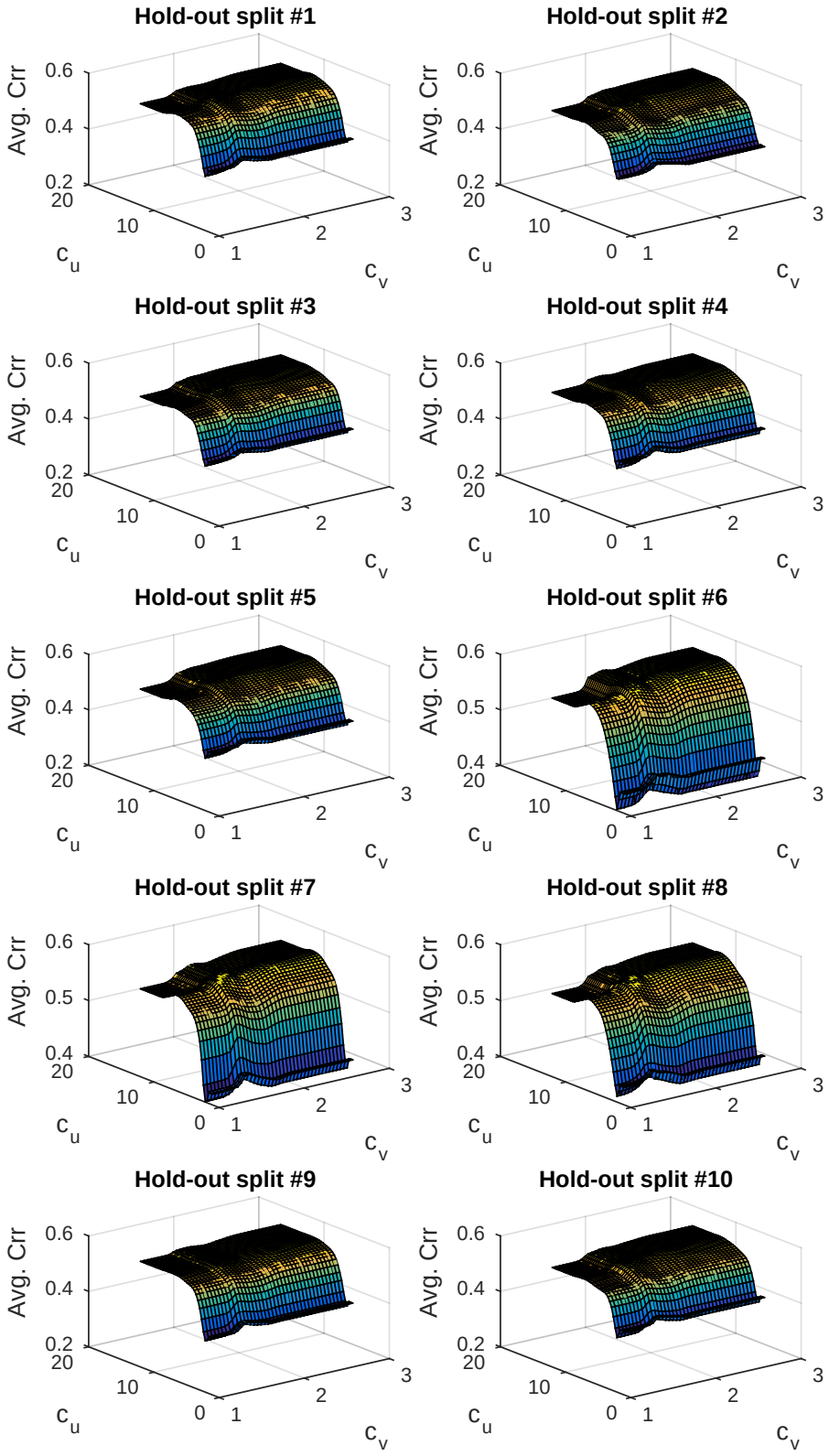


Figure C.2: Average test correlations for each hyper-parameter optimisation step, using SPLS-ALS-L1.

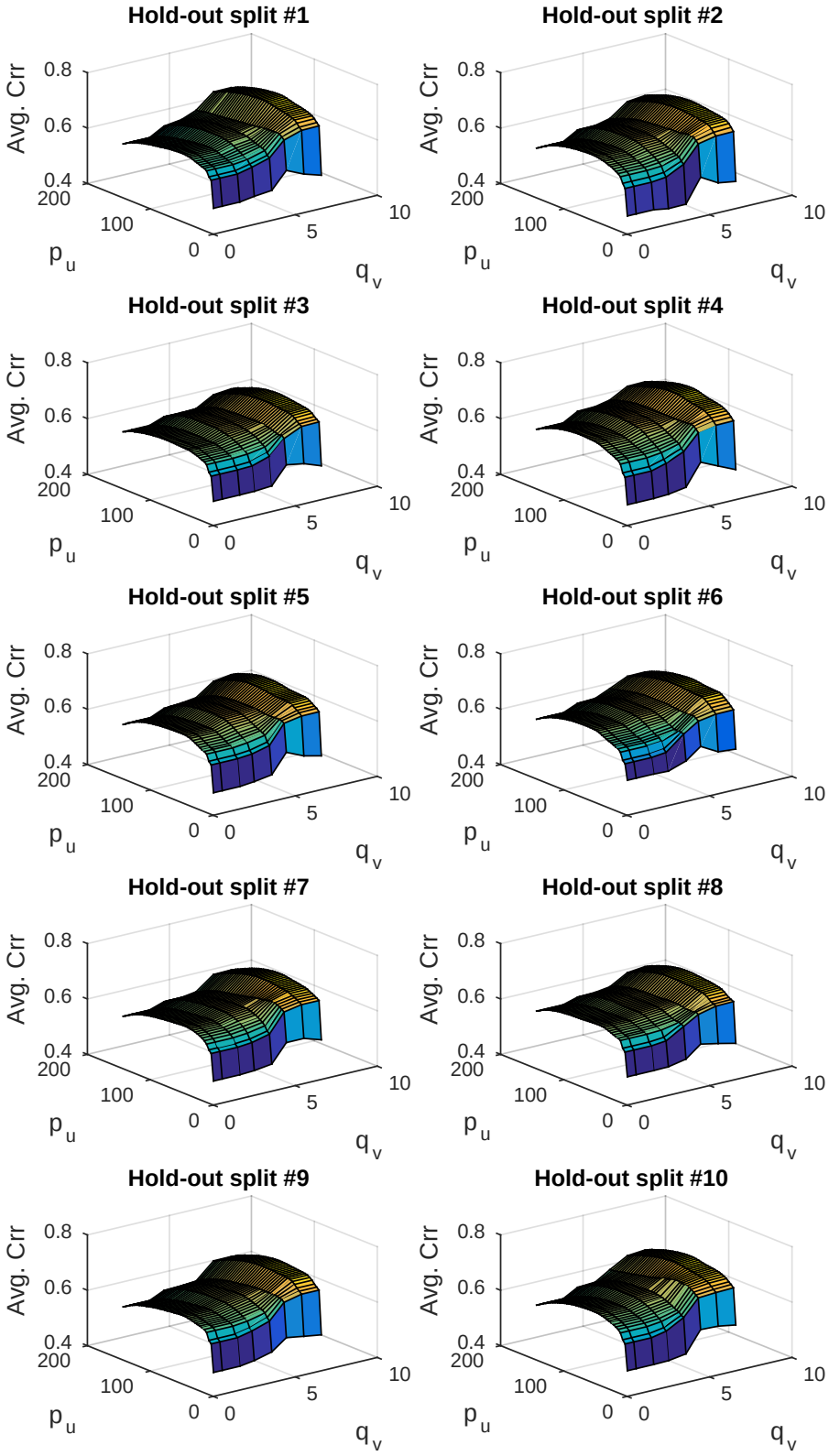


Figure C.3: Average test correlations for each hyper-parameter optimisation step, using SCCA-ALS-L1.

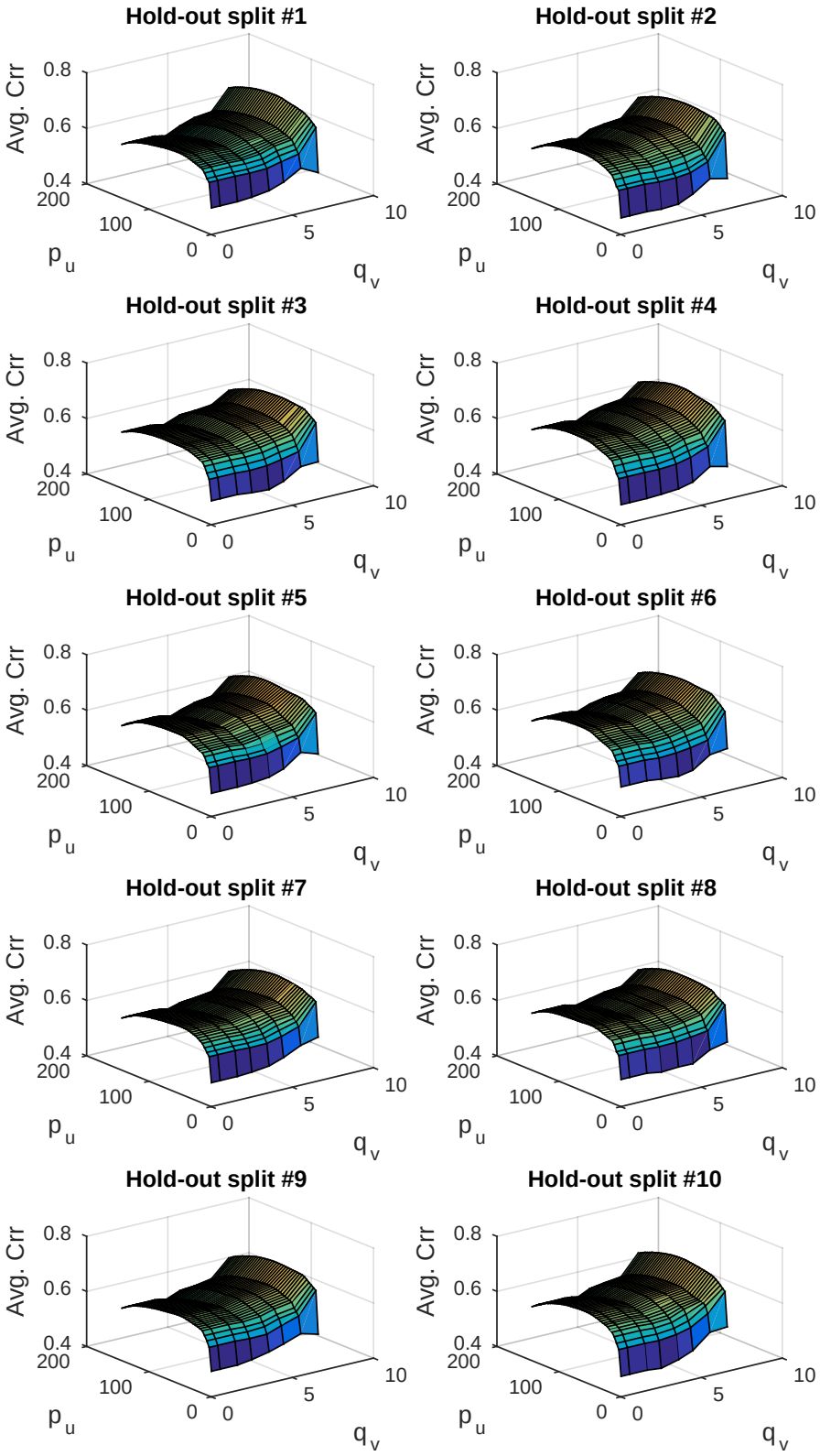


Figure C.4: Average test correlations for each hyper-parameter optimisation step, using SCCA-ALS-EN.

