ADVANCES IN MULTIVARIATE STATISTICS AND ITS APPLICATIONS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF STATISTICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Elena Tuzhilina
May 2022

© Copyright by Elena Tuzhilina 2022
All Rights Reserved

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Trevor Hastie) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Robert Tibshirani)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Mark Segal)

Approved for the Stanford University Committee on Graduate Studies

Preface

My research focuses on multivariate statistics, dimension reduction, and applied statistical modeling. During my Ph.D. studies at the Department of Statistics at Stanford, I took part in various collaborative projects, developing methodology and tools for analysis of complex phenomena in such areas as biology, genetics, and neuroscience. This dissertation will cover four big branches of my research, which I will present in separate chapters.

Principal curves and conformation reconstruction Conformation reconstruction is one of the main challenges in computational biology. In this study we develop a model for the 3D spatial organization of chromatin, a crucial component of numerous cellular processes (e.g. transcription). The central object in this study is the so-called contact matrix. It represents the frequency of contacts between each pair of genomic loci and thus can be used to infer the 3D structure. Most of the existing algorithms operating on contact matrices are based on multidimensional scaling (MDS) and produce reconstructed 3D configurations in the form of a polygonal chain. However, none of the methods exploit the fact that the target solution is a smooth curve in 3D. The smoothness attribute is either ignored or indirectly addressed via introducing highly non-convex penalties in the model. This typically leads to increased computational complexity and instability of the reconstruction algorithm. In our work we develop Principal Curve Metric Scaling (PCMS), a novel approach modeling chromatin directly by a smooth curve. We subsequently use PCMS as a building block to create more complex distribution-based models for the conformation. The resulting reconstruction technique therefore combines advantages of MDS and smoothness penalties whereas being computationally efficient.

Weighted low-rank matrix approximation Low-rank matrix approximation (LRMA) is one of the central concepts in machine learning. It is closely related to such areas as dimension reduction and de-noising. A recent extension to LRMA is called low-rank matrix completion. It solves the LRMA problem when some observations are missing and is especially useful for recommender systems (see, for example, the famous Netflix Prize competition). In this study we consider a weighted generalization of LRMA. We build an algorithm for solving the weighted problem as well as two

important modifications: one for high-dimensional, one for sparse data. In addition, we propose an efficient way to accelerate the WLRMA algorithm. Although our previous research mainly focuses on developing the WLRMA methodology, the technique has a strong potential for applications. Beyond matrix completion, which it covers as a special case, it can serve as a building block for generalized linear models (GLM) with a matrix structure. For example, in ecology, populations of species can be modeled via Poisson GLMs. In this case, the population matrices (with rows and columns corresponding to sites and species, respectively) can be analyzed using low-rank models and the WLRMA technique will be of great importance.

Canonical correlation analysis and brain studies Canonical correlation analysis (CCA) is one of the core approaches in multivariate statistics. It is a technique for measuring the association between two multivariate sets of variables, which has a wide variety of applications. This part of the research was motivated by a study in neuroscience aimed to explore the influence of emotional disorders on brain activity. While working with the brain imaging data we encountered the following challenges. First, the measurements are made for a very dense grid of brain loci leading to extremely high-dimensional data. Second, the data was collected only for a few patients; therefore, it is underrepresented. Finally, the data has a structure, which is defined by the brain geometry. To address the first two challenges we consider Regularized CCA and develop a "kernel trick" that allows us to handle extreme data size. We subsequently incorporate brain structure in the regularization introducing Group Regularized CCA (GRCCA) and extend the "kernel trick" to the structured data setting. The resulting GRCCA technique has demonstrated strong potential for brain imaging applications while being computationally efficient.

Multi-period forecasting and COVID-19 pandemic Epidemic forecasting became a very in-demand area during the COVID-19 era. In this research, we have studied the trajectory of the COVID-19 pandemic by means of the open-source COVIDcast dataset collected by Delphi Group. This dataset contains a wide variety of features such as cases, deaths, hospitalizations, and many auxiliary indicators of COVID-19 activity and therefore opens up a wealth of research directions. In particular, we develop the multi-period forecasting (MPF) methodology, which aims to predict the number of cases for multiple "ahead" values. The MPF technique solves a multi-response regression problem, where the response columns represent the same phenomenon measured at different time points. To incorporate this time dependence, it assumes the model coefficients to be smooth functions depending on time. We test this idea for the point estimation of the COVID-19 cases and subsequently extend it to predicting the cases' confidence intervals via quantile regression.

Acknowledgments

The past five years were the happiest and the most fulfilling years in my life, despite all the challenges. I will greatly miss Stanford, my life as a Ph.D student here, my friends and my colleagues from Sequoia Hall.

My dear advisor Professor Hastie, there are no words to express how grateful I am. Thank you for always being so kind to me: for accepting me, for being supportive in all my research efforts, for helping to cope with my fears and self-doubts, and for boosting my confidence. Thank you and Lynda for making me feel welcomed and for taking so much care of me both as a student and a human being. I have learned so many lessons from you: how to be generous, committed, and curious in research, how to balance between applications and mathematical rigorosity, how to tackle complex research ideas. ("The funniest lessons are related to the paper-writing, though: Elena, do not start the section with "Denote by X . . . "! Why do you motivate your paper with a math equation?!") I hope I was able to absorb at least a tiny portion of your scientific intuition and teaching skills, and I hope I will have a chance to convey them to my future students.

My precious family Mom and Dad, I am grateful for all the effort and endless time you put into nurturing my mind and inculcating your values in me. You allured me into loving of reasoning and knowledge, and you have always been my main encouragement and inspiration in pursuing my dreams. Denis, my incredible husband, and my best ally, you are the unfailing source of moral support and my emotional well-being. Thank you for your unconditional belief in me and for being willing to discuss my (sometimes wild) research ideas. This Ph.D. would be infeasible without you standing behind my back. My brother, grandparents, and parents-in-law, I am grateful to you for being so caring and comforting. You kept my heart warm and empowered me to overcome all the difficulties I encountered while living far from home. Finally, my dear uncle Alex, thank you for all your priceless advice and morale boosts during my job search this year (it was tough!).

My amazing collaborators Robert Tibshirani, Mark Segal, and Leonardo Tozzi, you spent hours and hours discussing and elaborating on research ideas with me and you helped me to learn so much about applications. I am thankful for your patience and infinite guidance, and I am incredibly fortunate to have worked with you all. Professor Tibshirani, I would also like to express my sincere gratitude to you for being supportive during my job search and for your kind help in mastering my

job talk. My colleagues from Williams PanLab and Delphi Research Group, it was a great honor and pleasure to work with you, and I hope to have more exciting joint projects with you in the nearest future. My peers and senior colleagues from Stanford Data Science Institute, thank you for creating a very fruitful and engaging environment and for all the fascinating discussions on the data science topic.

My awesome Stanford friends Olga Fuks, Suyash Gupta, Disha Ghandwani, Michael Feldman, Shuangning Li, Ismael Lemhadri, Claire Donnat, Mona Azadkia and Pragya Sur, thank you for all the happy moments, serious conversations, hilarious occasions, triumphant victories, and invigorating struggles we went through together. I was fortunate to have you on my side and share this exciting adventure with you. I hope to keep our bonds tight after graduation. Here I cannot help but mention my non-Stanford friends as well. Anna Doledenok, Daniil Trifonov, Anatoly Antipov, Elizaveta Senatorova, and Alexander Kagan, although I saw you only a few times per year, you kept me deeply connected to home. Thank you all for creating so many happy memories that supported me throughout a difficult time.

My home Department Iain Johnston, John Duchi, Emmanuel Candès, Julia Palacios, Bradley Efron, Trevor Hastie, Andrea Montanari, David Siegmund, Sourav Chatterjee, and Jerome Friedman, I am sincerely grateful to you for taking part in my education as a statistician. Statistical Learning group, particularly Robert Tibshirani and Jonathan Taylor, thank you for so many valuable inputs and helpful suggestions on my research. I would also like to add some special thanks to Balasubramanian Narasimhan and John Chambers. I deeply appreciate all the exciting discussions we had and your help connecting me to the Stanford data science community. Finally, dear staff of the Department of Statistics and, especially, my favorite problem solver Susie Ementon who helped me on various occasions so many times: thank you! You are the most efficient and skilled people who are not only able to manage fifty desperate Ph.D. students at once but also take good care of their mental health. I am very grateful for all the advice and assistance I received from you over the past five years.

My wonderful students Students I had a chance to interact with either as a TA or instructor, I greatly appreciate the time we spent together scrutinizing various statistical concepts and problems. You helped me perceive how important and inspiring sharing knowledge is and how rewarding teaching and mentoring can be.

My dear creators of the ELSII book Trevor Hastie, Robert Tibshirani, and Jerome Friedman, I am grateful to you for writing this book. In Spring 2016, this book made a very drastic twist in my life and completely changed my trajectory. Even though I do not believe in destiny, it was such an elegant coincidence and a stroke of pure luck.

Contents

Preface **iv**

Acknowledgments **vi**

1 Principal curves and conformation reconstruction **2**

1.1 Introduction and background . . . 2
1.1.1 Hi-C assays . . . 2
1.1.2 Existing approaches to 3D chromatin reconstruction . . . 3
1.2 Principal curve metric scaling . . . 6
1.2.1 Metric scaling with a smooth curve constraint . . . 6
1.2.2 The *PCMS* solution via eigen-decomposition . . . 6
1.2.3 Chromatin reconstruction via *PCMS* . . . 7
1.3 Determination of principal curve degrees-of-freedom . . . 8
1.4 Weighted principal curve metric scaling . . . 10
1.4.1 A distance-based model for chromatin reconstruction . . . 10
1.4.2 Projected gradient descent and the *WPCMS* algorithm . . . 11
1.4.3 Chromatin reconstruction via *WPCMS* . . . 13
1.5 Poisson metric scaling . . . 14
1.5.1 Poisson model for contact counts . . . 14
1.5.2 Newton's method and the *PoisMS* algorithm . . . 14
1.5.3 Computational complexity . . . 18
1.5.4 Chromatin reconstruction via *PoisMS* . . . 18
1.6 Assessment of the reconstruction accuracy . . . 18
1.6.1 Multiplex FISH . . . 18
1.6.2 A contrasting reconstruction algorithm: HSA . . . 21
1.6.3 Validation of the *PoisMS* reconstructions . . . 22
1.7 Discussion . . . 22
1.8 Appendix . . . 24

1.8.1 The *WPCMS* algorithm extensions . . . 24
1.8.2 The *PoisMS* algorithm extensions . . . 26
1.8.3 *PoisMS* reconstructions for IMR90 cell chromosome 21 . . . 28

2 Weighted low-rank matrix approximation **32**
2.1 Introduction and background . . . 32
2.1.1 Low-rank matrix approximation . . . 32
2.1.2 Missing values . . . 32
2.1.3 Weighted generalization . . . 34
2.2 Link to proximal gradient descent . . . 35
2.3 Acceleration Techniques . . . 37
2.3.1 Nesterov . . . 38
2.3.2 Anderson . . . 38
2.4 Simulation . . . 40
2.5 Stabilizing Anderson acceleration coefficients . . . 41
2.6 High dimensions . . . 44
2.6.1 Alternating least squares . . . 44
2.6.2 Sparse weights . . . 45
2.6.3 Acceleration in high dimensions . . . 47
2.6.4 Find the solution rank . . . 47
2.7 MovieLens data . . . 48
2.8 Discussion . . . 48

3 Canonical correlation analysis and brain studies **51**
3.1 Introduction and background . . . 51
3.2 Canonical correlation analysis . . . 52
3.3 Regularized canonical correlation analysis . . . 53
3.3.1 Dealing with high dimensions . . . 53
3.3.2 Shrinkage property . . . 54
3.3.3 The *RCCA* kernel trick . . . 54
3.3.4 Hyperparameter tuning . . . 55
3.4 Human Connectome data study . . . 56
3.4.1 Correlation analysis via *RCCA* . . . 56
3.4.2 Interpretability of canonical coefficients . . . 59
3.5 Partially regularized canonical correlation analysis . . . 59
3.5.1 Penalizing a subset of canonical coefficients . . . 59
3.5.2 The *PRCCA* kernel trick . . . 61
3.5.3 Testing *PRCCA* on the Human Connectome data . . . 63

3.6 Group regularized canonical correlation analysis . . . 64
3.6.1 Handling data with a group structure . . . 64
3.7 General regularized canonical correlation analysis . . . 67
3.8 General penalty matrix . . . 67
3.8.1 Link to the flexibility vs. performance trade-off . . . 68
3.9 Link between *GRCCA* and *RCCA*/*PRCCA* . . . 69
3.9.1 Approach 1: via the SVD of the penalty matrix . . . 69
3.9.2 Approach 2: via the feature matrix extension . . . 70
3.9.3 Analysis of the Human Connectome dataset via *GRCCA* . . . 71
3.9.4 Using GRCCA for visualization . . . 73
3.10 Simulation study . . . 73
3.11 Discussion . . . 77
3.12 Appendix . . . 79
3.12.1 Neuroimaging analysis . . . 79

4 Multi-period forecasting and COVID-19 pandemic 81
4.1 Introduction and background . . . 81
4.2 Forecasting problem . . . 82
4.3 Multi-period forecaster . . . 83
4.3.1 Baseline linear model . . . 83
4.3.2 Smoothing constraint . . . 84
4.4 Missing values . . . 86
4.5 Simulation experiment . . . 87
4.6 COVIDCast data experiments . . . 88
4.7 Quantile forecasting . . . 93
4.8 Quantile forecasting in COVIDCast study . . . 94
4.9 Conformal calibration . . . 95
4.10 Discussion . . . 98

Chapter 1

Principal curves and conformation reconstruction

1.1 Introduction and background

1.1.1 Hi-C assays

The three-dimensional (3D) configuration of chromosomes within the eukaryote nucleus is important for several cellular functions, including gene expression regulation, and has also been linked to translocation events and cancer driving gene fusions (Mitelman et al., 2007). While direct visualization of 3D architecture has improved (see Section 1.6), imaging challenges pertaining to chromatin compaction and dynamics persist. However, the ability to *infer* chromatin architectures at increasing resolution has been enabled by chromosome conformation capture (3C) assays (Dekker et al., 2002). In particular, when coupled with next generation sequencing, such Hi-C methods (Lieberman-Aiden et al., 2009; Duan et al., 2010) yield an inventory of pairwise, genome-wide chromatin interactions, or contacts. In turn, the contact data form the basis for *reconstructing* 3D configurations (Zhang et al., 2013; Varoquaux et al., 2014; Ay et al., 2014; Zou et al., 2016; Rieber and Mahony, 2017). While many novel conformational-related findings have flowed from direct analysis of contact level data, added value of performing downstream analysis based on attendant 3D reconstructions has been demonstrated. These benefits derive from the ability to superpose genomic features on the reconstruction. Examples include co-localization of genomic landmarks such as early replication origins in yeast (Witten and Noble, 2012; Capurso and Segal, 2014), gene expression gradients in relation to telomeric distance and co-localization of virulence genes in the malaria parasite (Ay et al., 2014), the impact of spatial organization on double strand break repair (Lee et al., 2016), and elucidation of '3D hotspots' corresponding to (say) overlaid ChIP-Seq transcription factor extremes which can reveal novel regulatory interactions (Capurso et al., 2016).

The contact or interaction matrices resulting from Hi-C assays, which are typically performed on bulk cell populations, are depicted as heatmaps, which record the frequency with which pairs of binned genomic loci are cross-linked, reflecting spatial proximity of the respective loci bins within the nucleus. A common first step toward 3D reconstruction is the conversion of contact frequencies into *distances*, typically assuming inverse power-law relationships (Varoquaux et al., 2014; Ay et al., 2014; Shavit et al., 2014; Rieber and Mahony, 2017), from which 3D chromatin architecture can be obtained via versions of the multi-dimensional scaling (MDS) paradigm. In response to (i) the bulk cell population underpinnings of contact data, (ii) computational challenges posed by the dimensionality of the MDS reconstruction problem as governed by bin extent, and (iii) accommodating biological considerations, several competing reconstruction algorithms have been advanced. However, none of these take advantage of the fact that the 3D solution for individual chromosomes corresponds to a one-dimensional (1D) curve in 3-space. Rather, this aspect has been addressed by imposition of constraints (Duan et al., 2010; Ay et al., 2014; Stevens et al., 2017), which are cell type specific and require prescription of constraint parameters. These parameters can be difficult to specify and their inclusion substantially increases the computational burden. Other approaches (Zhang et al., 2013; Park and Lin, 2017; Rieber and Mahony, 2017) do not formally incorporate contiguity but impose it post hoc, creating chromatin reconstructions by "connecting the dots" of the 3D solution according to the ordering of corresponding genomic bins. In this work we directly target chromosome reconstruction by finding a 1D curve approximation to the contact matrix via extending principal curve methodology (Hastie and Stuetzle, 1989) to the metric scaling problem.

This chapter is organized as follows. After reviewing problem formulation and current reconstruction techniques in Section 1.1.2, we develop two building blocks, *Principal Curve Metric Scaling* (*PCMS*; Section 1.2) and *Weighted PCMS* (*WPCMS*; Sections 1.4), that enable our novel *Poisson Metric Scaling* (*PoisMS*; Sections 1.5) approach. Strategies for selecting a specific reconstruction from a degrees-of-freedom indexed series of solutions are described in Section 1.3. Methods for appraising the accuracy of candidate reconstructions using orthogonal imaging data are outlined in Section 1.6.3. Results from applying the methodology to Hi-C data from IMR90 cells are presented in Sections 1.2.3, 1.4.3 and 1.5.4, while the Discussion indicates directions for future work.

1.1.2 Existing approaches to 3D chromatin reconstruction

Our focus is on reconstruction of *individual* chromosomes; whole genome architecture can follow by appropriately positioning these solutions (Segal and Bengtsson, 2015; Rieber and Mahony, 2017). As is standard, we disregard complexities deriving from chromosome pairing arising in diploid cells (which can be disentangled at high resolutions; Rao et al., 2014) and address issues surrounding bulk cell experiments and inter-cell variation in the Discussion.

The result of a Hi-C experiment is the *contact map*, a symmetric matrix $C = [C_{ij}] \in \mathbb{Z}_+^{n\times n}$ of

contact counts between n (binned) genomic loci i, j on a genome-wide basis; Figure 1.4 provides an example. We defer questions surrounding contact matrix normalization. This matrix can be exceedingly sparse, even after binning. The 3D chromatin reconstruction problem is to use the contact matrix C to obtain a 3D point configuration $x_1, \ldots, x_n \in \mathbb{R}^3$ corresponding to the spatial coordinates of loci $1, \ldots, n$ respectively; Figure 1.7 gives an illustration.

Many approaches have been proposed to tackle this problem with broad distinction between optimization and model-based methods (Varoquaux et al., 2014; Rieber and Mahony, 2017). A common first step is conversion of the contact matrix into a distance matrix $D = [D_{ij}]$ (Duan et al., 2010; Varoquaux et al., 2014; Ay et al., 2014; Shavit et al., 2014), followed by solving the *multi-dimensional scaling* (MDS; Hastie et al., 2009) problem: position points (corresponding to genomic loci) in 3D so that the resultant interpoint distances best conform to the distance matrix.

A variety of methods have also been used for transforming contacts to distances. At one extreme, in terms of imposing biological assumptions, are methods that relate observed intra-chromosomal contacts to genomic distances and then ascribe *physical* distances based on organism specific findings on chromatin packing (Duan et al., 2010) or relationships between genomic and physical distances for crumpled polymers (Ay et al., 2014). Such distances inform the subsequent optimization step as they permit incorporation of known biological constraints that can be expressed in terms of physical separation. Importantly, these constraints include prescriptions on the 3D separation between contiguous genomic bins. It is by this means that obtaining a 1D curve is indirectly facilitated. However, obtaining physical distances requires both strong assumptions and organism specific data (Fudenberg and Mirny, 2012). More broadly, a number of approaches (Zhang et al., 2013; Varoquaux et al., 2014; Zou et al., 2016; Rieber and Mahony, 2017) utilize power law transfer functions to map contacts to (non-physical) distances $D_{ij} = \begin{cases} (C_{ij})^{-\alpha} \text{ if } C_{ij} > 0, \\ \infty \text{ if } C_{ij} = 0. \end{cases}$ Adoption of the power law derives from empirical and theoretical work but again constitutes a strong assumption (Fudenberg and Mirny, 2012).

Once we have a distance matrix $\mathcal{D}$, optimization approaches seek a 3D configuration $x_1, \ldots, x_n$ that best fits $\mathcal{D}$ according to an MDS criterion. If $\|\cdot\|$ designates the Euclidean norm, then an example of MDS loss incorporating weights and penalty (Zhang et al., 2013) is

$$\ell(x_1, \ldots, x_n) = \sum_{\{i,j|D_{ij}<\infty\}} W_{ij}(\|x_i - x_j\| - D_{ij})^2 - \lambda \sum_{\{i,j|D_{ij}=\infty\}} \|x_i - x_j\|^2 \tag{1.1}$$

with the corresponding optimization problem

$$\underset{x_1,\ldots,x_n \in \mathbb{R}^3}{\text{minimize}} \; \ell(x_1, \ldots, x_n). \tag{1.2}$$

Here common choices for the weights W_{ij} include D_{ij}^{-1} (Zhang et al., 2013) and D_{ij}^{-2} (Varoquaux

et al., 2014), these being analogous to precision weighting since large C_{ij} (small D_{ij}) are more accurately measured. Similarly, the penalty (second) term maximizes the pairwise distances for loci bins with $C_{ij} = 0$ under the presumption that such loci should not be too close.

It is worth noting that (1.1), and related criteria, correspond to a nonconvex, nonlinear optimization problem that is NP hard and while various devices have been employed to mitigate the computational burden (e.g., Zhang et al., 2013), computational concerns, particularly for high resolution (many loci bins) problems, remain forefront.

Probabilistic methods model the contact counts with an optimization goal of maximizing the corresponding log-likelihood.

In particular, Poisson models, $C_{ij} \sim \text{Pois}(\lambda_{ij})$, are widely used (Varoquaux et al., 2014; Zou et al., 2016; Park and Lin, 2017), where $\lambda_{ij} = \lambda_{ij}(x_1, \ldots, x_n)$ is a function of the genomic loci spatial coordinates $x_1, \ldots, x_n$. For example, Rosenthal et al. (2019) prescribe exponential dependence between the Poisson rate parameter and inter-loci distances: $\lambda_{ij} = \beta\|x_i - x_j\|^\alpha$ for some $\alpha < 0$, a framework we slightly modify in Section 1.5.

All existing approaches implicitly represent chromatin as a polygonal chain. Constraints on the geometrical structure of the polygonal chain can be imposed via penalties on edge lengths and angles between successive edges, with even quaternion-based formulations employed (Caudai et al., 2015). Rosenthal et al. (2019) utilize penalties to control smoothness of the resulting conformations. However, despite imparting targeted properties to the resulting reconstruction, such penalty-based approaches increase the complexity of the objective, its gradient and Hessian, both slowing and limiting, especially with respect to resolution, associated algorithms.

Here we develop a suite of novel approaches that directly model chromatin configuration as a 1D curve in 3D. Our primary method, *Poisson metric scaling* (*PoisMS*), is based on a Poisson model for contact counts and provides an efficient means for obtaining smooth 1D reconstructions, that combines advantages of both MDS and probabilistic models. This technique utilizes two building blocks of intrinsic interest. First, we introduce the *principal curve metric scaling* (*PCMS*) approach that features an optimization problem inspired by MDS and stated in terms of inner products. This problem admits a simple solution obtained via the singular value decomposition. Next, we develop *weighted PCMS* (*WPCMS*), a weighted version of *PCMS* that, importantly, models distances rather than inner products and further permits control over the influence of particular elements of the contact matrix on the resulting reconstruction. This technique requires an iterative algorithm that uses *PCMS* as the core component. Finally, *WPCMS* in turn can be used in conjunction with projected gradient descent to solve a second order approximation of the Poisson log-likelihood, yielding our *PoisMS* algorithm.

1.2 Principal curve metric scaling

1.2.1 Metric scaling with a smooth curve constraint

The *PCMS* technique is based on classical MDS. Given a symmetric matrix Z, *PCMS* treats it as a similarity matrix and approximates it by an inner product matrix (Buja et al., 2008). In particular, Z can correspond to the contact matrix after conversion to a distance matrix followed by double centering, the standard MDS device that turns (Euclidean) squared distances into inner products.

Let $X = \begin{pmatrix} -x_1^T- \\ \cdots \\ -x_n^T- \end{pmatrix} \in \mathbb{R}^{n\times 3}$ be the matrix of genomic loci coordinates and let $S(X) = XX^T$ refer to the inner product matrix of the reconstruction X. If $\|\cdot\|_F$ denotes the Frobenius norm, then the goal is to minimize the *Strain* objective:

$$\ell(x_1,\ldots,x_n) = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(Z_{ij} - \langle x_i,\ x_j\rangle\right)^2 \quad\Longleftrightarrow\quad \ell(X) = \|Z - S(X)\|_F^2. \tag{1.3}$$

Instead of adding a smoothness penalty to the objective, we impose an additional constraint:

$$x_1,\ldots,x_n \in \gamma, \text{ where } \gamma \text{ is a smooth one-dimensional curve in } \mathbb{R}^3. \tag{1.4}$$

This constraint will serve to capture the inherent contiguity of chromatin. We model the curve γ by a cubic spline with k degrees-of-freedom as follows (Hastie et al., 2009). Suppose $h_1(t),\ldots,h_k(t)$ are cubic spline basis functions in $\mathbb{R}^1$ then

$$\gamma(t) = (\gamma_1(t),\gamma_2(t),\gamma_3(t))^T, \text{ where } \gamma_j(t) = \textstyle\sum_{\ell=1}^{k}\Theta_{\ell j}\ h_\ell(t) \text{ for } j = 1,2,3.$$

Let t_i index the genomic locus of x_i in the parameter space of γ, i.e. $x_i = \gamma(t_i)$, and $H \in \mathbb{R}^{n\times k}$ be the matrix of spline basis evaluations at t_i, i.e. $H_{i\ell} = h_\ell(t_i)$. Since binning typically results in evenly spaced genomic loci it is convenient to set $t_1 = 1,\ t_2 = 2,\ldots,t_n = n$, although irregular spacing is readily handled. So, the constraint (1.4) can be written as $X_{ij} = \sum_{\ell=1}^{k}\Theta_{\ell j}\ h_\ell(t_i)$, or equivalently, in matrix form as $X = H\Theta$ leading to the optimization problem

$$\underset{\Theta\in\mathbb{R}^{k\times 3}}{\text{minimize}}\, \ell_{PCMS}(\Theta) = \|Z - S(H\Theta)\|_F^2. \tag{1.5}$$

Hereafter we denote the corresponding solution by $\hat{\Theta} = PCMS(Z,H)$, the resulting chromatin reconstruction by $\hat{X} = H\hat{\Theta}$ and the approximation of the original matrix Z as $\hat{Z} = S(\hat{X})$.

1.2.2 The *PCMS* solution via eigen-decomposition

Note that the parameter Θ in the *PCMS* problem (1.5) is unconstrained. Since Θ is defined up to a multiplication by a full-rank matrix, we can assume H to be a matrix with orthogonal columns. To find the *PCMS* solution the following lemma is useful.

Lemma 1. *If $H \in \mathbb{R}^{n\times k}$ is a matrix with orthogonal columns, i.e. $H^TH = I$, then problem (1.5) is equivalent to*

$$\underset{\Theta\in\mathbb{R}^{k\times 3}}{\text{minimize}}\, \tilde{\ell}_{PCMS}(\Theta) = \|H^TZH - \Theta\Theta^T\|_F^2. \tag{1.6}$$

Proof. Suppose $\tilde{H} = \begin{pmatrix} H & H_\perp \end{pmatrix}$ is a column-wise combination of H and its orthogonal complement $H_\perp$. Therefore, $\tilde{H}$ is a square orthogonal matrix and for any $B \in \mathbb{R}^{n\times n}$ the following relation holds

$$\|B\|_F^2 = \|\tilde{H}^TB\|_F^2 = \|\tilde{H}^TB\tilde{H}\|_F^2 = \|H^TBH\|_F^2 + \|H_\perp^TBH_\perp\|_F^2.$$

Substituting $B = Z - H\Theta\Theta^TH^T$ we conclude that

$$\ell_{PCMS}(\Theta) = \|Z - H\Theta\Theta^TH^T\|_F^2 = \|H^TZH - \Theta\Theta^T\|_F^2 + \|H_\perp^TZH_\perp\|_F^2 = \tilde{\ell}_{PCMS}(\Theta) + const.$$

In the last equation the constant term does not depend on the parameter Θ implying the equivalence of the above two optimization problems. □

Minimizing $\tilde{\ell}_{PCMS}(\Theta)$ can be interpreted as approximating the matrix H^TZH by a positive semi-definite rank 3 matrix $\Theta\Theta^T$. Assuming that the symmetric matrix H^TZH has at least three positive eigenvalues the solution can be found via eigen-decomposition of H^TZH: let $H^TZH = Q\Lambda Q^T$ for orthogonal Q and diagonal $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$ with $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$, then

$$\Theta = Q\sqrt{\Lambda_3}, \text{ where } \sqrt{\Lambda_3} = \text{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \sqrt{\lambda_3}, 0, \ldots, 0).$$

The computational efficiency of *PCMS* derives from the fact that it relies on eigen-decomposition of a small $k \times k$ matrix, requiring only $O(k^3)$ additional operations.