Appendix D

Chapter 8

D.1 p-values

Table D.1: Correlations on the 10 hold-out sets using the KCCA methods, with the corresponding p-values in parenthesis. Statistically significant p-values are shown in bold.

	KCCA-LK-LK	KCCA-PK-PK	KCCA-GK-GK
1	0.6962 (**0.0001**)	0.7103 (**0.0001**)	0.4959 (**0.0024**)
2	0.7918 (**0.0001**)	0.7919 (**0.0001**)	0.6908 (**0.0009**)
3	0.8230 (**0.0001**)	0.8268 (**0.0001**)	0.6298 (**0.0009**)
4	0.6889 (**0.0001**)	0.7004 (**0.0001**)	0.4329 (0.0058)
5	0.7920 (**0.0001**)	0.7938 (**0.0001**)	0.6347 (**0.0007**)
6	0.7052 (**0.0001**)	0.7337 (**0.0001**)	0.4576 (**0.0024**)
7	0.7636 (**0.0001**)	0.7781 (**0.0001**)	0.5313 (**0.0013**)
8	0.7985 (**0.0001**)	0.7734 (**0.0001**)	0.6850 (**0.0001**)
9	0.7624 (**0.0001**)	0.7685 (**0.0001**)	0.4527 (0.0071)
10	0.6535 (**0.0001**)	0.6621 (**0.0001**)	0.5719 (**0.0003**)
Rej. H_{omni}	Yes	Yes	Yes

Table D.2: Correlations on the 10 hold-out sets using the primal-dual SCCA methods, with the corresponding p-values in parenthesis. Statistically significant p-values are shown in bold.

	SCCA-L1-LK	SCCA-L1-PK	SCCA-L1-GK	SCCA-LK-L1	SCCA-PK-L1	SCCA-GK-L1
1	0.6895 (**0.0001**)	0.6618 (**0.0001**)	0.6704 (**0.0001**)	0.6947 (**0.0001**)	0.7136 (**0.0001**)	0.4933 (**0.0012**)
2	0.8011 (**0.0001**)	0.8083 (**0.0001**)	0.7560 (**0.0001**)	0.7906 (**0.0001**)	0.7817 (**0.0001**)	0.6877 (**0.0002**)
3	0.8337 (**0.0001**)	0.8335 (**0.0001**)	0.7796 (**0.0001**)	0.8294 (**0.0001**)	0.8373 (**0.0001**)	0.6310 (**0.0002**)
4	0.6903 (**0.0001**)	0.6811 (**0.0001**)	0.6498 (**0.0001**)	0.6862 (**0.0001**)	0.7234 (**0.0001**)	0.6327 (**0.0001**)
5	0.8010 (**0.0001**)	0.8030 (**0.0001**)	0.7334 (**0.0001**)	0.7948 (**0.0001**)	0.8026 (**0.0001**)	0.6564 (**0.0004**)
6	0.7417 (**0.0001**)	0.7391 (**0.0001**)	0.7086 (**0.0001**)	0.7220 (**0.0001**)	0.7339 (**0.0001**)	0.4901 (**0.0005**)
7	0.7743 (**0.0001**)	0.7697 (**0.0001**)	0.6850 (**0.0001**)	0.7723 (**0.0001**)	0.7886 (**0.0001**)	0.6074 (**0.0002**)
8	0.7888 (**0.0001**)	0.7707 (**0.0001**)	0.7825 (**0.0001**)	0.7983 (**0.0001**)	0.8006 (**0.0001**)	0.6492 (**0.0001**)
9	0.7797 (**0.0001**)	0.7675 (**0.0001**)	0.6952 (**0.0001**)	0.7635 (**0.0001**)	0.7797 (**0.0001**)	0.5387 (**0.0013**)
10	0.6580 (**0.0001**)	0.6645 (**0.0001**)	0.6599 (**0.0001**)	0.6569 (**0.0001**)	0.6608 (**0.0001**)	0.5478 (**0.0008**)
Rej. H_{omni}	Yes	Yes	Yes	Yes	Yes	Yes

D.2 Hyper-parameter optimisation

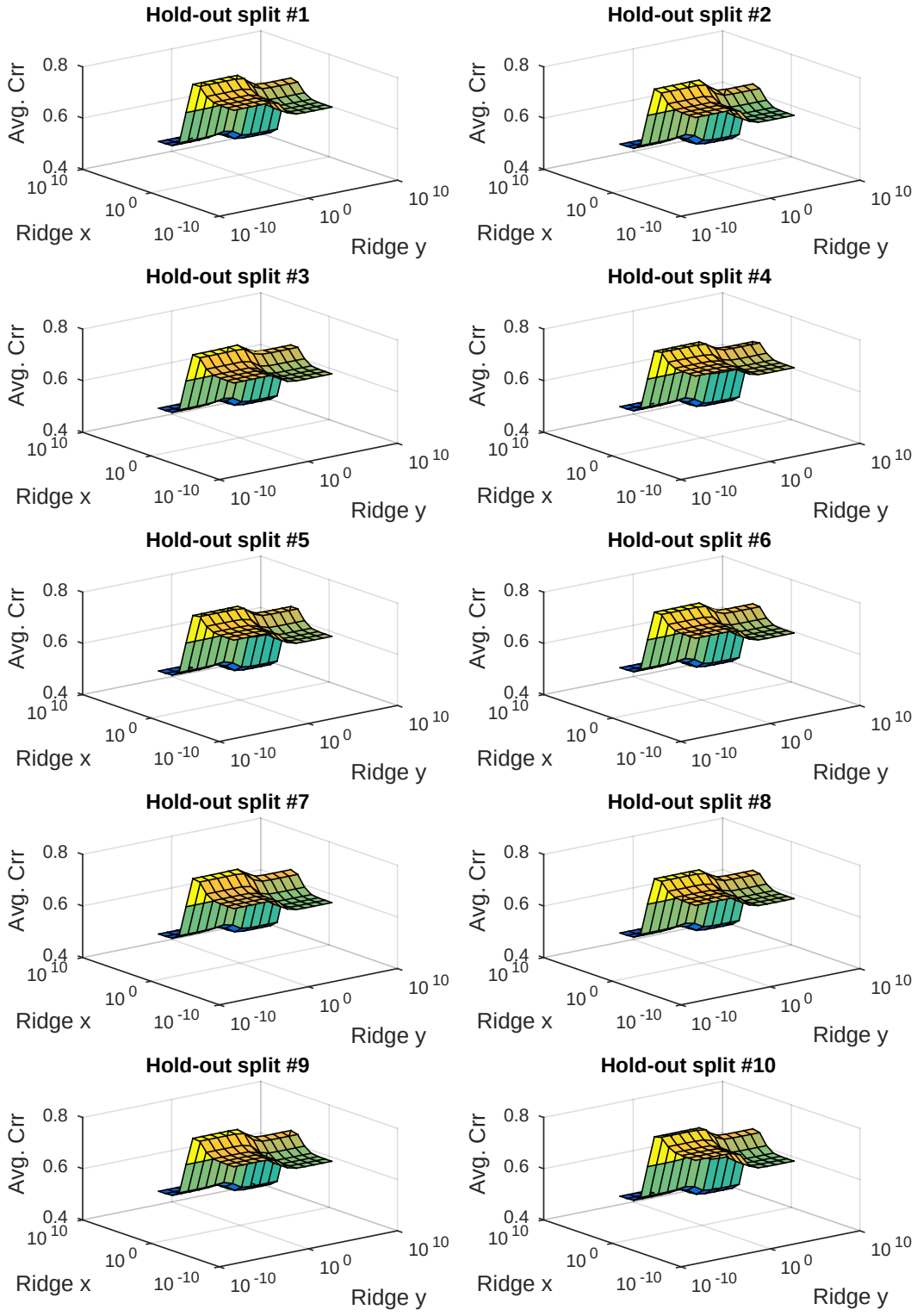


Figure D.1: Average test correlations for each hyper-parameter optimisation step, using SCCA-LK-LK.