1.2.3 Chromatin reconstruction via *PCMS*

In this section we present *PCMS* reconstructions for IMR90 cell chromosome 20 at 100kb resolution. By combining power law transformation with double centering we transform the original contact matrix to a suitable similarity matrix. Specifically, we first convert contacts to distances via $\mathcal{D}_{ij} = \frac{1}{C_{ij}+1}$. This corresponds to power law transformation with index $\alpha = -1$ (Lesne et al., 2014), where we add one in the denominator to avoid dividing by zero. Next, we use the centering operator $J = I - \frac{\mathbf{1}\mathbf{1}^T}{n}$ to transform distances to similarities via double centering: $Z = -\frac{1}{2}J\mathcal{D}^2J$ (Buja et al., 2008). The heatmap of the resulting similarity matrix is displayed in Figure 1.1.

Now, we apply *PCMS* to Z. The solution $\hat{\Theta} = PCMS(Z, H)$ leads to the reconstructions $\hat{X} = H\hat{\Theta}$ and approximations of the similarity matrix $Z \approx S(\hat{X})$ presented in Figure 1.2 for a range of degrees-of-freedom (*df*) values.

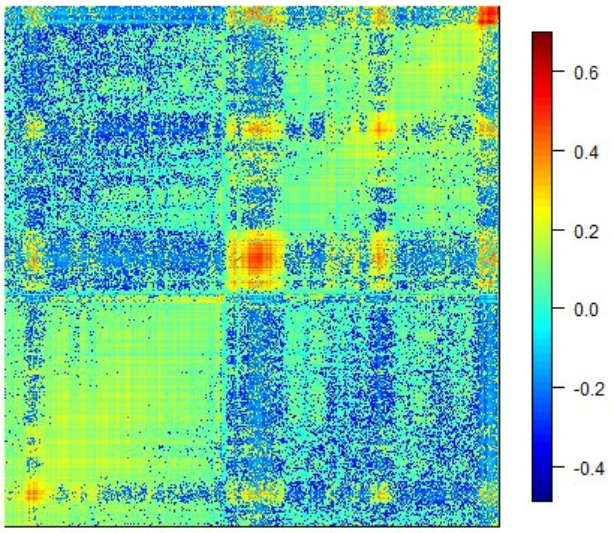


Figure 1.1: Transformed contact matrix $Z = -\frac{1}{2} J \left(\frac{1}{C+1}\right)^2 J$.

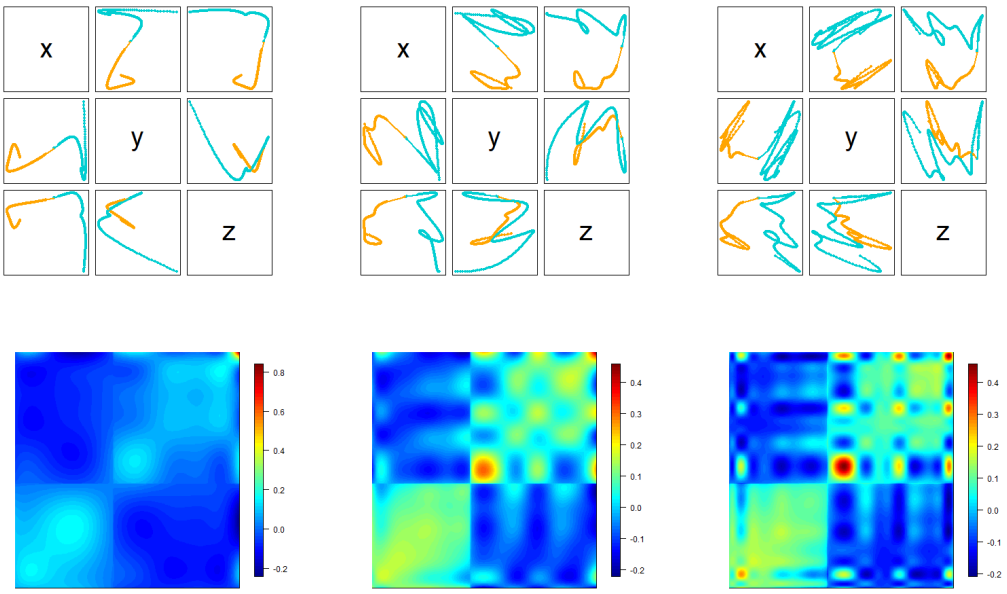
1.3 Determination of principal curve degrees-of-freedom

The main hyperparameter of the *PCMS* approach is the spline degrees-of-freedom *df* (spline basis size), which controls the smoothness of the resulting reconstruction. To determine the optimal value, for each *df* we create the spline basis matrix H_{df}, find the corresponding solution $\hat{\Theta}_{df}$ and the resulting reconstruction $\hat{X}_{df} = H_{df}\hat{\Theta}_{df}$. We measure the error rate by the normalized loss function. In the case of *PCMS* it corresponds to the normalized residual-sum-of-squares, i.e

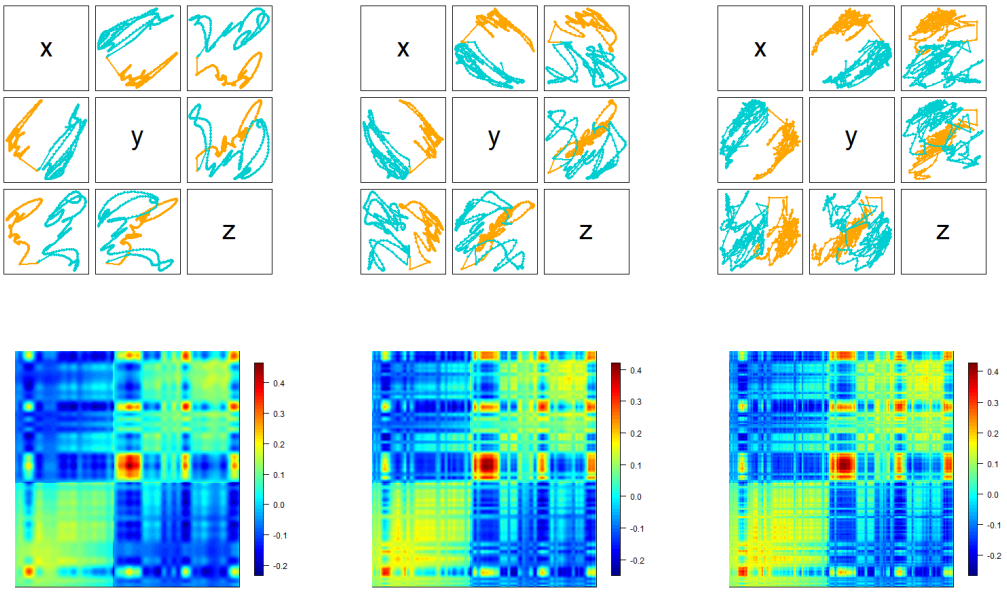
$$err(\hat{X}_{df}) = \frac{1}{n^2} \left\| Z - S(\hat{X}_{df}) \right\|_F^2 \tag{1.7}$$

Initially, we tried cross-validation to find the optimal value of *df*, as is common for smoothing (penalty) parameter determination. However, the complex and structural dependencies that characterize contact matrices made this approach problematic. As an alternative we adopted an approach based on identifying the "elbow" that is prototypic in graphs of resubstitution error, here $err(\hat{X}_{df})$, versus model complexity, here *df*. The logic as to why this change point constitutes a basis for model complexity determination is described in Breiman et al. (1984) in terms of bias-variance tradeoff. Elbow identification is also used for determining appropriate numbers of principal components (Jolliffe, 2002) and clusters (Hastie et al., 2009), as well as dimension in MDS (Kruskal and Wish, 1978) and non-negative matrix factorization (NMF; see Hutchins et al., 2008) problems.

The corresponding plot of approximation error against degrees-of-freedom is presented in Figure 1.3. Use of segmented regression (performed using the R package `segmented`; Muggeo, 2008) to



(a) $df = 10$ (b) $df = 15$ (c) $df = 25$



(d) $df = 50$ (e) $df = 100$ (f) $df = 200$

Figure 1.2: $\hat{X}_{df}$, the projections of the resulting reconstruction, with colors (orange, teal) distinguishing chromosome arms, and $S(\hat{X}_{df})$, the approximation of Z, obtained via *PCMS* for different degrees-of-freedom values df.

identify reasonable model size, given by the elbow, suggests a value $df = 25$ as shown.

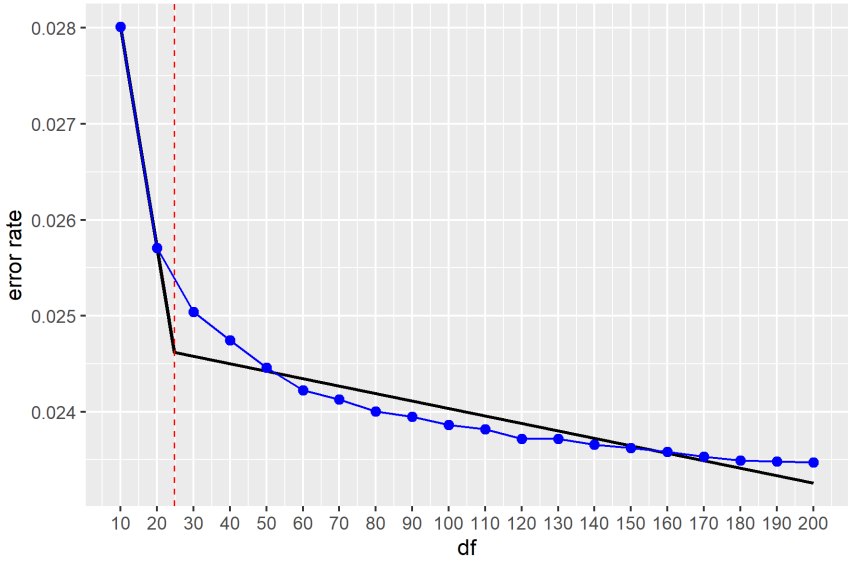


Figure 1.3: Error rate $err(\hat{X}_{df})$ vs. degrees-of-freedom df plot for the *PCMS* approach. The segmented regression is given by the piecewise linear fit (black) with the degrees-of-freedom selected via kink estimation indicated by the red vertical line and segmentation change point corresponding to $df = 25$.

1.4 Weighted principal curve metric scaling

1.4.1 A distance-based model for chromatin reconstruction

As indicated, direct application of *PCMS* to Hi-C data is limited by the need to convert contact counts to distances and then (via double centering) to inner products since such conversion can be problematic. Even simplistic approaches, based on power law transformation, prescribe a value for the index parameter, failing to accommodate dependence of the index on influencing factors such as cell type, chromosome, organism and resolution. Moreover, the double centering trick requires that resultant distances be Euclidean.

Accordingly, we develop a distance-based version of *PCMS*, wherein the symmetric matrix Z contains pairwise squared distances, as opposed to inner products. Additional flexibility is facilitated by introducing weights to the problem setup, which permits control over the impact of particular elements Z_{ij} on the reconstruction, for example to counteract diagonal dominance (Yang et al., 2017).

We introduce a matrix of weights $W \in [0,1]^{n\times n}$, denote by $D(X)$ the matrix of pairwise distances

between genomic loci and consider the following loss function

$$\ell(x_1,\ldots,x_n)=\sum_{i=1}^{n}\sum_{j=1}^{n}W_{ij}\left(Z_{ij}-\|x_i-x_j\|^2\right)^2 \quad\Longleftrightarrow\quad \ell(X)=\left\|\sqrt{W}*\left(Z-D^2(X)\right)\right\|_F^2 \tag{1.8}$$

where $*$ refers to the Hadamard (element-wise) product and matrix squaring is also element-wise. The *WPCMS* problem can be stated as follows:

$$\underset{\Theta\in\mathbb{R}^{k\times 3}}{\text{minimize}}\,\ell_{WPCMS}(\Theta)=\left\|\sqrt{W}*\left(Z-D^2(H\Theta)\right)\right\|_F^2. \tag{1.9}$$

The corresponding solution and reconstruction are denoted by $\hat{\Theta}=PCMS_W(Z,H)$ and $\hat{X}=H\hat{\Theta}$, respectively, along with the corresponding approximation $\hat{Z}=D^2(\hat{X})$ of matrix Z.

1.4.2 Projected gradient descent and the *WPCMS* algorithm

Problem (1.9) can be elegantly solved using *projected gradient descent* (PGD) (Hastie et al., 2015), broadly used to solve constrained optimization problems. We first exploit the fact that the matrix of squared distances can be rewritten in terms of the inner product matrix:

$$D^2(X)=\operatorname{diag}(S(X))\cdot\mathbf{1}^T+\mathbf{1}\cdot\operatorname{diag}(S(X))^T-2S(X). \tag{1.10}$$

Here $\mathbf{1}=(1,1,\ldots,1)^T\in\mathbb{R}^n$ and $\operatorname{diag}(S(X))=\left(\|x_1\|^2,\|x_2\|^2,\ldots,\|x_n\|^2\right)^T$ is the diagonal of the inner product matrix. So (1.9) can be restated in terms of the inner products:

$$\underset{S\in\mathcal{M}(H)}{\text{minimize}}\,\ell_{WPCMS}(S)=\left\|\sqrt{W}*\left(Z-\operatorname{diag}(S)\mathbf{1}^T-\mathbf{1}\operatorname{diag}(S)^T+2S\right)\right\|_F^2 \tag{1.11}$$

where $\mathcal{M}(H)=\left\{H\Theta\Theta^TH^T:\Theta\in\mathbb{R}^{k\times 3}\right\}$. The PGD procedure alternates the following two steps:

$$\textbf{[Gradient]}\;\; S:=S-\nabla\ell_{WPCMS}(S) \quad\text{and}\quad \textbf{[Projection]}\;\; S:=\operatorname{proj}_{\mathcal{M}(H)}(S).$$

Here $\operatorname{proj}_{\mathcal{M}(H)}(S)$ denotes the projection of matrix S onto the matrix manifold $\mathcal{M}(H)$. The **[Gradient]** step makes recourse to the following Lemma.

Lemma 2. *Let D^2 denote the matrix of squared distances corresponding to inner product matrix S (as in 1.10). For a given square matrix A denote by $\Phi(A)$ the operator that subtracts the row sums of A from its diagonal, i.e. $\Phi(A)=A-\operatorname{diag}(A\cdot\mathbf{1})$. If $G=W*\left(Z-D^2\right)$, then up to a scaling factor $\nabla\ell_{WPCMS}(S)=\Phi(G)$.*

Proof. Note that

$$\ell_{WPCMS}(S) = \left\| \sqrt{W} * \left(Z - \operatorname{diag}(S)\mathbf{1}^T - \mathbf{1}^T \operatorname{diag}(S) + 2S \right) \right\|_F^2 = \\ = \sum_{1 \le i,j \le n} W_{ij}(Z_{ij} - S_{ii} - S_{jj} + 2S_{ij})^2.$$

So the partial derivatives are

$$\frac{\partial \ell_{WPCMS}(S)}{\partial S_{ij}} = 4\ W_{ij}(Z_{ij} - S_{ii} - S_{jj} + 2S_{ij}) = 4\ W_{ij}(Z_{ij} - D_{ij}^2) = 4\ G_{ij} = 4\ \Phi(G)_{ij}$$

$$\frac{\partial \ell_{WPCMS}(S)}{\partial S_{ii}} = -4 \sum_{1 \le j \le n, j \ne i} W_{ij}(Z_{ij} - S_{ii} - S_{jj} + 2S_{ij}) = -4 \sum_{1 \le j \le n, j \ne i} W_{ij}(Z_{ij} - D_{ij}^2) = \\ = -4 \sum_{1 \le j \le n, j \ne i} G_{ij} = 4 \left(G_{ii} - \sum_{1 \le j \le n} G_{ij} \right) = 4\ \Phi(G)_{ij}.$$

In matrix form this is exactly $\nabla \ell_{WPCMS}(S) = \Phi(G)$. □

Next, note that the **[Projection]** step requires solving the optimization problem

$$\underset{\Theta \in \mathbb{R}^{k \times 3}}{\text{minimize}} \, \|S - H\Theta\Theta^T H^T\|_F^2,$$

which is easily done using *PCMS*. Thus, we end up with the following PGD procedure:

Weighted principal curve metric scaling (*WPCMS*)

---

1. **[Initialize]** Generate random $\Theta \in \mathbb{R}^{k \times 3}$, set the reconstruction $X = H\Theta$.
2. *Repeat until convergence*:
   - 2.1 **[SDG]** Calculate the current guess for the inner product matrix $S = XX^T$ and use it to compute the matrix of squared distances $D^2 = \operatorname{diag}(S) \cdot \mathbf{1}^T + \mathbf{1} \cdot \operatorname{diag}(S)^T - 2S$. Then compute $G = W * \left(Z - D^2 \right)$.
   - 2.2 **[Gradient]** Update matrix of inner products $S := S - \Phi(G)$.
   - 2.3 **[Projection]** Update spline coefficients using *PCMS*, i.e. set $\Theta := PCMS(S, H)$, then update the reconstruction $X = H\Theta$.

The algorithm convergence is assessed via the stopping criterion $\left| \frac{\ell_{WPCMS}(\Theta_{old}) - \ell_{WPCMS}(\Theta_{new})}{\ell_{WPCMS}(\Theta_{old})} \right| < \epsilon_1$, where ϵ_1 is a pre-chosen accuracy rate, and Θ_{old} and Θ_{new} are Θ values calculated at the previous and current iterations respectively. Details and extensions of *WPCMS* are provided in the Appendix.

1.4.3 Chromatin reconstruction via *WPCMS*

Similarly to *PCMS*, we can use *WPCMS* as a standalone chromatin reconstruction technique. In these experiments we seek for the approximation

$$\log(C) \approx -D(X)^2 + \beta,$$

which is consisted with the *PoisMS* model that we will develop in the next section. In Figure 1.4 we present the heatmap for the target matrix $\log(C)$.

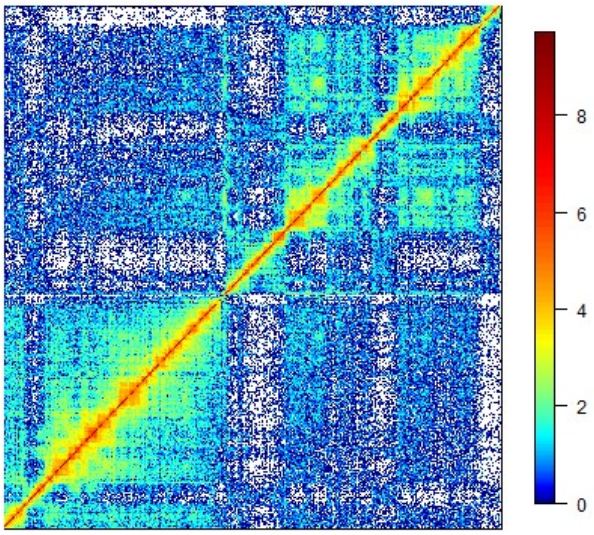


Figure 1.4: Log-transformed contact matrix $\log(C)$. White color corresponds to $C_{ij} = 0$ or, equivalently, $\log(C_{ij}) = -\infty$.

To state the *WPCMS* goal we first introduce the binary matrix of weights W with elements

$$W_{ij} = \begin{cases} 0, \text{ if } C_{ij} = 0, \\ 1, \text{ if } C_{ij} \neq 0. \end{cases} \tag{1.12}$$

This matrix will allow the algorithm to ignore the $\log(C_{ij})$ elements when $C_{ij} = 0$. The resulting weighted optimization problem

$$\underset{\Theta \in \mathbb{R}^{k\times 3},\ \beta\in\mathbb{R}}{\text{minimize}} \left\| \sqrt{W} * \left(-\log(C) - D^2(H\Theta) + \beta \right) \right\|_F^2 \text{ w.r.t.} \tag{1.13}$$

therefore, coincides with the *WPCMS* problem (1.22) with $Z = -\log(C)$ and binary W.

In Figure 1.5, we present the chromatin reconstructions $\hat{X} = H\hat{\Theta}$ as well as the approximations

$-D^2(\hat{X}) + \hat{\beta}$ of the log-transformed contact matrix $\log(C)$ for a series of degrees-of-freedom. The corresponding plot of approximation error

$$err(\hat{X}_{df}, \hat{\beta}_{df}) = \frac{1}{n^2} \left\| \sqrt{W} * (Z - D^2(\hat{X}_{df}) + \hat{\beta}_{df}) \right\|_F^2 \tag{1.14}$$

is presented in Figure 1.6. Use of segmented regression to identify reasonable model size, given by the elbow, suggests a value $df = 35$ as shown.

1.5 Poisson metric scaling

1.5.1 Poisson model for contact counts

We now develop our primary approach, Poisson metric scaling (*PoisMS*), using *WPCMS* as a building block. We define a probabilistic model for contact counts based on natural and previously adopted assumptions: Poisson distributed counts C_{ij} with dependence of the Poisson mean on chromatin 3D structure, specifically on pairwise (squared) distances between genomic loci:

$$C_{ij} \sim \text{Pois}(\lambda_{ij}), \quad \log(\lambda_{ij}) = -\|x_i - x_j\|^2 + \beta, \tag{1.15}$$

with $\beta \in \mathbb{R}$ an intercept parameter. The negative log-likelihood objective is

$$\ell_{PoisMS}(X, \beta) = \sum_{i=1}^{n} \sum_{j=1}^{n} \left[e^{-\|x_i - x_j\|^2 + \beta} - C_{ij} \left(-\|x_i - x_j\|^2 + \beta \right) \right] \tag{1.16}$$

and the MLE optimization problem under the smooth curve constraint (1.4) is

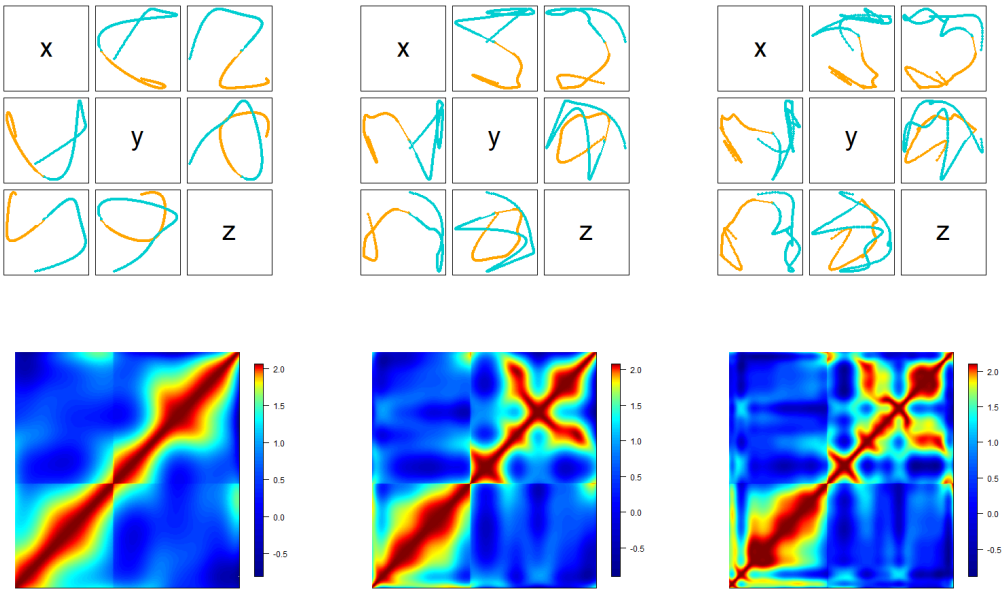
$$\underset{\Theta \in \mathbb{R}^{k \times 3},\ \beta \in \mathbb{R}}{\text{minimize}} \ \ell_{PoisMS}(H\Theta, \beta). \tag{1.17}$$

In (1.15) we use squared distance rather than distance, reflecting criterion (1.1) and conferring computational convenience. We denote the corresponding matrix of spline coefficients by $\hat{\Theta} = \mathit{PoisMS}(C, H)$ and the resulting chromatin reconstruction by $\hat{X} = H\hat{\Theta}$.

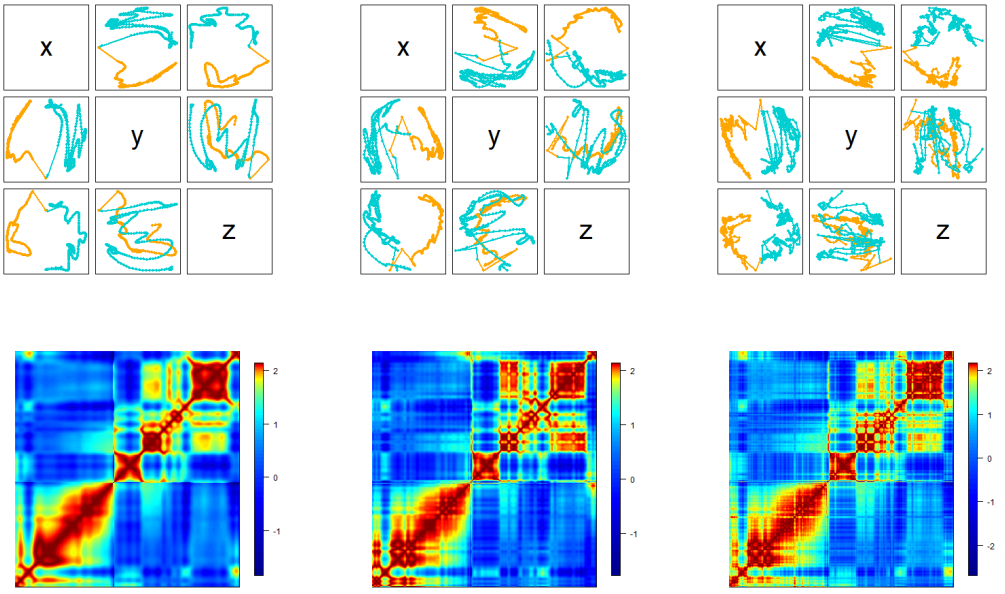
1.5.2 Newton's method and the *PoisMS* algorithm

A virtue of the Poisson model is that the second order Taylor approximation (SOA) of the negative log-likelihood (1.16) is simply the weighted Frobenius norm. Further, it is well known that the optimal value of this SOA amounts to one step of the Newton's method for optimizing the original loss function. We use these facts to develop an iterative algorithm based on the *WPCMS* technique, which is equivalent to a projected Newton's method.

First, we review the SOA of the negative Poisson log-likelihood in the univariate case. Suppose



(a) $df = 10$ (b) $df = 15$ (c) $df = 25$



(d) $df = 50$ (e) $df = 100$ (f) $df = 200$

Figure 1.5: $\hat{X}_{df}$, the projections of the resulting reconstruction, with colors (orange, teal) distinguishing chromosome arms, and $-D^2(\hat{X}_{df}) + \hat{\beta}_{df}$, the approximation of $\log(C)$, obtained via *WPCMS* for different degrees-of-freedom values df.

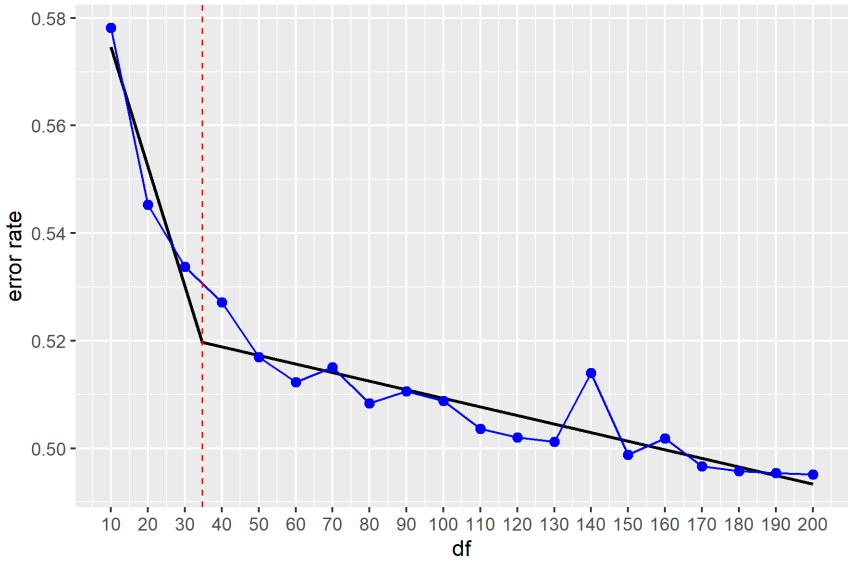


Figure 1.6: Error rate $err(\hat{X}_{df})$ vs. degrees-of-freedom df plot for the *WPCMS* approach. The segmented regression is given by the piecewise linear fit (black) with the degrees-of-freedom selected via kink estimation indicated by the red vertical line and segmentation change point corresponding to $df = 35$.