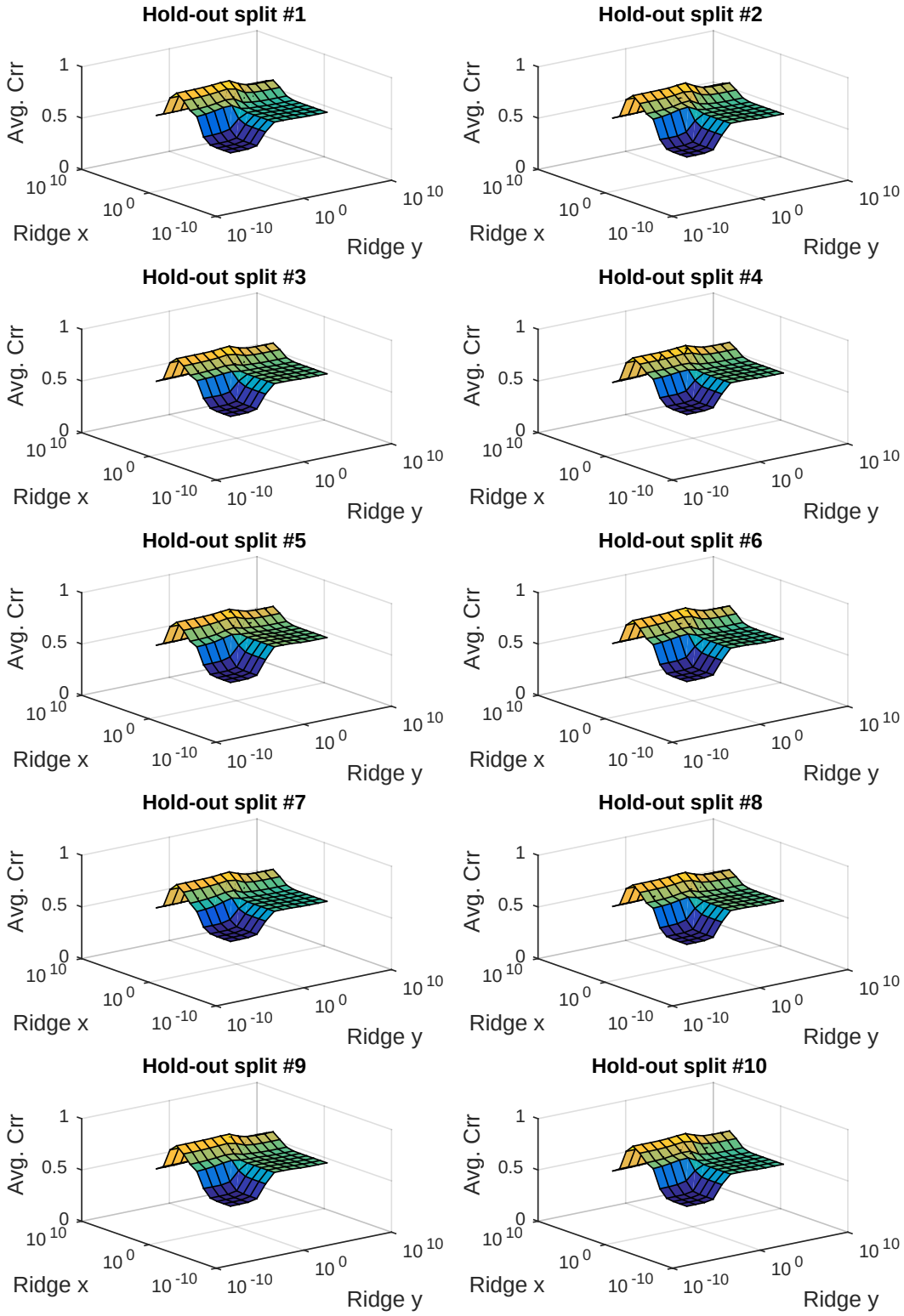


Figure D.2: Average test correlations for each hyper-parameter optimisation step, using SCCA-PK-PK.

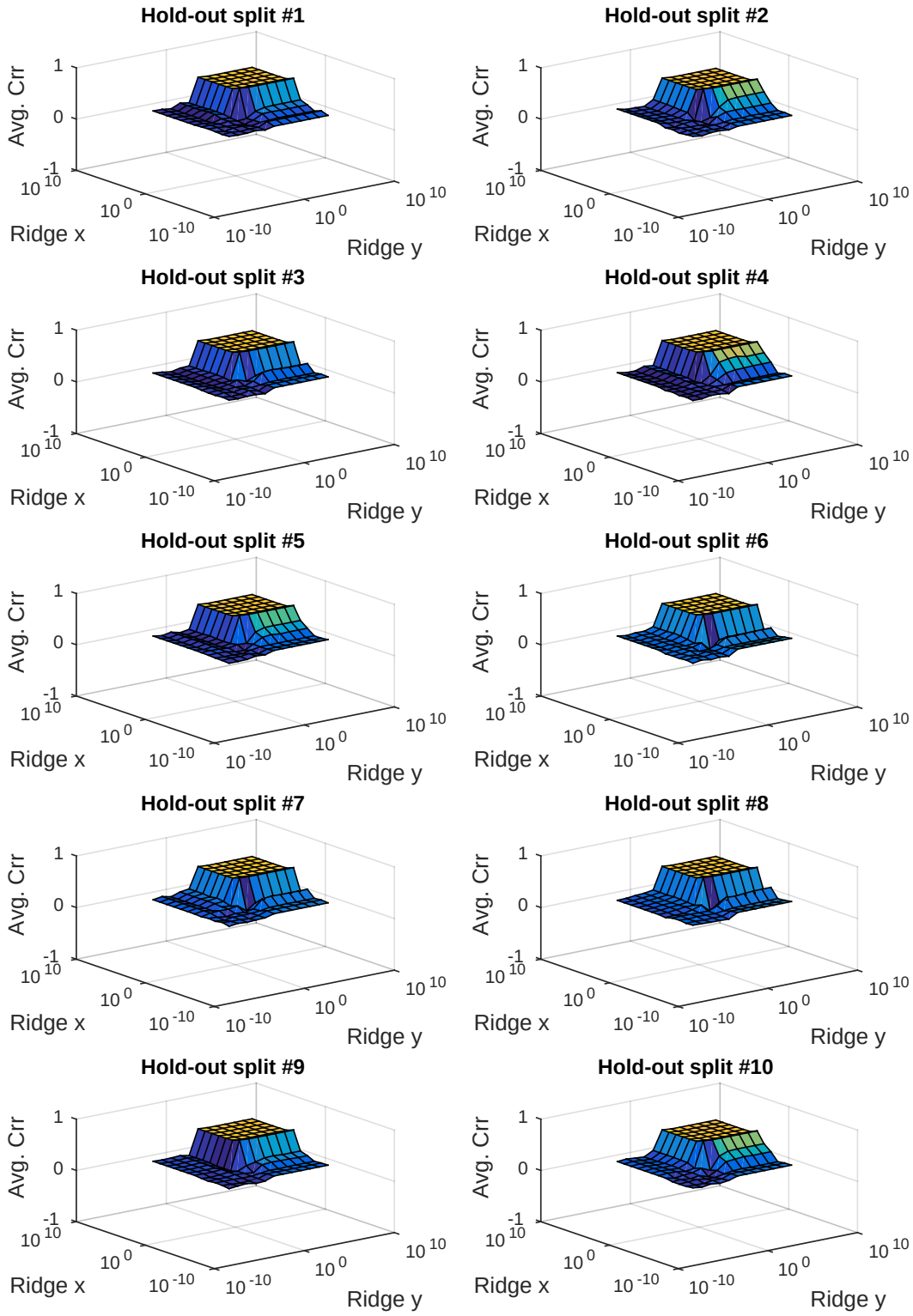


Figure D.3: Average test correlations for each hyper-parameter optimisation step, using SCCA-GK-GK.

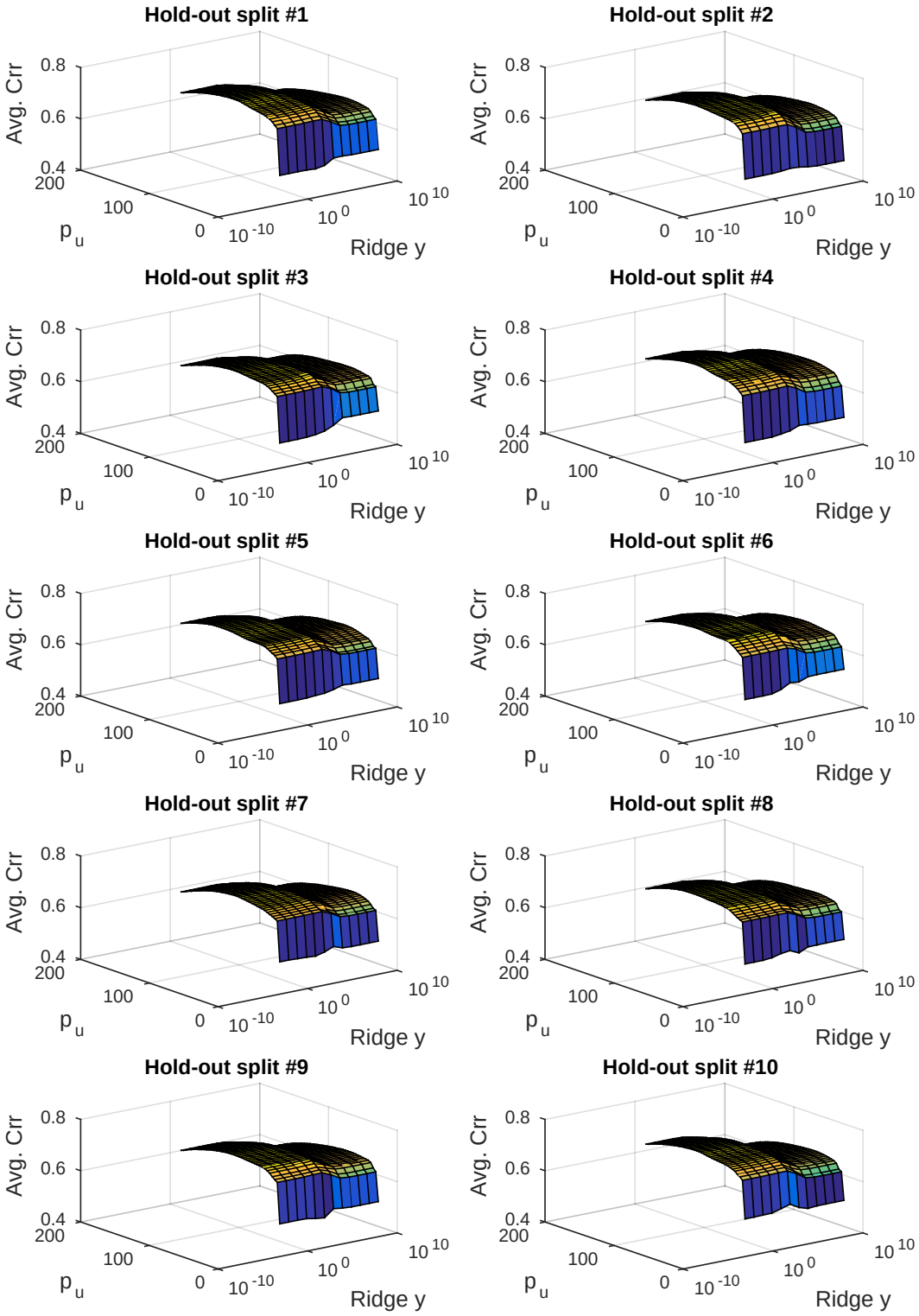


Figure D.4: Average test correlations for each hyper-parameter optimisation step, using SCCA-L1-LK.