$c \sim \text{Pois}(\lambda)$. The negative log likelihood $\ell(\lambda) = \lambda - c\log\lambda$ can be reparametrized in terms of the natural parameter $\eta = \log(\lambda)$ leading to $\ell(\eta) = e^{\eta} - c\eta$. Then the SOA of the reparametrized negative log-likelihood at some point $\eta_0 = \log\lambda_0$, up to scaling and shifting by a constant, is:

$$\ell(\eta) \approx \ell_{SOA}(\eta) = w(z-\eta)^2 \quad \text{where} \quad w = e^{\eta_0} = \lambda_0 \quad \text{and} \quad z = \eta_0 + \frac{c-\lambda_0}{\lambda_0}.$$

The multivariate version is as follows. Suppose $C \in \mathbb{Z}_+^{n\times n}$ where $C_{ij} \sim \text{Pois}(\lambda_{ij})$ and $\eta_{ij} = \log(\lambda_{ij})$. Let the respective matrices of Poisson and natural parameters be $\Lambda = [\lambda_{ij}] \in \mathbb{R}_+^{n\times n}$ and $\mathcal{H} = [\eta_{ij}] \in \mathbb{R}^{n\times n}$. Then the SOA of the negative log-likelihood at some point $\mathcal{H}_0$, up to scale and shift constants, is

$$\ell(\mathcal{H}) \approx \ell_{SOA}(\mathcal{H}) = \|\sqrt{W} * (Z-\mathcal{H})\|_F^2 \quad \text{where} \quad W = e^{\mathcal{H}_0} = \Lambda_0 \quad \text{and} \quad Z = \mathcal{H}_0 + \frac{C-\Lambda_0}{\Lambda_0}.$$

Here $*$ is the Hadamard (element-wise) product, with matrix exponentiation and division also being interpreted as element-wise operations.

Recall that in the Poisson model (1.15) the natural parameter depends linearly on the matrix of genomic loci pairwise distances: $\mathcal{H} = \log\Lambda = -D^2(X) + \beta$. So, the SOA can be rewritten as

$$\ell_{SOA}(X) = \left\|\sqrt{W} * \left(Z + D^2(X) - \beta\right)\right\|_F^2 = \left\|\sqrt{W} * \left(\tilde{Z} - D^2(X)\right)\right\|_F^2 \quad \text{for} \quad \tilde{Z} = -Z + \beta.$$

Suppose that the current reconstruction guess is X_0 with corresponding natural parameter value $\mathcal{H}_0 = -D^2(X_0) + \beta$. Then we have the following approximation of the Poisson loss (1.16) at point $\mathcal{H}_0$ again up to scaling and shifting by a constant:

$$\ell_{PoisMS}(X, \beta) \approx \ell_{SOA}(X) = \left\| \sqrt{W} * \left(Z - D^2(X) \right) \right\|_F^2$$
$$\text{where } W = e^{-D^2(X_0)+\beta} \text{ and } Z = D^2(X_0) - \tfrac{C-W}{W}.$$

Thus, under the smooth curve constraint $X = H\Theta$, the loss function $\ell_{SOA}(X)$ coincides with the *WPCMS* loss (1.9) and we obtain a nice application of the *WPCMS* algorithm, with the solution to the second order approximation of problem (1.17):

$$\underset{\Theta \in \mathbb{R}^{k\times 3}}{\text{minimize}}\, \ell_{SOA}(\Theta) = \left\| \sqrt{W} * \left(Z - D^2(H\Theta) \right) \right\|_F^2 \tag{1.18}$$

being exactly $\Theta = PCMS_W(Z, H)$. This observation can be applied to simplify computations for the Poisson model and underlies our *PoisMS* algorithm.

The last step of our *PoisMS* algorithm is to update β according to the current guess of Θ. This can be done by optimizing the negative log-likelihood with respect to β. All together this leads to the following algorithm that repeatedly approximates the Poisson objective at current guess Θ by a quadratic function and shifts Θ towards the global minimum of this quadratic approximation:

Poisson metric scaling (*PoisMS*)

---

1. **[Initialize]** Generate random $\Theta \in \mathbb{R}^{k\times 3}$, set the reconstruction $X = H\Theta$.

2. *Repeat until convergence*:

   2.1 **[Update β]** Update the intercept $\beta := \log\left(\frac{\sum_{1\le i,j\le n} C_{ij}}{\sum_{1\le i,j\le n} e^{-\|x_i - x_j\|^2}} \right)$.

   2.2 **[SOA]** Calculate matrices $W = e^{-D^2(X)+\beta}$ and $Z = D^2(X) - \frac{C-W}{W}$.

   2.3 **[WPCMS]** Update the spline coefficients using *WPCMS* approach $\Theta := PCMS_W(Z, H)$, then update the reconstruction $X = H\Theta$.

The stopping rule for the *PoisMS* algorithm is similar to *WPCMS*: for some fixed accuracy rate ϵ_2 we check if the updated $(\Theta_{new}, \beta_{new})$ meets the criteria $\left| \frac{\ell_{PoisMS}(\Theta_{old},\beta_{old}) - \ell_{PoisMS}(\Theta_{new},\beta_{new})}{\ell_{PoisMS}(\Theta_{old},\beta_{old})} \right| < \epsilon_2$ after each iteration of steps 2.1–2.3.

The non-convexity of the *PoisMS* criteria (1.17) implies that initialization can impact the resulting reconstruction. In the Appendix we discuss use of *WPCMS* to provide a warm start for the *PoisMS* algorithm, as well as algorithmic extensions.

1.5.3 Computational complexity

The computational complexity of *PCMS* is $O(n^2k)$ operations for computing the product H^TCH and extra $O(k^3)$ for eigen-decomposition (negligible for small k). Each iteration of *WPCMS* has the same complexity as *PCMS* plus $O(n^2)$ operations for computing Hadamard products with matrix W. Finally, the complexity of a *PoisMS* iteration is equivalent to a *WPCMS* iteration and requires an additional $O(n^2)$ operations for calculating the second order approximation.

1.5.4 Chromatin reconstruction via *PoisMS*

We present the *PoisMS* reconstructions for IMR90 cell chromosome 20 at 100kb resolution for which multiplex FISH and Hi-C data acquisition and processing has been previously described (Segal and Bengtsson, 2018). Results for chromosome 21 are presented in the Appendix.

In Figure 1.4 we presented the heatmap for $\log(C)$. The resulting *PoisMS* reconstructions $\hat{X}_{df}$ along with the Poisson parameter matrix $\log(\Lambda_{df}) = -D^2(\hat{X}_{df}) + \beta$, that can be viewed as an approximation of $\log(C)$, are presented in Figure 1.7 for a series of degrees-of-freedom values.

The graph of error rate

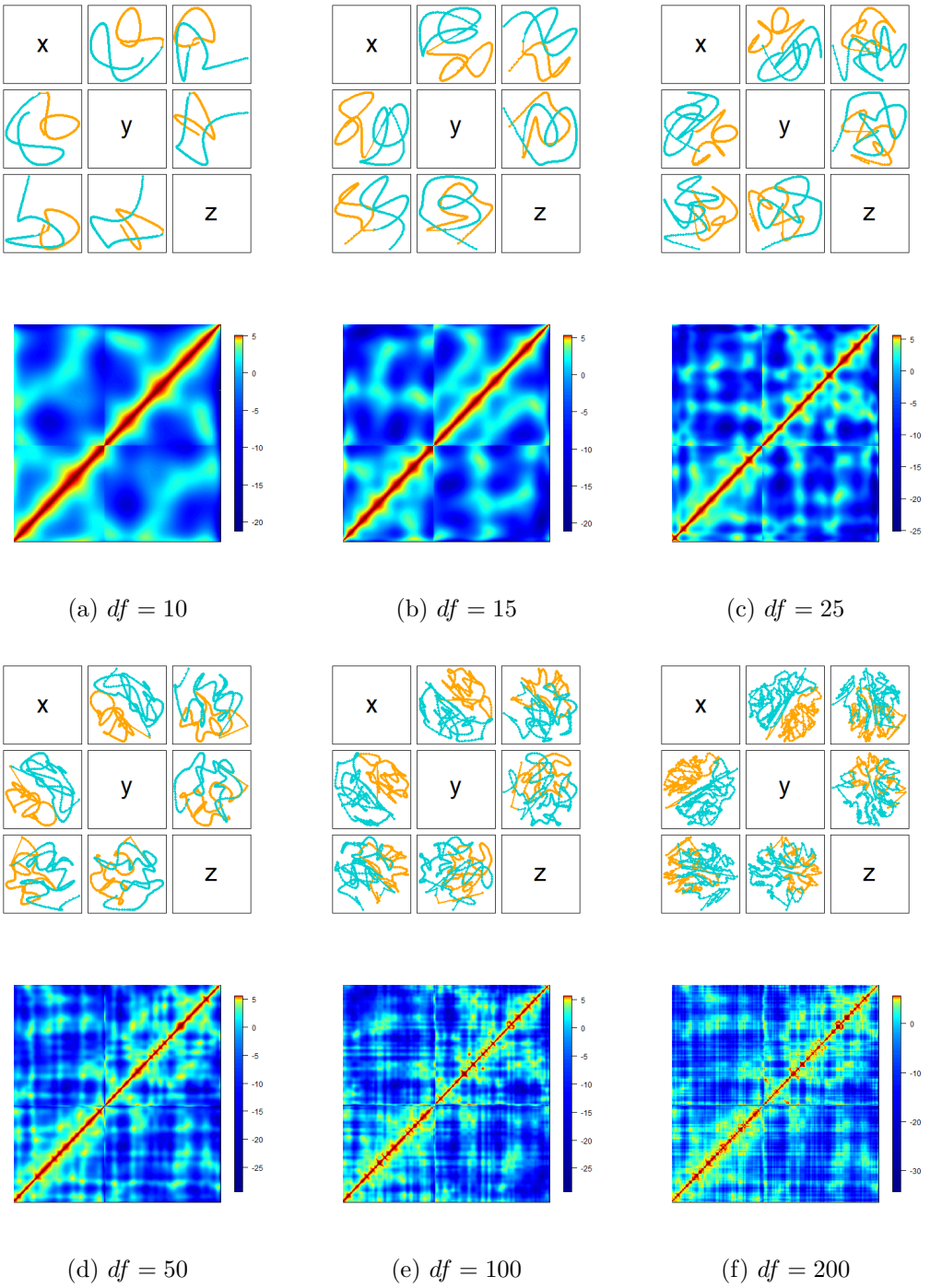
$$err(\hat{X}_{df}, \hat{\beta}_{df}) = \frac{2}{n^2} \sum_{1 \le i,j \le n} \left[C_{ij} \log \frac{C_{ij}}{\lambda_{ij}} - (C_{ij} - \lambda_{ij}) \right] \text{ with } \lambda_{ij} = -D^2(\hat{X}_{df}) + \hat{\beta}_{df}. \tag{1.19}$$

vs *df* reveals rapidly decreasing error rates up to $df = 30$ with subsequent gradual decline (Figure 1.8). The optimal *df* according to the elbow heuristic is $df = 25$, also shown in Figure 1.8.

1.6 Assessment of the reconstruction accuracy

1.6.1 Multiplex FISH

While the prescription in Section 1.3 provides a means for selecting a particular *PoisMS* model it does not address the accuracy of the chosen model. The absence of gold standards makes such assessment challenging. In comparing competing 3D genome reconstructions several authors have appealed to simulation (Zhang et al., 2013; Varoquaux et al., 2014; Zou et al., 2016; Park and Lin, 2017), however, real data referents are preferable. To that end, many of the same reconstruction algorithm developers have made recourse to fluorescence in situ hybridization (FISH) imaging as a basis for gauging accuracy. This proceeds by comparing distances between imaged probes with corresponding reconstruction-based distances. But such methods are necessarily limited by the sparse number of probes ($\sim 2-6$; see Lieberman-Aiden et al., 2009; Shavit et al., 2014; Park and Lin, 2017) and the modest resolution thereof, many straddling over 1 megabase. The recent advent of *multiplex* FISH (Wang et al., 2016) transforms 3D genome reconstruction accuracy evaluation



(a) $df = 10$ (b) $df = 15$ (c) $df = 25$

(d) $df = 50$ (e) $df = 100$ (f) $df = 200$

Figure 1.7: $\hat{X}_{df}$, the projections of the resulting reconstruction, with colors (orange, teal) distinguishing chromosome arms, and $-D^2(\hat{X}_{df}) + \beta$, the approximation of $\log(C)$, obtained via *PoisMS* for differing degrees-of-freedom values df.

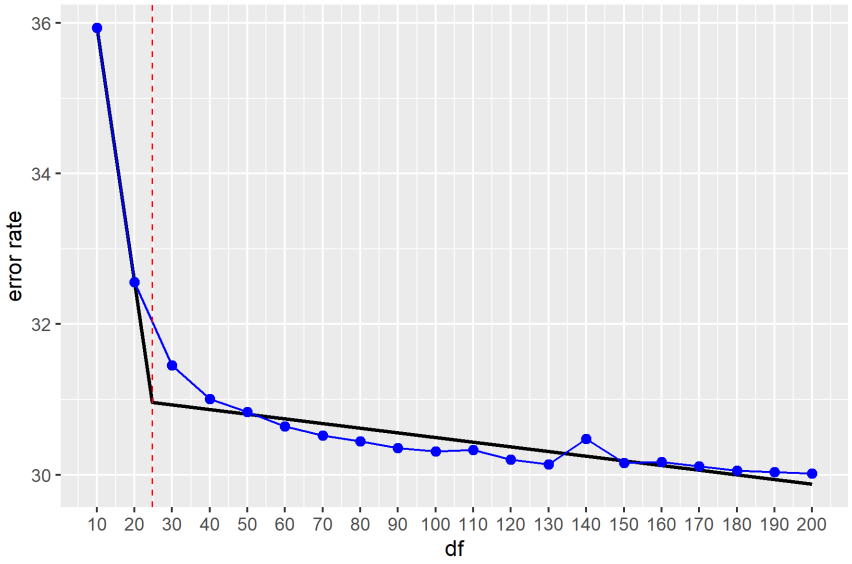


Figure 1.8: Error rate $err(\hat{X}_{df}, \hat{\beta}_{df})$ vs. degrees-of-freedom df plot for the *PoisMS* approach. The segmented regression is given by the piecewise linear fit (black) with the degrees-of-freedom selected via kink estimation indicated by the red vertical line and segmentation change point corresponding to $df = 25$.

by providing an order of magnitude more probes and hence two orders of magnitude more inter-probe distances than conventional FISH. Moreover, the probes are at higher resolution and centered at topologically associated domains (TADs; see Dixon et al., 2012). We use this imaging data, along with companion accuracy assessment approaches (Segal and Bengtsson, 2018) to evaluate our *PoisMS* reconstructions.

The image-based 3D genomic coordinates furnished from multiplex FISH serve to define the gold standard by which we assess reconstructions. The existence of numerous multiplex FISH replicates is crucial for this task and three steps are necessary to effect such evaluation.

Obtaining the gold standard Given N multiplex FISH replicates denote the matrix of the spatial coordinates for replicate $i \in \{1, \ldots, N\}$ by $M_i \in \mathbb{R}^{n_0 \times 3}$ where n_0 denotes the number of distinct multiplex FISH loci (probes) over all replicates. We start by defining the *medoid replicate*. For a pair of 3D conformations $X_1, X_2 \in \mathbf{R}^{n_0 \times 3}$ denote the number of observed loci by $n(X_1, X_2)$ and suppose $d_{proc}(X_1, X_2)$ is the squared Procrustes distance from X_2 to X_1 following alignment allowing translation, rotation and scaling (Hastie et al., 2009). Then the dissimilarity between X_1 and X_2 is defined by

$$d(X_1, X_2) = \frac{1}{n(X_1, X_2)} d_{proc}(X_1, X_2), \tag{1.20}$$

using asymmetric (scaling and rotation transforms applied to X_2 only) Procrustes distance. This measure of agreement between two reconstructions coincides with mean squared deviation (Segal and Bengtsson, 2018).

We next define the *medoid* replicate as the replicate whose (weighted) average dissimilarity to the other replicates is minimal:

$$j^* = \text{argmin}_{j=1,\ldots,N} \sum_{i=1}^{N} \frac{d(M_i, M_j)}{\sum_{k=1}^{N} d(M_i, M_k)}, \tag{1.21}$$

with weights $\frac{1}{\sum_{k=1}^{N} d(M_i, M_k)}$ chosen to adjust for different scales of the multiplex FISH replicates. Next, let M_i^{rot} be the Procrustes alignment of M_i to the medoid M_{j^*}. The *average Procrustes conformation* $\bar{M}$, defined as the locus-wise average of the M_i^{rot}, then serves as a gold standard. Our application of Procrustes alignment prior to this (noise reducing) averaging accommodates translation, rotation, and scaling differences between replicate conformations.

Computing the reference distribution Treating the average Procrustes conformation $\bar{M}$ as our gold standard we obtain a reference distribution by measuring the dissimilarity between it and the multiplex FISH replicates: $d(\bar{M}, M_i)$. The resulting empirical distribution captures experimental variation around the gold standard. A fine point is that this distribution will exhibit reduced dispersion compared to its target population quantity owing to data re-use since M_i contributes to $\bar{M}$. While this concern could be mitigated by employing leave-one-out techniques the large number of available replicates (> 110) renders this unnecessary (Segal and Bengtsson, 2018).

Evaluating chromatin reconstructions To evaluate reconstructions resulting from the *PoisMS* approach we first need to align the reconstruction with the gold standard. This may involve preliminary coarsening of one or other coordinate sets to yield comparable resolution. Here, the genomic coordinate ranges for each multiplex FISH probe are coarser than the Hi-C bins used in our reconstructions. So we calculate the average of the reconstruction coordinates falling in the corresponding multiplex FISH bins to obtain a lower resolution reconstruction $\hat{X}$ of the same dimension as $\bar{M}$. To quantify how close this reconstruction is to the gold standard $\bar{M}$ we again measure dissimilarity following alignment $d(\bar{M}, \hat{X})$

1.6.2 A contrasting reconstruction algorithm: HSA

To compare our *PoisMS* solution with an alternate reconstruction algorithm we make recourse to HSA (Zou et al., 2016). This technique provides an interesting contrast in that it employs a similar Poisson formulation to (1.16) but instead of contiguity being captured via principal curves per (1.4), it is indirectly imparted by constraints that induce dependencies on a hidden Gaussian Markov

chain over the solution coordinates. Obtaining these spatial coordinates is achieved via simulated annealing with further smoothness effected via distance-based penalization.

HSA has performed well in some benchmarking studies and features several compelling attributes including (i) simultaneously handling multiple data tracks allowing for integration of replicate contact maps and (ii) adaptively estimating the power-law index whereby contacts are transformed to distances as previously emphasized. Nonetheless, in contrast to *PoisMS*, HSA incurs a substantial compute and memory burden, and questions surrounding robustness have been raised (Rieber and Mahony, 2017).

To compare *PoisMS* performance with HSA we use the approach described in Section 1.6. Having obtained a HSA reconstruction we measure the dissimilarity between the reconstruction and the gold standard.

1.6.3 Validation of the *PoisMS* reconstructions

Procrustes alignment of 3D conformations, and calculation of the corresponding distances $d_{proc}(\cdot, \cdot)$, was performed using the R package `vegan` (Oksanen et al., 2016). We obtain the multiplex FISH medoid conformation based on the smallest row sum (1.21) of the dissimilarity matrix of normalized Procrustes distances (1.20) as described above. The 111 multiplex FISH replicate conformations are then aligned to the medoid as a prelude to calculating the average Procrustes conformation – our gold standard. Figure 1.9 shows the histogram of dissimilarities between multiplex FISH replicates and our derived gold standard that constitutes the reference distribution. We position the *PoisMS* reconstruction dissimilarities therein corresponding to the indicated series of degrees-of-freedom values. HSA reconstruction dissimilarity values are also included.

The following conclusions can be drawn from Figure 1.9. For chromosome 20 (see the Appendix 1.8.3 for chromosome 21), all fits for *PoisMS* lie within the range of the multiplex FISH dissimilarity distribution that reflects experimental variation. The fact that the *PoisMS* dissimilarity values are in the left tail of this distribution indicates the accuracy of the proposed reconstructions, highlighting the utility of the proposed methodology. Further, that larger dissimilarity values pertain for HSA, particularly for chromosome 21, suggests that *PoisMS* performs at least comparably to this well benchmarked alternative. That *PoisMS* wall clock times are minutes rather than days for HSA is notable.

1.7 Discussion

Central to our principal curve based approaches to 3D chromatin reconstruction is that the configuration of an individual chromosome within the nucleus can be treated as a contiguous 1D curve since the diameter of the chromatin fiber is negligible compared to the nuclear volume. The extent to which the curve is “smooth” is determined by an adaptively selected degrees-of-freedom parameter.

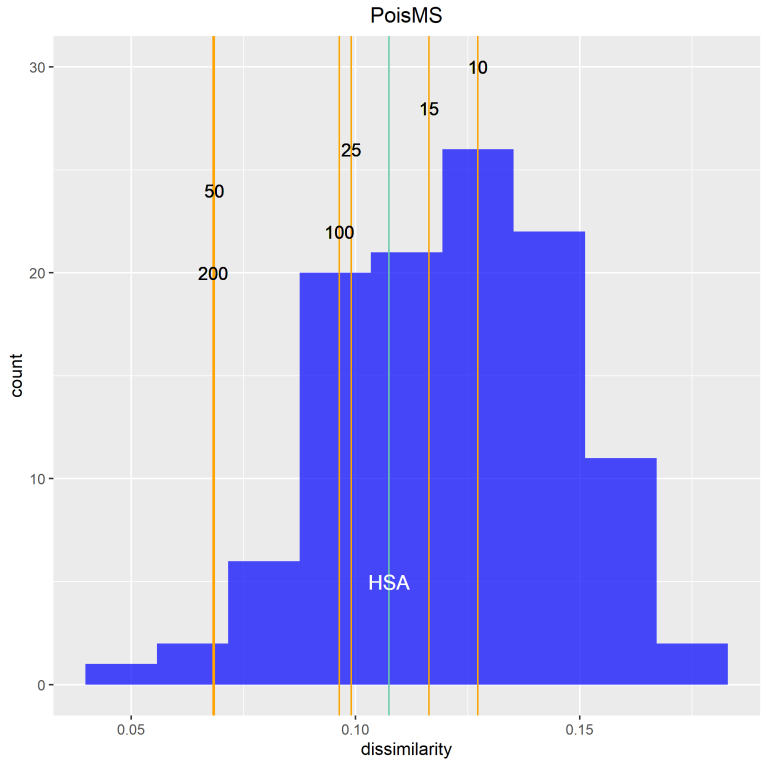


Figure 1.9: Reference distribution measuring the dissimilarity between the gold standard $\bar{M}$ and 111 multiplex FISH replicate conformations M_i for chromosome 20. The vertical orange lines correspond to the dissimilarity between $\bar{M}$ and the low-resolution reconstruction $\hat{X}_{df}$ calculated via *PoisMS* for different *df* values; the light blue line corresponds to the HSA reconstruction.

As mentioned in the introduction, previous reconstruction methods either impart contiguity indirectly by prescribing constraints, which are difficult to specify, or impose it post hoc. In comparison, our methods based on principal curves are computationally efficient, readily scale to high resolution contact data and are parsimonious with regard tuning parameters.

Our implementation of PoisMS utilizes cubic spline basis functions, which contribute to this computational efficiency. However, the nature of chromatin folding and attendant Hi-C data is such that these bases will be less effective in capturing fine 3D structure, as opposed to global backbone architecture. This derives from the hierarchical, domain-based organization of chromatin, aspects that have been tackled by some reconstruction algorithms using strategies that synthesize solutions obtained at differing scales (Rieber and Mahony, 2017; Trieu et al., 2019). We will investigate whether principal curve solutions can similarly serve as building blocks in addition to exploring the use of alternate basis functions, notably wavelets.

Our analyses of Hi-C data from IMR90 cells was motivated by the availability of corresponding multiplex FISH data enabling accuracy assessment. However, the extent and resolution of multiplex

FISH imaging is limited, narrowing the applicability of this means of evaluation. An even more fundamental issue pertains to attempting chromatin reconstruction using *bulk* Hi-C data from large cell populations. As has been emphasized (Lando et al., 2018), the presence of numerous conflicting contacts suggests that the notion of a consensus underlying 3D conformation is questionable and that there is substantial cell-to-cell structural variation. This places a premium on pursuing single cell reconstructions as enabled by the recent emergence of single cell Hi-C protocols (Ramani et al., 2017). That one of these advances (Stevens et al., 2017) also provides parallel imaging data, putatively enabling reconstruction accuracy determination, underscores the importance of applying reconstruction methods in single cell settings, despite contact map sparsity, and is the subject of future work.

1.8 Appendix

1.8.1 The *WPCMS* algorithm extensions

Learning rate Since the *WPCMS* algorithm is based on PGD, many extensions can be naturally derived. For example, a learning rate δ can be introduced by replacing G by δG at the [**Gradient**] step. Further, line search can be readily added to the algorithm:

$$\text{[Gradient] } S := S - \Phi(G) \qquad \Longrightarrow \qquad \begin{aligned} \text{[Search] } S_\delta &:= S - \delta\Phi(G) \\ \Theta_\delta &= PCMS(S_\delta, H) \\ \delta_1 &= \operatorname{argmin}_\delta \ell_{WPCMS}(\Theta_\delta) \\ \text{[Gradient] } S &:= S - \delta_1\Phi(G) \end{aligned}$$

Intercept It is possible to incorporate an intercept parameter into the *WPCMS* problem statement, leading to the optimization problem:

$$\underset{\Theta\in\mathbb{R}^{k\times 3},\ \beta\in\mathbb{R}}{\text{minimize}}\ \ell_{WPCMS}(\Theta,\beta) = \left\|\sqrt{W} * \left(Z - D^2(H\Theta) + \beta\right)\right\|_F^2. \tag{1.22}$$

The *WPCMS* iterative algorithm is then modified by the addition of the following step:

$$\text{[Update } \beta\text{]}\ \text{Update the intercept } \beta := -\frac{\sum_{i=1}^n\sum_{j=1}^n W_{ij}\left(Z_{ij} - \|x_i - x_j\|^2\right)}{\sum_{i=1}^n\sum_{j=1}^n W_{ij}}.$$

Initialization We study the influence of the initialization on the resulting *WPCMS* reconstructions. In our experiments we generated 50 random Θ with $\Theta_{ij} \sim \mathcal{N}(0,1)$. At the [**Initialize**] step for each of these Θ we set the starting value for the reconstruction to $X = H\Theta$, for fixed degrees-of-freedom $df = 50$, and run the *WPCMS* algorithm. We then compare the solutions obtained, as well as losses and β values. In Figure 1.10 we superpose coordinate-wise representations of all

fifty curves obtained via *WPCMS*. The *WPCMS* algorithm converges to two stable conformations. The relative change in the achieved values for both loss and β does not exceed 1%; the number of iterations required for the same convergence criterion varies from tens to a few hundreds.

Figure 1.11 depicts the fifty *WPCMS* solutions in terms of two key geometric characteristics of the reconstructions: the distances between two successive genomic loci, which reflects curvature, and the angles formed by three successive genomic loci, which reflects torsion. The plot showcases highly similar geometric structure across reconstructions, irrespective of initialization.

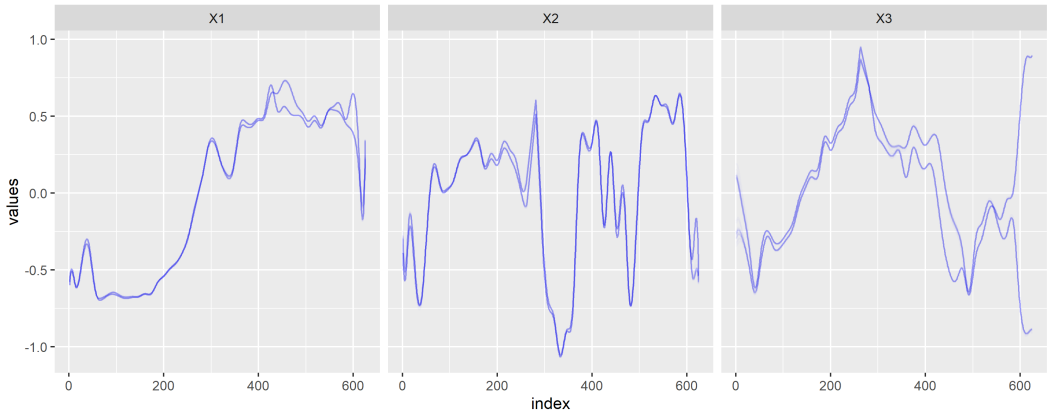


Figure 1.10: Fifty reconstructions $\hat{X}$ computed for $df = 50$ via *WPCMS*: each reconstruction corresponds to a random initialization and is represented coordinate-wise. Two largely similar stable conformations are obtained.

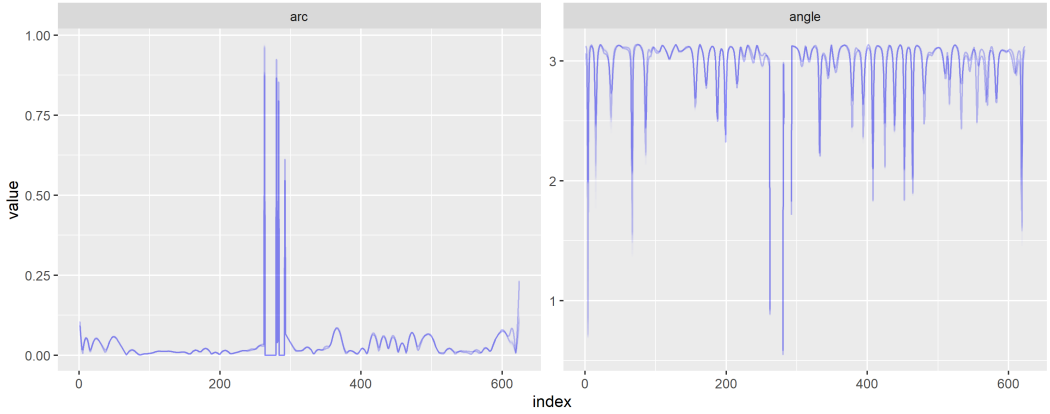


Figure 1.11: 50 reconstructions $\hat{X}$ computed for $df = 50$ via *WPCMS*; each reconstruction corresponds to a random initialization, and is represented as lengths of reconstruction edges and angles between successive edges.