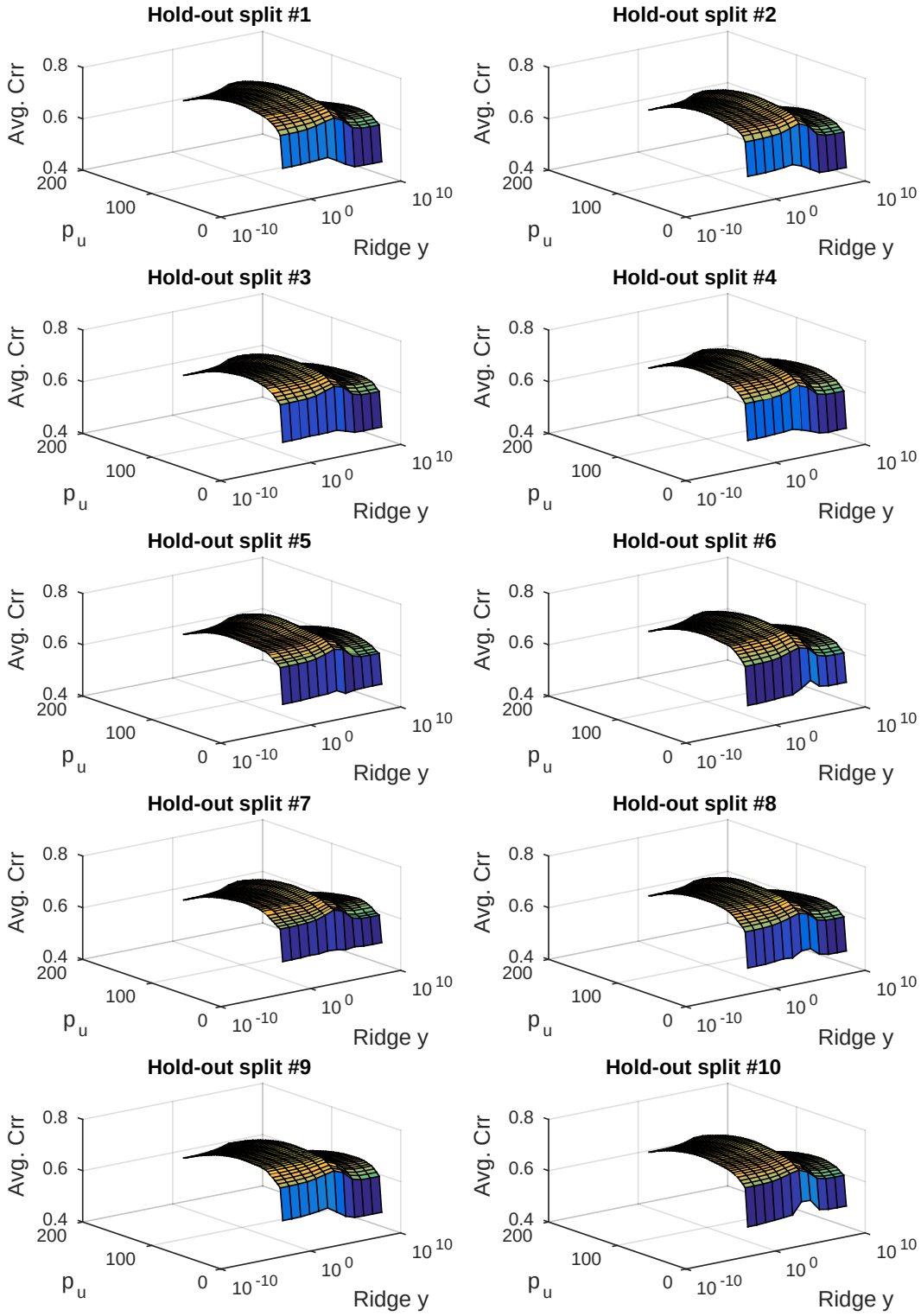


Figure D.5: Average test correlations for each hyper-parameter optimisation step, using SCCA-L1-PK.

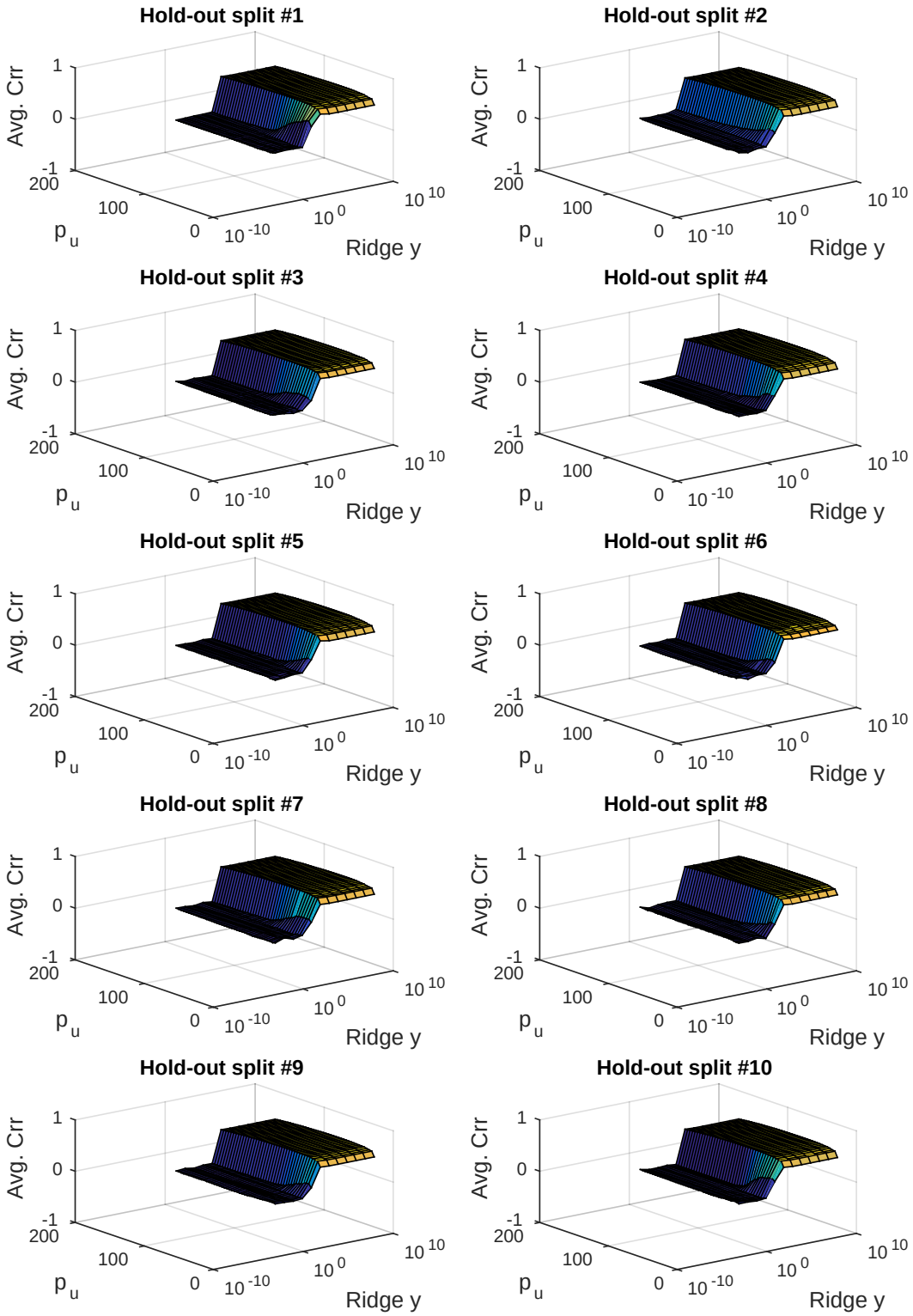


Figure D.6: Average test correlations for each hyper-parameter optimisation step, using SCCA-L1-GK.

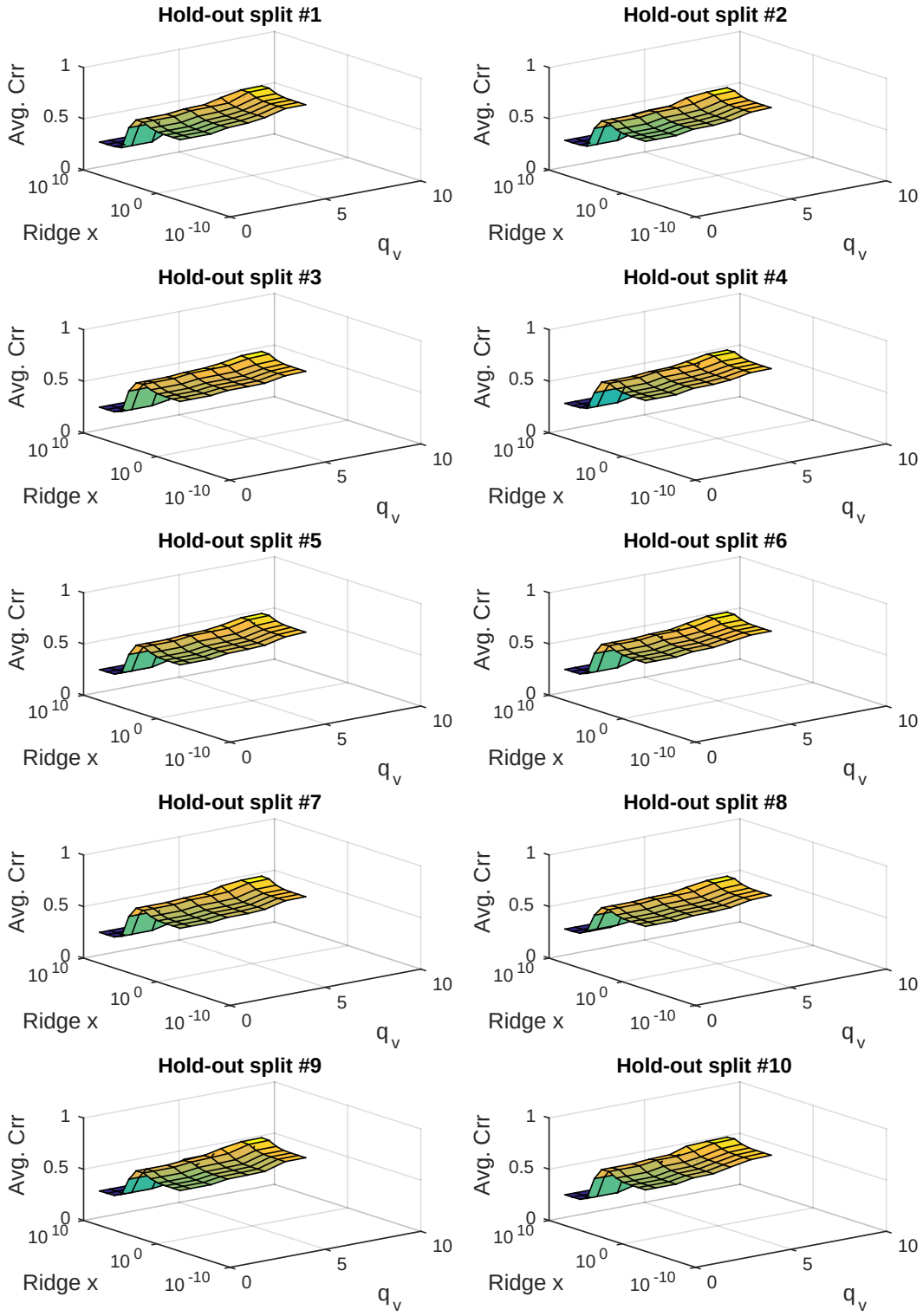


Figure D.7: Average test correlations for each hyper-parameter optimisation step, using SCCA-LK-L1.