1.8.2 The *PoisMS* algorithm extensions

Iterations It is not necessary to iterate until convergence of the objective $\ell_{SOA}(\Theta)$ at the [**WPCMS**] step. Based on our experiments, the *PoisMS* algorithm can be significantly accelerated by performing only a few steps of *WPCMS* between each update of W and Z. This can be readily accomplished by varying the accuracy rate ϵ_1 for the *WPCMS* stopping rule $\left|\frac{\ell_{SOA}(\Theta_{old})-\ell_{SOA}(\Theta_{new})}{\ell_{SOA}(\Theta_{old})}\right| < \epsilon_1$.

Learning rate Note that the update for the Poisson generalized linear model (GLM) amounts to one step of the Newton method in the space of the natural parameter $\mathcal{H} = -D^2(X) + \beta$. It is not difficult to show that it is equivalent to the Newton step in the parameter space of distances $D^2(X)$ and, consequently, in the parameter space of inner products $S(X)$ since there is a linear dependence of the elements of $D^2(X)$ on the elements of $S(X)$. Therefore, we can add a learning rate as well as line search (in the parameter space of $S(X)$) to the *PoisMS* algorithm as follows:

$$
\begin{array}{rl}
 & \textbf{[WPCMS]}\ \widetilde{\Theta} = PCMS_W(Z, H) \\
 & \textbf{[Search]}\ S_\delta = (1-\delta)\cdot S(H\Theta) + \delta \cdot S(H\widetilde{\Theta}) \\
\textbf{[WPCMS]}\ \Theta := PCMS_W(Z, H) \quad \Longrightarrow & \Theta_\delta = PCMS(S_\delta, H) \\
X := H\Theta & \delta_2 = \operatorname{argmin}_\delta \ell_{PoisMS}(\Theta_\delta) \\
 & \textbf{[Mix]}\ \Theta := \Theta_{\delta_2} \\
 & X = H\Theta
\end{array}
$$

Initialization Akin to our exploration of the impact of initialization on *WPCMS* (Section 1.8.1) here we examine initialization in the context of *PoisMS*. In Figure 1.12 we again superpose 50 curves obtained from different initializations of *PoisMS* and represent them coordinate-wise. As is apparent, *PoisMS* exhibits higher sensitivity to the starting value of Θ than *WPCMS*. However, the relative change in the achieved values for both loss and β is still small, i.e. does not exceed 3%.

While Figure 1.12 highlights reconstruction variability with respect to initialization, Figure 1.13 which, as per Figure 1.11, captures geometry via successive edge lengths and angles, suggests the existence of a common underlying similar structure.

Warm start In view of the sensitivity of *PoisMS* to initialization we pursue use of a warm start. Based on the heuristic from Section 1.8.1, we replace the random initialization step by initializing Θ at the *WPCMS* solution.

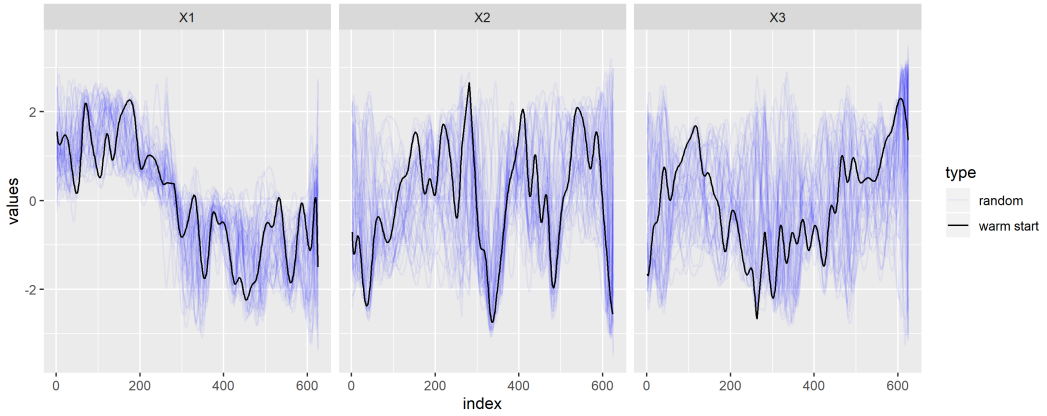


Figure 1.12: 50 reconstructions $\hat{X}$ computed for $df = 50$ via *PoisMS*; each reconstruction corresponds to an initialization (random initialization in blue, warm start in black) and is represented coordinate-wise.

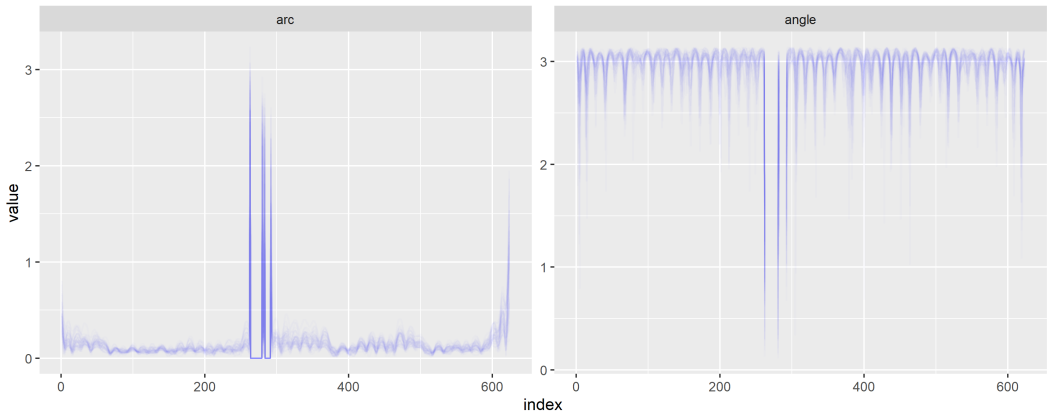


Figure 1.13: 50 reconstructions $\hat{X}$ computed for $df = 50$ via *PoisMS*; each reconstruction corresponds to a random initialization, and is represented as lengths of reconstruction edges and angles between successive edges.

$$\textbf{[Initialize]}\ \text{Generate random } \Theta \in \mathbb{R}^{k\times 3} \implies \textbf{[Warm Start]}\ \Theta := PCMS_W\left(-\log(C), H\right)$$
$$X := H\Theta \qquad\qquad X := H\Theta$$

where W is the binary matrix (1.12). Adoption of warm starts significantly decreases the number of iterations required for convergence and allows for lower loss value. The resulting reconstruction is presented in Figure 1.12 in black.

1.8.3 *PoisMS* reconstructions for IMR90 cell chromosome 21

We provide results for IMR90 chromosome 21 at 100kb resolution that parallel the presentation for chromosome 20.

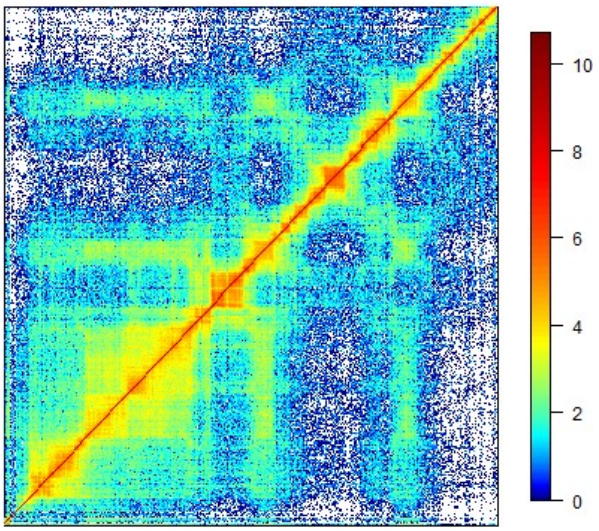
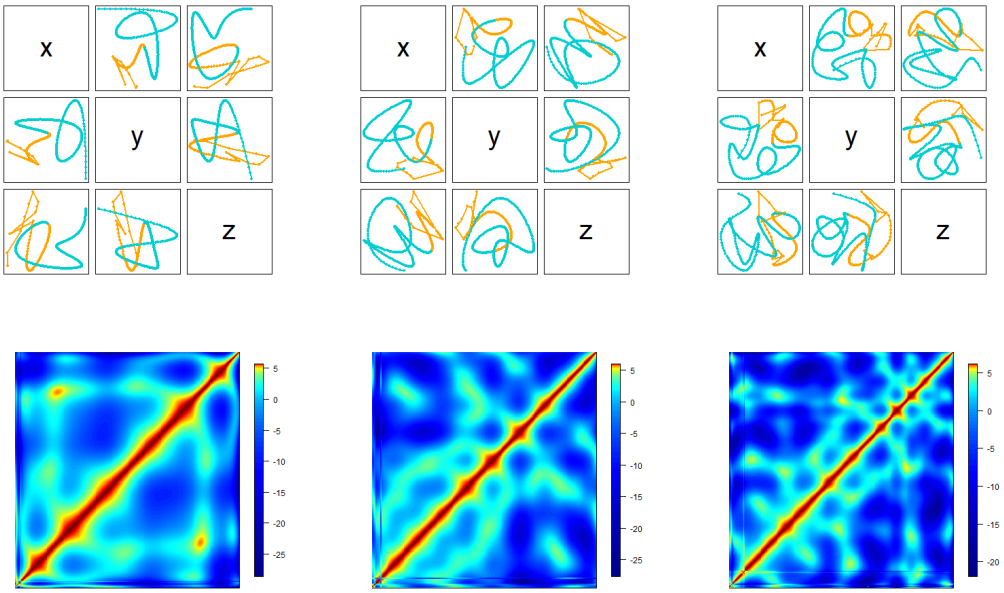
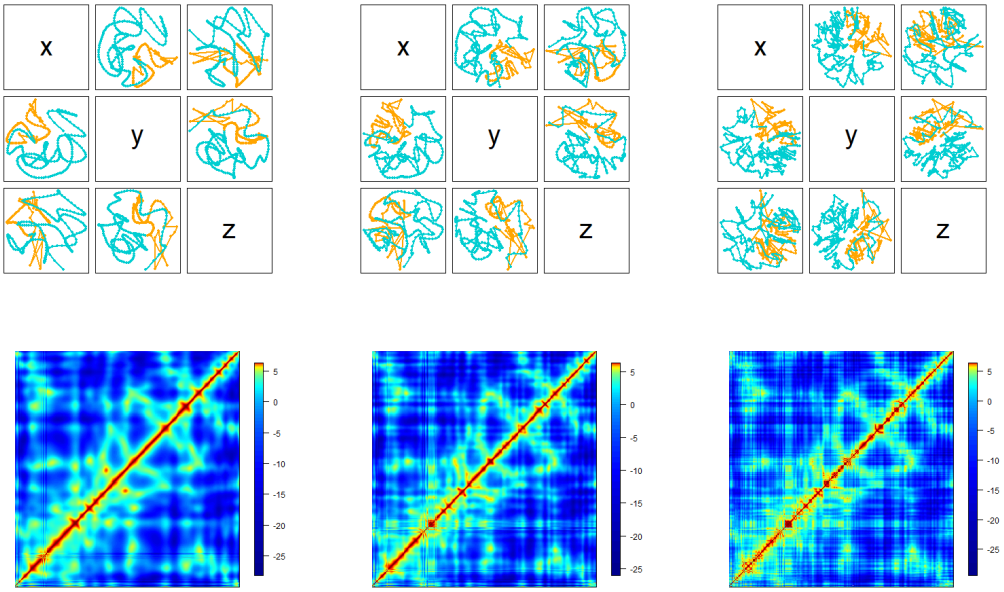


Figure 1.14: Log-transformed contact matrix $\log(C)$. White color corresponds to $C_{ij} = 0$ or, equivalently, $\log(C_{ij}) = -\infty$.



(a) $df = 10$ (b) $df = 15$ (c) $df = 25$



(d) $df = 50$ (e) $df = 100$ (f) $df = 200$

Figure 1.15: $\hat{X}_{df}$, the projections of the resulting reconstruction, with colors (orange, teal) distinguishing chromosome arms, and $-D^2(\hat{X}_{df}) + \beta$, the approximation of $\log(C)$, obtained via *PoisMS* for different degrees-of-freedom values df.

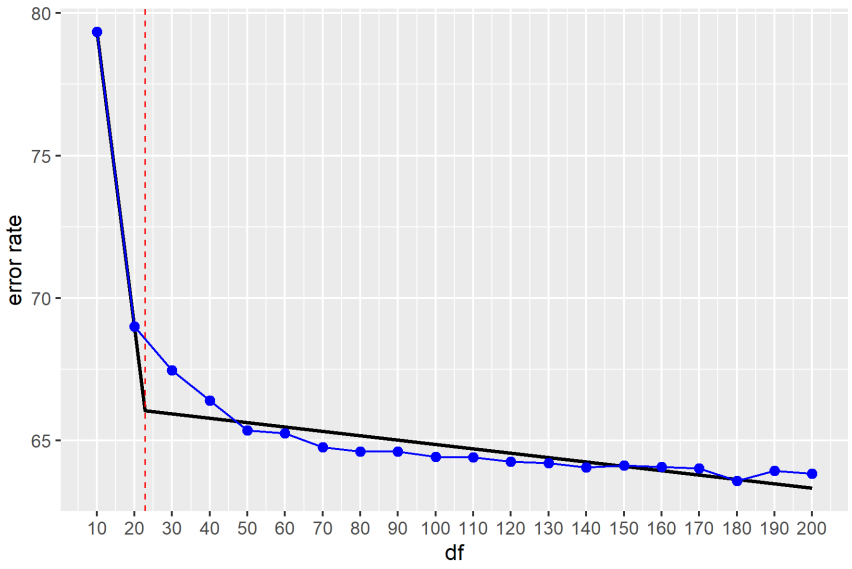


Figure 1.16: Error rate $err(\hat{X}_{df}, \hat{\beta}_{df})$ vs. degrees-of-freedom df plot for the *PoisMS* approach. The segmented regression is given by the piecewise linear fit (black) with the degrees-of-freedom selected via kink estimation indicated by the red vertical line and segmentation change point corresponding to $df = 23$.