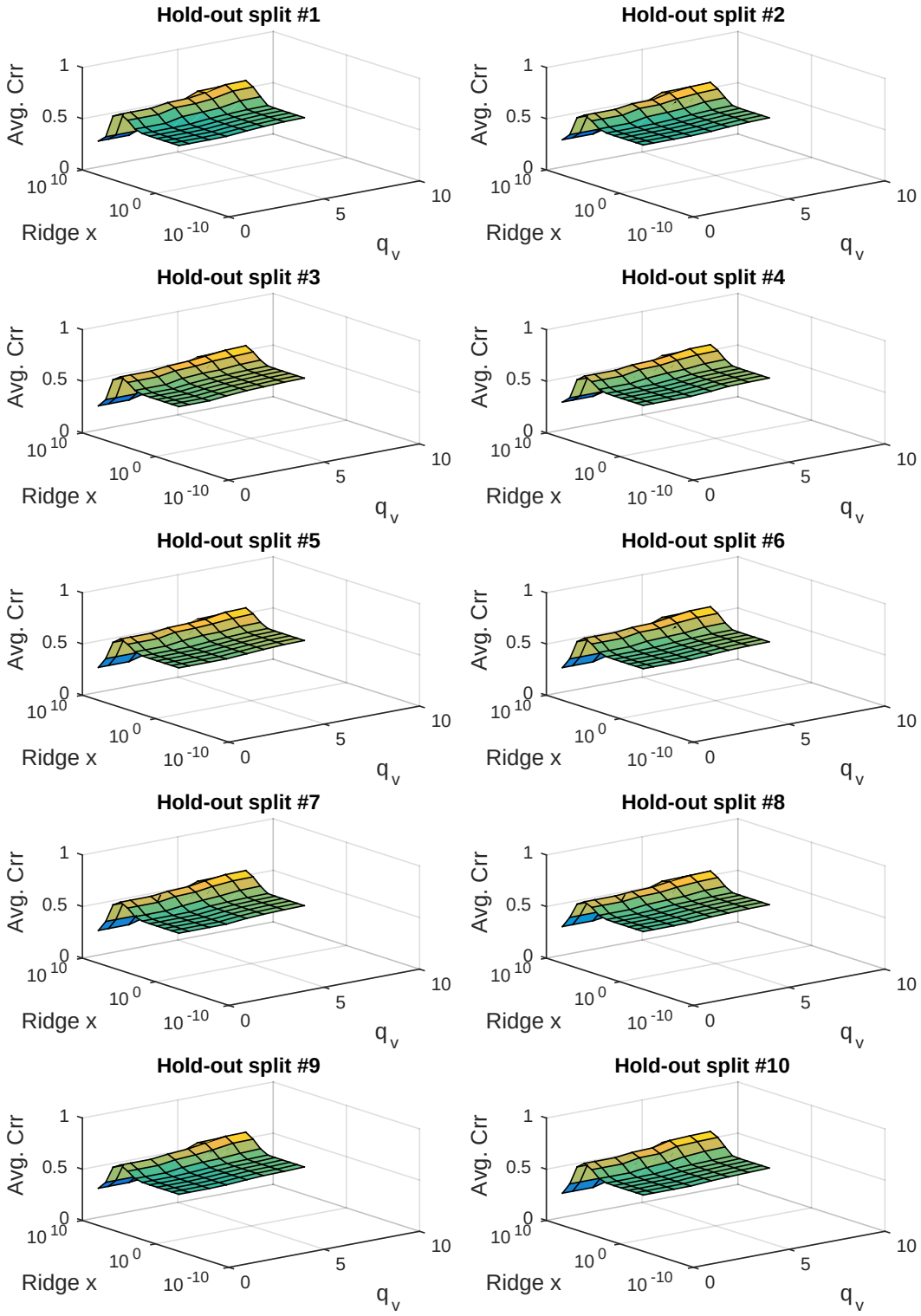


Figure D.8: Average test correlations for each hyper-parameter optimisation step, using SCCA-PK-L1.

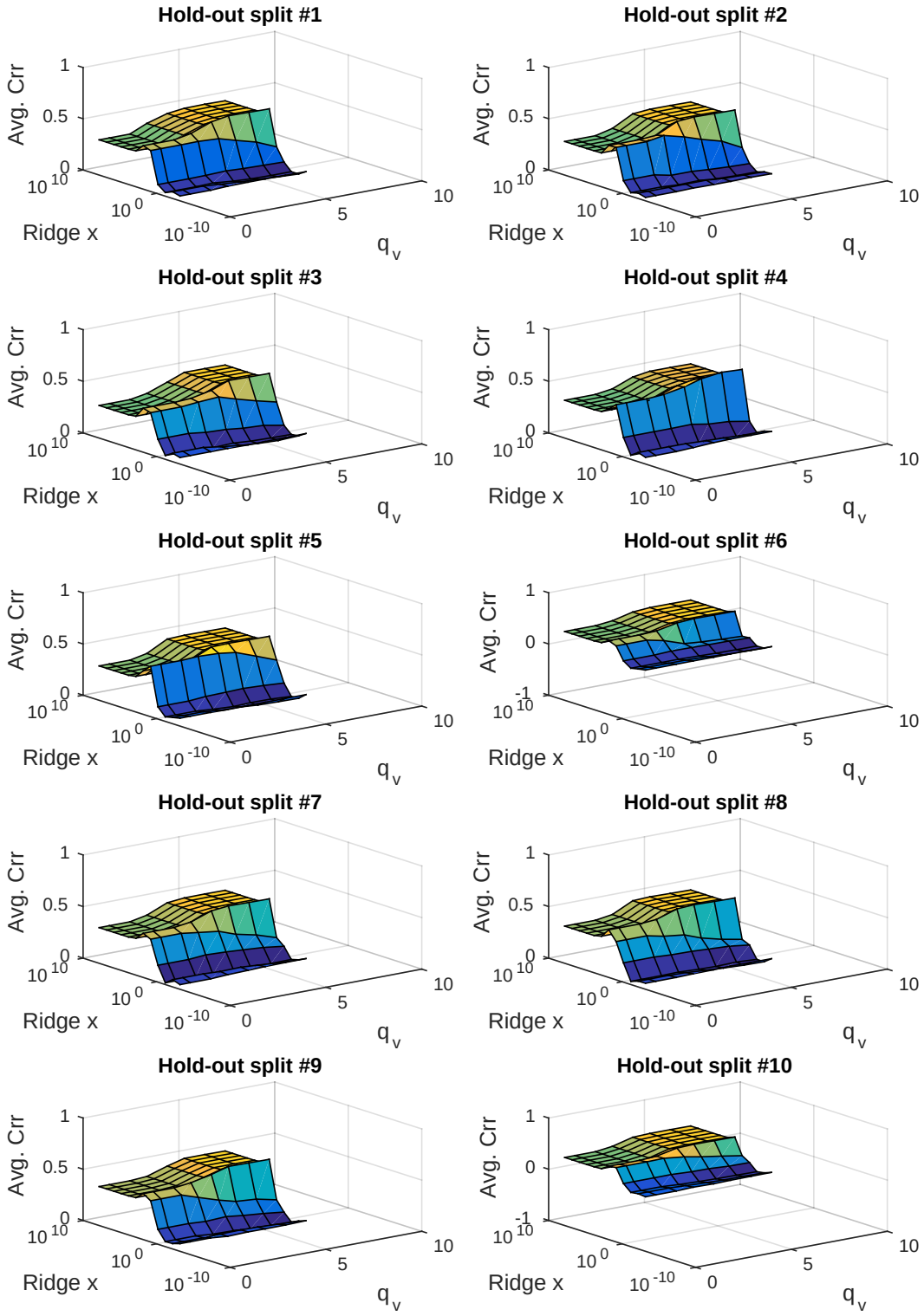
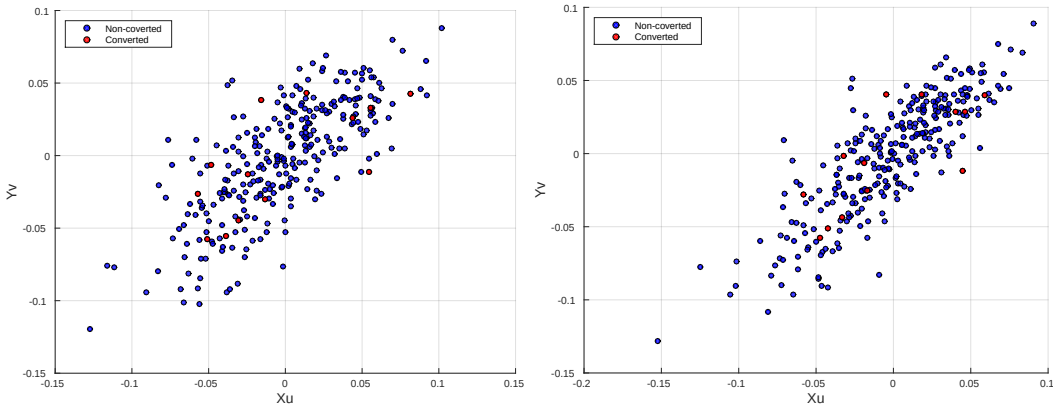


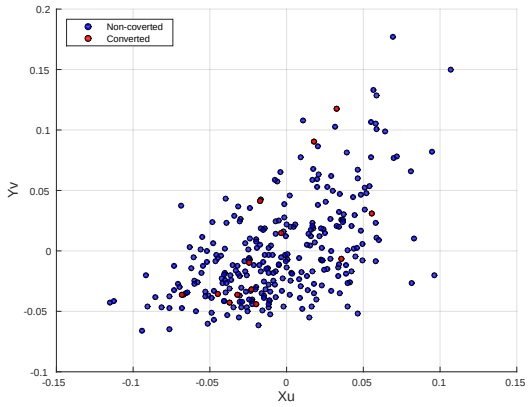
Figure D.9: Average test correlations for each hyper-parameter optimisation step, using SCCA-GK-L1.

D.3 Projections



(a) KCCA-LK-LK

(b) KCCA-PK-PK



(c) KCCA-GK-GK

Figure D.10: Subject conversion from MCI to AD after 6 months, using non-sparse methods.

Bibliography

NSPN - NeuroScience in Psychiatry Network. URL `http://www.nspn.org.uk/`.

ADNI general procedures manual, 2006.

A. Altmann, L. Tian, V. W. Henderson, and M. D. Greicius. Sex modifies the apoe-related risk of developing alzheimer disease. *Annals of neurology*, 75(4): 563–573, 2014.

J. Ashburner. A fast diffeomorphic image registration algorithm. *Neuroimage*, 38(1): 95–113, 2007.

J. Ashburner and K. J. Friston. Voxel-based morphometry—the methods. *Neuroimage*, 11(6):805–821, 2000.

J. Ashburner and K. J. Friston. Unified segmentation. *Neuroimage*, 26(3):839–851, 2005.

J. Ashburner, G. Barnes, C.-c. Chen, J. Daunizeau, R. Moran, R. Henson, V. Glauche, and C. Phillips. SPM12 Manual. *Functional Imaging Laboratory*, pages 475–1, 2013. ISSN 09621083. doi: 10.1111/j.1365-294X.2006.02813.x.

B. B. Avants, P. A. Cook, L. Ungar, J. C. Gee, and M. Grossman. Dementia induces correlated reductions in white matter integrity and cortical thickness: A multivariate neuroimaging study with sparse canonical correlation analysis. *NeuroImage*, 50(3):1004–1016, 2010.

B. B. Avants, D. J. Libon, K. Rascovsky, A. Boller, C. T. McMillan, L. Massimo, H. B. Coslett, A. Chatterjee, R. G. Gross, and M. Grossman. Sparse canonical correlation analysis relates network-level atrophy to multivariate cognitive measures in a neurodegenerative population. *NeuroImage*, 84(0):698–711, 2014.

F. R. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of machine learning research*, 3(Jul):1–48, 2002.

r. Björck and G. H. Golub. Numerical methods for computing angles between linear subspaces. *Mathematics of computation*, 27(123):579–594, 1973.

M. B. Blaschko, J. A. Shelton, A. Bartels, C. H. Lampert, and A. Gretton. Semi-supervised kernel canonical correlation analysis with application to human fmri. *Pattern Recognition Letters*, 32(11):1572–1583, 2011.

M. Borga, O. Friman, P. Lundberg, and H. Knutsson. A canonical correlation approach to exploratory data analysis in fmri. In *Proceedings of the ISMRM Annual Meeting, Honolulu, Hawaii*, 2002.

D. Chan, N. Fox, R. Scahill, W. Crum, J. Whitwell, G. Leschziner, A. Rossor, J. Stevens, L. Cipolotti, and M. Rossor. Patterns of temporal lobe atrophy in semantic dementia and Alzheimer's disease. *Annals of Neurology*, 49(2):433–442, 2001. ISSN 0364-5134.

E. C. Chi, G. I. Allen, H. Zhou, O. Kohannim, K. Lange, and P. M. Thompson. Imaging genetics via sparse canonical correlation analysis. In *2013 IEEE 10th International Symposium on Biomedical Imaging*, pages 740–743. IEEE, 2013.

C.-Y. C. Chu. *Pattern recognition and machine learning for magnetic resonance images with kernel methods*. PhD thesis, University College London, 2009.

H. Chun and S. Keleş. Sparse partial least squares regression for simultaneous dimension reduction and variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(1):3–25, 2010.

H. Chun, D. H. Ballard, J. Cho, and H. Zhao. Identification of association between disease and multiple markers via sparse partial least-squares regression. *Genetic epidemiology*, 35(6):479–486, 2011.

N. M. Correa, T. Eichele, T. Adalı, Y.-O. Li, and V. D. Calhoun. Multi-set canonical correlation analysis for the fusion of concurrent single trial erp and functional mri. *Neuroimage*, 50(4):1438–1445, 2010.

S. G. Costafreda, C. Chu, J. Ashburner, and C. H. Fu. Prognostic and diagnostic potential of the structural neuroanatomy of depression. *PloS one*, 4(7):e6353, 2009.

C. Davatzikos, Y. Fan, X. Wu, D. Shen, and S. M. Resnick. Detection of prodromal alzheimer's disease via pattern classification of magnetic resonance imaging. *Neurobiology of aging*, 29(4):514–523, 2008.

V. Della-Maggiore, A. B. Sekuler, C. L. Grady, P. J. Bennett, R. Sekuler, and A. R. McIntosh. Corticolimbic interactions associated with performance on a short-term memory task are modified by age. *J Neurosci*, 20(22):8410–8416, 2000.

L. Dong, Y. Zhang, R. Zhang, X. Zhang, D. Gong, P. A. Valdes-Sosa, P. Xu, C. Luo, and D. Yao. Characterizing nonlinear relationships in functional imaging data using eigenspace maximal information canonical correlation analysis (emicca). *NeuroImage*, 109:388–401, 2015.

M. Donini, J. M. Monteiro, M. Pontil, J. Shawe-Taylor, and J. Mourao-Miranda. A multimodal multiple kernel learning approach to Alzheimer's disease detection. In *Machine Learning for Signal Processing (MLSP), 2016 IEEE 26th International Workshop on*, pages 1–6. IEEE, 2016.

M. Donini, J. M. Monteiro, M. Pontil, T. Hahn, A. J. Fallgatter, J. Shawe-Taylor, and J. Mourão-Miranda. Combining heterogeneous data sources for prediction: re-weighting and selecting what is important. *NeuroImage*, (submitted).

L. Du, J. Yan, S. Kim, S. L. Risacher, H. Huang, M. Inlow, J. H. Moore, A. J. Saykin, L. Shen, et al. GN-SCCA: Graphnet based sparse canonical correlation analysis for brain imaging genetics. In *International Conference on Brain Informatics and Health*, pages 275–284. Springer, 2015.

C. Ecker, A. Marquand, J. Mourão-Miranda, P. Johnston, E. M. Daly, M. J. Brammer, S. Maltezos, C. M. Murphy, D. Robertson, S. C. Williams, et al. Describing the brain in autism in five dimensions—magnetic resonance imaging-assisted diagnosis of autism spectrum disorder using a multiparameter classification approach. *The Journal of Neuroscience*, 30(32):10612–10623, 2010.

B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, et al. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

M. F. Folstein, S. E. Folstein, and P. R. McHugh. "Mini-mental state". A practical method for grading the cognitive state of patients for the clinician. *Journal of psychiatric research*, 12:189–198, 1975. ISSN 0022-3956.

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

O. Friman, J. Cedefamn, P. Lundberg, M. Borga, and H. Knutsson. Detection of neural activity in functional mri using canonical correlation analysis. *Magnetic Resonance in Medicine*, 45(2):323–330, 2001.

K. J. Friston, A. P. Holmes, K. J. Worsley, J.-P. Poline, C. D. Frith, and R. S. Frackowiak. Statistical parametric maps in functional imaging: a general linear approach. *Human brain mapping*, 2(4):189–210, 1994.

X. Fu, K. Huang, M. Hong, N. D. Sidiropoulos, and A. M.-C. So. Scalable and flexible multiview max-var canonical correlation analysis. *arXiv preprint arXiv:1605.09459*, 2016.

C. J. Galton, K. Patterson, K. Graham, M. A. Lambon-Ralph, G. Williams, N. Antoun, B. J. Sahakian, and J. R. Hodges. Differing patterns of temporal atrophy in Alzheimer's disease and semantic dementia. *Neurology*, 57(2):216–225, 2001. ISSN 0028-3878, 1526-632X.

C. Giessing, G. R. Fink, F. Rösler, and C. M. Thiel. fMRI data predict individual differences of behavioral effects of nicotine: a partial least square analysis. *Journal of cognitive neuroscience*, 19(4):658–670, 2007.

G. H. Golub and H. Zha. The canonical correlations of matrix pairs and their numerical computation. 1992. ISSN 00243795. doi: 10.1016/0024-3795(94)90463-4.

C. Grellmann, S. Bitzer, J. Neumann, L. T. Westlye, O. A. Andreassen, A. Villringer, and A. Horstmann. Comparison of variants of canonical correlation analysis and partial least squares for combined analysis of mri and genetic data. *NeuroImage*, 107:289–310, 2015.

D. R. Hardoon and J. Shawe-Taylor. Sparse canonical correlation analysis. *Machine Learning*, 83(3):331–353, 2011.

D. R. Hardoon, J. Mourão-Miranda, M. Brammer, and J. Shawe-Taylor. Unsupervised analysis of fMRI data using kernel canonical correlation. *NeuroImage*, 37(4):1250–1259, 2007.

D. R. Hardoon, U. Ettinger, J. Mourão-Miranda, E. Antonova, D. Collier, V. Kumari, S. C. Williams, and M. Brammer. Correlation-based multivariate analysis of genetic influence on brain volume. *Neuroscience Letters*, 450(3):281–286, 2009.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*, volume 1. 2009. ISBN 9780387848570. doi: 10.1007/b94608. URL `http://www.springerlink.com/index/10.1007/b94608`.

M. Healy. A rotation method for computing canonical correlations. *Mathematics of Computation*, 11(58):83–86, 1957.

A. P. Holmes. Statistical issues in functional brain mapping. 1994.

H. Hotelling. Relations between two sets of variates. *Biometrika*, 28(3/4):321–377, 1936.

T. Insel, B. Cuthbert, M. Garvey, R. Heinssen, D. S. Pine, K. Quinn, C. Sanislow, and P. Wang. Research domain criteria (RDoC): toward a new classification framework for research on mental disorders. *American Journal of Psychiatry*, 167(7):748–751, 2010.

C. Jack, R. Petersen, Y. Xu, P. O'brien, G. Smith, R. Ivnik, B. Boeve, E. Tangalos, and E. Kokmen. Rates of hippocampal atrophy correlate with change in clinical status in aging and AD. *Neurology*, 55(4):484–490, 2000.

M. L. Keightley, D. A. Seminowicz, R. M. Bagby, P. T. Costa, P. Fossati, and H. S. Mayberg. Personality influences limbic-cortical interactions during sad mood induction. *NeuroImage*, 20(4):2031–2039, 2003a.

M. L. Keightley, G. Winocur, S. J. Graham, H. S. Mayberg, S. J. Hevenor, and C. L. Grady. An fMRI study investigating cognitive modulation of brain regions associated with emotional processing of visual stimuli. *Neuropsychologia*, 41(5):585–596, 2003b.

S. Klöppel, C. M. Stonnington, C. Chu, B. Draganski, R. I. Scahill, J. D. Rohrer, N. C. Fox, C. R. Jack, J. Ashburner, and R. S. Frackowiak. Automatic classification of MR scans in Alzheimer's disease. *Brain*, 131(3):681–689, 2008.

A. Krishnan, L. J. Williams, A. R. McIntosh, and H. Abdi. Partial Least Squares (PLS) methods for neuroimaging: A tutorial and review. *NeuroImage*, 56(2): 455–475, 2011.

J. S. Labus, J. D. Van Horn, A. Gupta, M. Alaverdyan, C. Torgerson, C. Ashe-McNalley, A. Irimia, J.-Y. Hong, B. Naliboff, K. Tillisch, et al. Multivariate morphological brain signatures predict chronic abdominal pain patients from healthy control subjects. *Pain*, 156(8):1545, 2015.

P. L. Lai and C. Fyfe. Kernel and nonlinear canonical correlation analysis. *International Journal of Neural Systems*, 10(05):365–377, 2000.

K.-A. Lê Cao, D. Rossouw, C. Robert-Granié, and P. Besse. A sparse PLS for variable selection when integrating omics data. *Statistical applications in genetics and molecular biology*, 7(1):Article 35, 2008. ISSN 1544-6115.

K.-A. Lê Cao, P. G. P. Martin, C. Robert-Granié, and P. Besse. Sparse canonical methods for biological data integration: application to a cross-platform study. *BMC bioinformatics*, 10:34, 2009. ISSN 1471-2105.

K.-A. Lê Cao, S. Boitard, and P. Besse. Sparse PLS discriminant analysis: biologically relevant feature selection and graphical displays for multiclass problems. *BMC bioinformatics*, 12(1):253, 2011. ISSN 1471-2105.

E. Le Floch, V. Guillemot, V. Frouin, P. Pinel, C. Lalanne, L. Trinchera, A. Tenenhaus, A. Moreno, M. Zilbovicius, T. Bourgeron, S. Dehaene, B. Thirion, J. B. Poline, and E. Duchesnay. Significant correlation between a set of genetic polymorphisms and a functional brain network revealed by feature selection and sparse Partial Least Squares. *NeuroImage*, 63(1):11–24, 2012. ISSN 10538119.

D. Lin, J. Zhang, J. Li, V. D. Calhoun, H.-W. Deng, and Y.-P. Wang. Group sparse canonical correlation analysis for genomic data integration. *BMC bioinformatics*, 14(1):245, 2013.

D. Lin, V. D. Calhoun, and Y. P. Wang. Correspondence between fMRI and SNP data by group sparse canonical correlation analysis. *Medical Image Analysis*, 18 (6):891–902, 2014. ISSN 13618423.

A. Lykou and J. Whittaker. Sparse cca using a lasso with positivity constraints. *Computational Statistics & Data Analysis*, 54(12):3144–3157, 2010.

L. Mackey. Deflation Methods for Sparse PCA. *NIPS*, pages 1–8, 2008.

D. S. Marcus, A. F. Fotenos, J. G. Csernansky, J. C. Morris, and R. L. Buckner. Open access series of imaging studies: Longitudinal MRI data in nondemented and demented older adults. *Journal of Cognitive Neuroscience*, 22(12):2677–2684, 2009.

J. C. Mazziotta, A. W. Toga, A. Evans, P. Fox, and J. Lancaster. A probabilistic atlas of the human brain: Theory and rationale for its development: The international consortium for brain mapping (ICBM). *Neuroimage*, 2(2):89–101, 1995.

A. McIntosh, F. Bookstein, J. Haxby, and C. Grady. Spatial Pattern Analysis of Functional Brain Images Using Partial Least Squares. *NeuroImage*, 3(3):143–157, 1996.

N. Meinshausen and P. Bühlmann. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473, 2010.

K. L. Miller, F. Alfaro-Almagro, N. K. Bangerter, D. L. Thomas, E. Yacoub, J. Xu, A. J. Bartsch, S. Jbabdi, S. N. Sotiropoulos, J. L. Andersson, et al. Multimodal population brain imaging in the uk biobank prospective epidemiological study. *Nature Neuroscience*, 2016.

M. Misaki, G. L. Wallace, N. Dankner, A. Martin, and P. A. Bandettini. Characteristic cortical thickness patterns in adolescents with autism spectrum disorders: interactions with age and intellectual ability revealed by canonical correlation analysis. *Neuroimage*, 60(3):1890–1901, 2012.

R. Mohs. *Administration and scoring manual for the Alzheimer's Disease Assessment Scale, 1994 revised edition*, 1994.

J. M. Monteiro, A. Rao, J. Ashburner, J. Shawe-Taylor, and J. Mourão-Miranda. Leveraging clinical data to enhance localization of brain atrophy. In *International Workshop on Machine Learning and Interpretation in Neuroimaging*, pages 60–68. Springer, 2014.

J. M. Monteiro, A. Rao, J. Ashburner, J. Shawe-Taylor, and J. Mourão Miranda. Multivariate effect ranking via adaptive sparse PLS. In *Pattern Recognition in NeuroImaging (PRNI), 2015 International Workshop on*, pages 25–28. IEEE, 2015.

J. M. Monteiro, A. Rao, J. Shawe-Taylor, and J. Mourão-Miranda. A multiple hold-out framework for sparse partial least squares. *Journal of Neuroscience Methods*, 271:182–194, 2016.

E. Moradi, I. Hallikainen, T. Hänninen, J. Tohka, A. D. N. Initiative, et al. Rey's auditory verbal learning test scores can be predicted from whole brainmri in alzheimer's disease. *NeuroImage: Clinical*, 2016.

J. Mourão-Miranda, A. L. Bokde, C. Born, H. Hampel, and M. Stetter. Classifying brain states and determining the discriminating activation patterns: support vector machine on functional MRI data. *NeuroImage*, 28(4):980–995, 2005.

J. Mourao-Miranda, A. Reinders, V. Rocha-Rego, J. Lappin, J. Rondina, C. Morgan, K. D. Morgan, P. Fearon, P. B. Jones, G. A. Doody, et al. Individualized prediction of illness course at the first psychotic episode: a support vector machine mri study. *Psychological medicine*, 42(5):1037, 2012.

P. G. Nestor, B. F. O'Donnell, R. W. McCarley, M. Niznikiewicz, J. Barnard, Z. J. Shen, F. L. Bookstein, and M. E. Shenton. A new statistical method for testing hypotheses of neuropsychological/MRI relationships in schizophrenia: partial least squares analysis. *Schizophrenia Research*, 53(1–2):57–66, 2002.

T. E. Nichols and A. P. Holmes. Nonparametric Permutation Tests for PET functional Neuroimaging Experiments: A Primer with examples. *Human Brain Mapping*, 15 (August 1999):1–25, 2001. ISSN 1065-9471.

I. Nouretdinov, S. G. Costafreda, A. Gammerman, A. Chervonenkis, V. Vovk, V. Vapnik, and C. H. Fu. Machine learning classification with confidence: application of

transductive conformal predictors to MRI-based diagnostic and prognostic markers in depression. *Neuroimage*, 56(2):809–813, 2011.

L. Nyberg, A. R. McIntosh, R. Cabeza, R. Habib, S. Houle, and E. Tulving. General and specific brain regions involved in encoding and retrieval of events: what, where, and when. *Proceedings of the National Academy of Sciences*, 93(20):11280–11285, 1996.

K. Nybo, J. Shawe-Taylor, S. Kaski, and J. Mourao-Miranda. Data-driven selection of relevant fmri voxels using sparse cca and stability selection.

M. J. Olson Hunt, L. Weissfeld, R. M. Boudreau, H. Aizenstein, A. B. Newman, E. M. Simonsick, D. R. Van Domelen, F. Thomas, K. Yaffe, and C. Rosano. A variant of sparse partial least squares for variable selection and data exploration. *Frontiers in neuroinformatics*, 8(March):18, 2014. ISSN 1662-5196.

G. Orrù, W. Pettersson-Yeo, A. F. Marquand, G. Sartori, and A. Mechelli. Using support vector machine to identify imaging biomarkers of neurological and psychiatric disease: a critical review. *Neuroscience & Biobehavioral Reviews*, 36(4): 1140–1152, 2012.

E. Parkhomenko, D. Tritchler, and J. Beyene. Sparse canonical correlation analysis with application to genomic data integration. *Statistical Applications in Genetics and Molecular Biology*, 8(1):1–34, 2009.

F. Pereira, T. Mitchell, and M. Botvinick. Machine learning classifiers and fmri: a tutorial overview. *Neuroimage*, 45(1):S199–S209, 2009.

R. C. Petersen, R. G. Thomas, M. Grundman, D. Bennett, R. Doody, S. Ferris, D. Galasko, S. Jin, J. Kaye, A. Levey, et al. Vitamin e and donepezil for the treatment of mild cognitive impairment. *New England Journal of Medicine*, 352 (23):2379–2388, 2005.

R. Pfeffer, T. Kurosaki, C. Harrah, J. Chance, and S. Filos. Measurement of functional activities in older adults in the community. *Journal of gerontology*, 37(3):323–329, 1982.

E. Polajnar. Using Restricted CCA for Cross-language Information Retrieval. *Communications in Statistics - Simulation and Computation*, 0918(June 2016): 0–0, 2015. ISSN 0361-0918. doi: 10.1080/03610918.2015.1122054. URL `http://www.tandfonline.com/doi/full/10.1080/03610918.2015.1122054`.

J. Price, S. K. Ziolko, L. Weissfeld, W. E. Klunk, X. Lu, J. A. Hoge, C. Meltzer, S. Davis, B. J. Lopresti, D. P. Holt, S. T. DeKosky, and C. A. Mathis. Quantitative and statistical analyses of pet imaging studies of amyloid deposition in humans. In *Nuclear Science Symposium Conference Record, 2004 IEEE*, volume 5, pages 3161–3164, 2004.

J. Qian, T. Hastie, J. Friedman, R. Tibshirani, and N. Simon. Glmnet for Matlab. `http://www.stanford.edu/~hastie/glmnet_matlab/`, 2013.

A. Rao, P. Aljabar, and D. Rueckert. Hierarchical statistical shape analysis and prediction of sub-cortical brain structures. *Medical Image Analysis*, 12(1):55–68, 2008.

A. Rao, Y. Lee, A. Gass, and A. Monsch. Classification of Alzheimer's Disease from structural MRI using sparse logistic regression with optional spatial regularization. In *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pages 4499–4502. IEEE, 2011.

A. Rao, J. M. Monteiro, J. Ashburner, L. Portugal, O. Fernandes, L. De Oliveira, M. Pereira, and J. Mourão-Miranda. A comparison of strategies for incorporating nuisance variables into predictive neuroimaging models. In *Pattern Recognition in NeuroImaging (PRNI), 2015 International Workshop on*, pages 61–64. IEEE, 2015.

A. Rao, J. Monteiro, and J. Mourao-Miranda. Prediction of clinical scores from neuroimaging data with censored likelihood Gaussian processes. In *Pattern Recognition in Neuroimaging (PRNI), 2016 International Workshop on*, pages 1–4. IEEE, 2016.

A. Rao, J. M. Monteiro, and J. Mourão-Miranda. Predictive modelling using neuroimaging data in the presence of confounds. *NeuroImage*, 2017.

J. M. Rondina, T. Hahn, L. de Oliveira, A. F. Marquand, T. Dresler, T. Leitner, A. J. Fallgatter, J. Shawe-Taylor, and J. Mourao-Miranda. SCoRS—A method based

on stability for feature selection and mapping in neuroimaging. *IEEE transactions on medical imaging*, 33(1):85–98, 2014.

M. J. Rosa, M. A. Mehta, E. M. Pich, C. Risterucci, F. Zelaya, A. A. Reinders, S. C. Williams, P. Dazzan, O. M. Doyle, and A. F. Marquand. Estimating multivariate similarity between neuroimaging datasets with sparse canonical correlation analysis: an application to perfusion imaging. *Frontiers in neuroscience*, 9, 2015.

M. J. Rosa, M. Moutoussis, G. Ziegler, J. M. Monteiro, L. Portugal, F. S. Ferreira, E. T. Bullmore, P. Fonagy, I. M. Goodyer, P. B. Jones, the NSPN Consortium, R. Dolan, and J. Mourao-Miranda. Brain-behavior modes of covariation in healthy and clinically depressed young people. *Scientific Reports*, (in preparation).

W. G. Rosen, R. C. Mohs, and K. L. Davis. A new rating scale for alzheimer's disease. *The American journal of psychiatry*, 1984.

R. Rosipal and N. Krämer. Overview and recent advances in partial least squares. In *Subspace, latent structure and feature selection*, pages 34–51. Springer, 2006.

J. Rousu, D. D. Agranoff, O. Sodeinde, J. Shawe-Taylor, and D. Fernandez-Reyes. Biomarker discovery by sparse canonical correlation analysis of complex clinical phenotypes of tuberculosis and malaria. *PLoS Comput Biol*, 9(4):e1003018, 2013.

M. Schmidt et al. *Rey auditory verbal learning test: a handbook*. Western Psychological Services Los Angeles, 1996.

J. Schrouff, J. M. Monteiro, L. Portugal, M. J. Rosa, C. Phillips, and J. Mourão-Miranda. Embedding anatomical or functional knowledge in whole-brain multiple kernel learning models. *Neuroinformatics*, (accepted).

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

S. M. Smith, T. E. Nichols, D. Vidaurre, A. M. Winkler, T. E. Behrens, M. F. Glasser, K. Ugurbil, D. M. Barch, D. C. Van Essen, and K. L. Miller. A positive-negative mode of population covariation links brain connectivity, demographics and behavior. *Nature neuroscience*, 18(11):1565–1567, 2015.

L. Song, B. Boots, S. M. Siddiqi, G. J. Gordon, and A. Smola. Hilbert space embeddings of hidden markov models. 2010.

G. Strang. *Linear Algebra and Its Applications*. Cole Thomson Learning Inc, 4th edition, 2006.

C. Sudlow, J. Gallacher, N. Allen, V. Beral, P. Burton, J. Danesh, P. Downey, P. Elliott, J. Green, M. Landray, et al. Uk biobank: an open access resource for identifying the causes of a wide range of complex diseases of middle and old age. *PLoS medicine*, 12(3):e1001779, 2015.

J. Sui, H. He, J. Liu, Q. Yu, T. Adali, G. D. Pearlson, and V. D. Calhoun. Three-way FMRI-DTI-methylation data fusion based on mCCA+ jICA and its application to schizophrenia. In *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, pages 2692–2695. IEEE, 2012.

J. Sui, G. D. Pearlson, Y. Du, Q. Yu, T. R. Jones, J. Chen, T. Jiang, J. Bustillo, and V. D. Calhoun. In search of multimodal neuroimaging biomarkers of cognitive deficits in schizophrenia. *Biological psychiatry*, 78(11):794–804, 2015a.

X. Sui, S. Li, J. Liu, X. Zhang, C. Yu, and T. Jiang. Sparse canonical correlation analysis reveals correlated patterns of gray matter loss and white matter impairment in alzheimer's disease. In *Biomedical Imaging (ISBI), 2015 IEEE 12th International Symposium on*, pages 470–473. IEEE, 2015b.

P. M. Thompson, J. L. Stein, S. E. Medland, D. P. Hibar, A. A. Vasquez, M. E. Renteria, R. Toro, N. Jahanshad, G. Schumann, B. Franke, et al. The enigma consortium: large-scale collaborative analyses of neuroimaging and genetic data. *Brain imaging and behavior*, 8(2):153–182, 2014.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

N. Tzourio-Mazoyer, B. Landeau, D. Papathanassiou, F. Crivello, O. Etard, N. Delcroix, B. Mazoyer, and M. Joliot. Automated anatomical labeling of activations in spm using a macroscopic anatomical parcellation of the mni mri single-subject brain. *Neuroimage*, 15(1):273–289, 2002.

V. Uurtio, M. Bomberg, K. Nybo, M. Itävaara, and J. Rousu. Canonical correlation methods for exploring microbe-environment interactions in deep subsurface. In *International Conference on Discovery Science*, pages 299–307. Springer, 2015.

V. Uurtio, J. M. Monteiro, J. Kandola, J. Shawe-Taylor, D. Fernandez-Reyes, and J. Rousu. A tutorial on canonical correlation methods. *ACM Computing Surveys*, (accepted).

E. van der Burg. *Nonlinear canonical correlation and some related techniques*, volume 11. DSWO press, 1988.

S. Van Vaerenbergh, J. Vía, and I. Santamaría. Adaptive kernel canonical correlation analysis algorithms for nonparametric identification of wiener and hammerstein systems. *EURASIP Journal on Advances in Signal Processing*, 2008(1):875351, 2008.

G. Varoquaux, P. R. Raamana, D. A. Engemann, A. Hoyos-Idrobo, Y. Schwartz, and B. Thirion. Assessing and tuning brain decoders: cross-validation, caveats, and guidelines. *NeuroImage*, 145:166–179, 2017.

S. Waaijenborg, P. C. Verselewel de Witt Hamer, and A. H. Zwinderman. Quantifying the association between gene expressions and DNA-markers by penalized canonical correlation analysis. *Statistical applications in genetics and molecular biology*, 7 (1):Article3, 2008. ISSN 1544-6115.

J. Wan, S. Kim, M. Inlow, K. Nho, S. Swaminathan, S. L. Risacher, S. Fang, M. W. Weiner, M. F. Beg, L. Wang, et al. Hippocampal surface mapping of genetic risk factors in ad via sparse learning models. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 376–383. Springer, 2011.

D. Wang, L. Shi, D. S. Yeung, and E. Tsang. Nonlinear canonical correlation analysis of fmri signals using hdr models. In *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, pages 5896–5899. IEEE, 2005.

J. Wegelin. A survey of Partial Least Squares (PLS) methods, with emphasis on the two-block case, 2000.

I. Wilms and C. Croux. Sparse canonical correlation analysis from a predictive point of view. *Biometrical Journal*, 57(5):834–851, 2015.

D. M. Witten and R. J. Tibshirani. Extensions of sparse canonical correlation analysis with applications to genomic data. *Statistical applications in genetics and molecular biology*, 8(1):1–27, 2009.

D. M. Witten, R. Tibshirani, and T. Hastie. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, page kxp008, 2009.

H. Wold. Causal flows with latent variables: partings of the ways in the light of nipals modelling. *European Economic Review*, 5(1):67–86, 1974.

H. Wold. Partial least squares. *Encyclopedia of statistical sciences*, 1985.

H. Yoshida, A. Kawaguchi, and K. Tsuruya. Radial basis function-sparse partial least squares for application to brain imaging data. *Computational and mathematical methods in medicine*, 2013, 2013.

A. L. Young, N. P. Oxtoby, P. Daga, D. M. Cash, N. C. Fox, S. Ourselin, J. M. Schott, and D. C. Alexander. A data-driven model of biomarker changes in sporadic alzheimer's disease. *Brain*, 137(9):2564–2577, 2014.

W. Zheng, X. Zhou, C. Zou, and L. Zhao. Facial expression recognition using kernel canonical correlation analysis (kcca). *IEEE Transactions on Neural Networks*, 17 (1):233–238, 2006.

G. Ziegler, R. Dahnke, A. Winkler, and C. Gaser. Partial least squares correlation of multivariate cognitive abilities and local brain structure in children and adolescents. *NeuroImage*, 82(0):284–294, 2013.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 67: 301–320, 2005. ISSN 13697412.

Glossary

accuracy 44

Alternating Least Squares (ALS) 32, 118, 120

Alzheimer's Disease (AD) 71, 102

Alzheimer's Disease Assessment Scale (ADAS) 124

Canonical Correlation Analysis (CCA) 31, 56

Cerebrospinal Fluid (CSF) 29, 163

Clinical Dementia Rating (CDR) 78, 88

Cross-Validation (CV) 45

data matrix 33

deflation 52

Diffusion Tensor Imaging (DTI) 71

dimension 33

dual representation 43

eigen-decomposition 52

eigenvalue 51

eigenvector 51

elastic net 38, 137

feature 33

feature selection 38

Frontotemporal Dementia (FTD) 71

Full Width at Half Maximum (FWHM) 30, 78

Functional Assessment Questionnaire (FAQ) 124

Functional Magnetic Resonance Imaging (fMRI) 68

Grey Matter (GM) 29

hyper-parameter 35, 45

kernel 40

Kernel Ridge Regression (KRR) 43

kernel trick 41

label 34

Least Absolute Shrinkage and Selection Operator (LASSO) 38

linear regression 36

loss function 35

Magnetic Resonance Imaging (MRI) 29

Mean Squared Error (MSE) 44

Mild Cognitive Impairment (MCI) 102

Mini-Mental State Examination (MMSE) 71, 78, 88, 95, 103, 124

nested cross-validation (nested-CV) 47

objective function 35

omnibus hypothesis 101

optimisation problem 36

orthonormal 52

overfitting 35

Partial Least Squares (PLS) 31, 57, 59

Partial Least Squares Regression (PLSR) 69

penalty 35

permutation test 48

power method 53

primal representation 43

primal-dual 153, 155

Principal Component Analysis (PCA) 54

projection 56

Region Of Interest (ROI) 70

Rey Auditory Verbal Learning Test (RAVALT) 124

ridge penalty 38

sample 33

Single Nucleotide Polymorphism (SNP) 70

singular value 54

Singular Value Decomposition (SVD) 53

Socioeconomic Status (SES) 78, 88

sparse 38

Sparse Canonical Correlation Analysis (SCCA) 64

Sparse Partial Least Squares (SPLS) 61

Sparse Partial Least Squares Discrimination Analysis (SPLS-DA) 72

stability selection 66

target 34

underfitting 35

view 56

voxel 29

Voxel-Based Morphometry (VBM) 30

weight vector 36

White Matter (WM) 29