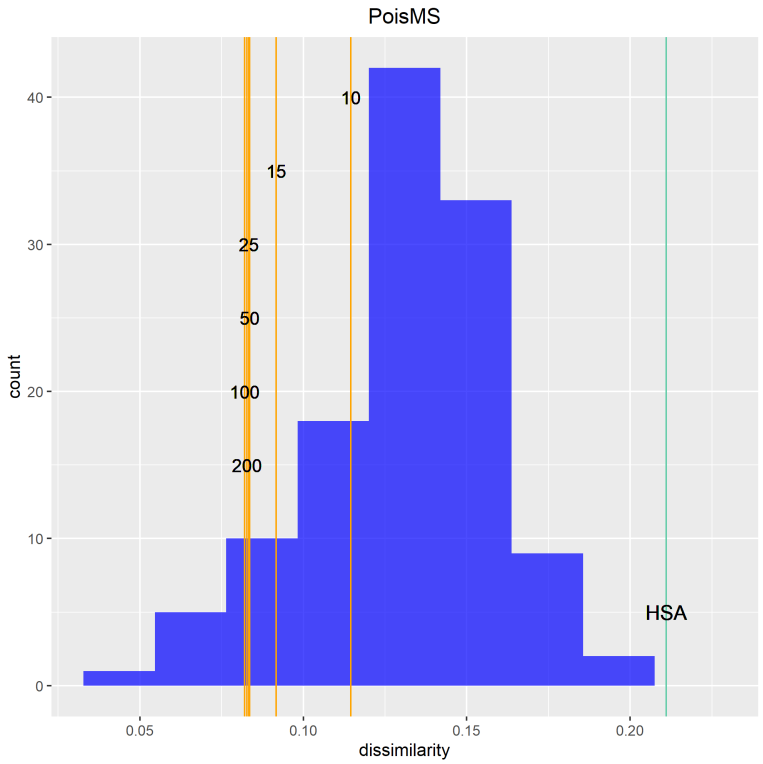


Figure 1.17: Reference distribution measuring the dissimilarity between the gold standard $\bar{M}$ and 111 multiplex FISH replicate conformations M_i for chromosome 21. The vertical orange lines correspond to the dissimilarity between $\bar{M}$ and the low-resolution reconstruction $\hat{X}_{df}$ calculated via *PoisMS* for different *df* values; the light blue line corresponds to the HSA reconstruction.

Chapter 2

Weighted low-rank matrix approximation

2.1 Introduction and background

2.1.1 Low-rank matrix approximation

Low-rank matrix approximation (*LRMA*) has always attracted a lot of attention. It arises in a variety of core machine learning techniques including data compression, dimension reduction and de-noising (Markovsky, 2011), and is central to many multivariate statistical techniques such as principal components, linear discriminant analysis and canonical correlation analysis (Mardia et al., 1979).

The main goal of the *LRMA* technique is to identify the "best" way to approximate some given matrix $M \in \mathbb{R}^{n\times p}$ with a rank-k matrix $X \in \mathbb{R}^{n\times p}$ for $k < \min(n, p)$. There are many ways to define "goodness of fit" between a matrix and its approximation. The most natural one is to use the residual sum of squares $\sum_{i=1}^{n}\sum_{j=1}^{p}(M_{ij} - X_{ij})^2$ leading to the *LRMA* problem:

$$\underset{X\in\mathbb{R}^{n\times p}}{\text{minimize}} \|M - X\|_F^2 \quad \text{subject to } \operatorname{rk}(X) \leq k. \tag{2.1}$$

Here $\|\cdot\|_F$ refers to Frobenius norm. The solution to this problem is well-know and can be found via the singular value decomposition (SVD) of the matrix M (Eckart and Young, 1936).

2.1.2 Missing values

The *LRMA* problem can be extended to the case when only a subset of entries in M is observed. The resulting technique, also known as *low-rank matrix completion* (*LRMC*), aims to recover the missing entries of matrix M from partial observations (Candès and Recht, 2008; Mazumder et al.,

2010; Nguyen et al., 2019). One of the most famous examples of widespread use of *LRMC* is the "Netflix" competition, where the goal is to build a movie recommendation system based on users ratings of movies they have seen (Feuerverger et al., 2012). Here M is the matrix of movie ratings (score from one to five), where each column represents a movie, each row represents a user, and most elements in M are missing. We seek a low-rank matrix X that approximates the observed entries of M well, and can be used to fill in the missing values. The low-rank structure has a heuristic justification in terms of "cliques" of users and "genres" of movies.

Following Candès and Tao (2010), to state the corresponding optimization problem one denotes a set of observed entries by $\Omega \in \{1, \ldots, n\} \times \{1, \ldots, p\}$ and defines the projection operator $P_\Omega : \mathbb{R}^{n \times p} \to \mathbb{R}^{n \times p}$ as follows

$$(P_\Omega(X))_{ij} = \begin{cases} 0, \text{ if } (i,j) \notin \Omega \\ X_{ij}, \text{ if } (i,j) \in \Omega. \end{cases}$$

In other words, the projection operator replaces all missing elements by zeroes. The *LRMC* problem becomes

$$\underset{X \in \mathbb{R}^{n \times p}}{\text{minimize}} \, \|P_\Omega(M - X)\|_F^2 \quad \text{subject to } \operatorname{rk}(X) \leq k. \tag{2.2}$$

The objective in (2.2) is nothing but the residual sum of squares computed for the set of observed values only, i.e. $\sum_{(i,j) \in \Omega} (M_{ij} - X_{ij})^2$. Unlike the complete matrix case, where an explicit solution exists, *LRMC* requires an iterative algorithm. *Hard-impute* is an example of such an algorithm, and alternates two steps: the imputation step, where the current guess for X is used to complete the unobserved entries in M; and the SVD step which updates the low-rank matrix approximation X (Mazumder et al., 2010).

Note that problem (2.2) is non-convex, which makes it hard to establish a formal proof of convergence of the algorithm. A natural convex relaxation of the *LRMC* problem can be stated as follows (Mazumder et al., 2010; Candès and Tao, 2010):

$$\underset{X \in \mathbb{R}^{n \times m}}{\text{minimize}} \, \frac{1}{2} \|P_\Omega(M - X)\|_F^2 + \lambda \|X\|_* \tag{2.3}$$

Here $\|\cdot\|_*$ refers to the nuclear norm of a matrix (sum of singular values) and is a convex relaxation of the matrix rank function. The iterative algorithm solving (2.3) is called *soft-impute*. It is essentially the same as hard-impute, but instead of truncating some eigenvalues to zero at the second step (i.e. computing low-rank approximation) they are shrunk via a soft-thresholding operator (Mazumder et al., 2010). There is a link between problem (2.2) and its convex relaxation. Since shrinking its nuclear norm is a smooth way to reduce the rank of a matrix, then for λ large enough problem (2.3) will produce a rank k solution.

An alternative view of *LRMA* was suggested by Rennie and Srebro (2005). One can show that

any rank k matrix $X \in \mathbb{R}^{n\times p}$ can be rewritten as $X = AB^\top$, where $A \in \mathbb{R}^{n\times k}, B \in \mathbb{R}^{p\times k}$. Restating problem (2.2) in terms of A and B leads to an equivalent bi-convex problem:

$$\underset{A\in\mathbb{R}^{n\times k},\ B\in\mathbb{R}^{p\times k}}{\text{minimize}} \|P_\Omega(M - AB^\top)\|_F^2 \tag{2.4}$$

For the convex relaxation the following result holds: if r is the rank of the solution to problem (2.3), then for any $k > r$ problem (2.3) is equivalent to

$$\underset{A\in\mathbb{R}^{n\times k},\ B\in\mathbb{R}^{p\times k}}{\text{minimize}} \frac{1}{2}\|P_\Omega(M - AB^\top)\|_F^2 + \frac{\lambda}{2}\|A\|_F^2 + \frac{\lambda}{2}\|B\|_F^2 \tag{2.5}$$

Hastie et al. (2015) suggest an efficient alternating least squares algorithm for solving problem (2.5).

2.1.3 Weighted generalization

In this chapter we focus on a weighted generalization of *LRMA*. There are many applications of *WLRMA* beyond matrix completion. In Ecology, Robin et al. (2019) and Kidziński et al. (2020) propose modelling populations of species using Poisson GLMs. The authors analyse the population matrices (with rows and columns corresponding to sites and species, respectively) using low-rank models. As a result, they develop an iterative algorithm that solves a matrix-type Poisson GLM optimization problem with a low-rank constraint. The algorithm is based on Newton descent where each Newtons step can be restated in the form of a *WLRMA* problem.

In Tuzhilina et al. (2020) the authors build a technique for chromatin conformation reconstruction. They model the so-called contact matrix, which represents the frequency of contacts between each pair of genomic loci, via a Poisson distribution and link mean contact counts to the pairwise distances between loci. The low-rank constraint is natural here, since chromatin is a 3D structure; in addition the authors model chromatin as a smooth curve. The resulting optimization problem can be solved via an iterative algorithm, where each iteration is equivalent to solving a *WLRMA* problem.

Note that the projection operator introduced in the previous section is equivalent to multiplying each element of a matrix either by zero (for unobserved entries) or by one (for observed entries). In other words, $P_\Omega(X)$ can be rewritten as $W * X$, where $W \in \mathbb{R}^{n\times p}$ is a binary matrix with

$$W_{ij} = \begin{cases} 0, \text{ if } (i,j) \notin \Omega \\ 1, \text{ if } (i,j) \in \Omega \end{cases}$$

and $*$ refers to the Hadamard element-wise product. The natural generalization of the *LRMC*

problem is therefore

$$\underset{X\in\mathbb{R}^{n\times p}}{\text{minimize}}\,\|\sqrt{W}*(M-X)\|_F^2 \quad \text{subject to } \operatorname{rk}(X)\leq k. \tag{2.6}$$

Here $W\in[0,1]^{n\times p}$ is a matrix of weights (not necessary binary) and the objective is simply weighted residual sum of squares $\sum_{i=1}^{n}\sum_{j=1}^{p}W_{ij}(M_{ij}-X_{ij})^2$.

Of course, there is a convex relaxation of problem (2.6), i.e.

$$\underset{X\in\mathbb{R}^{n\times p}}{\text{minimize}}\,\frac{1}{2}\|\sqrt{W}*(M-X)\|_F^2+\lambda\|X\|_* \tag{2.7}$$

The *WLRMA* problem was first introduced by Srebro and Jaakkola (2003) (see also Razenshteyn et al., 2016; Dutta et al., 2018; Ban et al., 2019). Originally the authors suggested a simple iterative algorithm that successively replaces the current guess of X by a low-rank-matrix approximation of $W*M+(1-W)*X$. This iterative procedure has a natural interpretation: you first "blend" your current guess for X with the ground truth M (using weights W and $1-W$) and then project the combination back to the manifold of rank-k matrices. With a matrix of binary weights, this procedure coincides with the hard-impute approach discussed above.

This chapter is organized as follows. In Section 2.2 we link the existing *WLRMA* approach (hereafter we call it baseline) as well as the soft- and hard-impute algorithms to projected gradient descent. In Section 2.3.1 we use this link to propose an acceleration technique for the baseline method based on Nesterov acceleration. Further we provide an alternative way to improve the speed of *WLRMA* convergence: we restate the original problem as a fixed point equation and combine projected gradient descent with Anderson acceleration. We discuss these ideas as well as the way to stabilize the convergence behavior of Anderson acceleration in Sections 2.3.2 and 2.5. All of the proposed algorithms require us to compute an SVD at each iteration, which is computationally expensive in high-dimensions. In Section 2.6 we discuss an adaptation of the proposed techniques to the high-dimensional data setting. Finally, we demonstrate and compare the performance of all the methods on a simulation example as well as a real-data example from recomendataion systems (see Sections 2.4 and 2.7). When we finished writing this chapter we discovered a recent preprint, which shares some of the ideas we propose in Sections 2.3.1 and 2.6 (Dutta et al., 2021).

2.2 Link to proximal gradient descent

In this section we link the *WLRMA* algorthm proposed in Srebro and Jaakkola (2003) to proximal gradient descent. This connection will subsequently help us to develop several useful algorithmic extensions. Note that the same link can be established for the convex relaxation of the *WLRMA* problem, as well as the soft- and hard-impute approaches.

Proximal gradient descent (PGD) is a method that allows one to solve non-differentiable convex

optimization problems (Combettes and Wajs, 2005; Combettes and Pesquet, 2011). Specifically, suppose you aim to minimize $g(x) + h(x)$ w.r.t. $x \in \mathbb{R}^d$, where $g(x)$ is convex and differentiable and $h(x)$ is convex and not necessarily differentiable. For the non-differentiable part we define the *proximal operator* as follows:

$$\text{prox}_h(x) = \text{argmin}_z \frac{1}{2}\|x - z\|_2^2 + h(z).$$

If $x^{(0)}$ refers to some initialization of x and $x^{(i)}$ is the value of the optimized parameter at iteration i then the proximal gradient descent update is

$$x^{(i+1)} := \text{prox}_h(x^{(i)} - \nabla g(x^{(i)})).$$

The update can be split in two parts: first the standard gradient step is done using the differentiable part of the objective, then the non-differentiable part is taken into account via proximal mapping.

Now we link PGD to the *WLRMA* problems (2.6) and (2.7). For the non-convex version we first restate problem (2.6) as follows:

$$\underset{X \in \mathbb{R}^{n \times p}}{\text{minimize}} \frac{1}{2}\|\sqrt{W} * (M - X)\|_F^2 + \mathbf{1}\{X \in C\}$$

Here $C = \{X \in \mathbb{R}^{n \times p},\ \text{rk}(X) \leq k\}$ is the set of rank k matrices and

$$\mathbf{1}\{X \in C\} = \begin{cases} 0, \text{ if } X \in C, \\ \infty, \text{ if } X \notin C. \end{cases}$$

is the set indicator function. Since indicator is a non-differentiable function, we set

$$g(X) = \tfrac{1}{2}\|\sqrt{W} * (M - X)\|_F^2 \quad \text{and} \quad h(X) = 1_C(X).$$

It is not difficult to prove that the proximal operator of an indicator function of a set is just the projection onto this set (in the case of the set constraint the algorithm is therefore referred as *Projected Gradient Descent*). Further, note that projecting onto the set of rank k matrices is equivalent to solving *LRMA* problem (2.1). Denote the rank k approximation of matrix X by $\text{SVD}_k(X)$, i.e. if $X = UDV^\top$ is the singular value decomposition then

$$\text{SVD}_k(X) = U\,\text{diag}(d_1, \ldots, d_k, 0, \ldots, 0)V^\top.$$

Computing the gradient of the differential part of the objective, i.e.

$$\nabla g(X) = -W * (M - X),$$

we get the following PGD update for the *WLRMA* problem:

$$X^{(i+1)} := \mathrm{SVD}_k(W * M + (1 - W) * X^{(i)}). \tag{2.8}$$

Although motivated by the EM algorithm, the update suggested in Srebro and Jaakkola (2003) coincides with the one we just derived via PGD.

Now let us consider the convex relaxation. Since in problem (2.7) the nuclear norm is a non-differentiable function of X, we set

$$f(X) = \tfrac{1}{2}\|\sqrt{W} * (M - X)\|_F^2 \text{ and } h(X) = \lambda\|X\|_*.$$

One can show that proximal operator for $h(X)$ in this case is the soft-threshold operator (Mazumder et al., 2010), i.e. $\mathrm{prox}_C(X) = \mathrm{S}_\lambda(X)$ where

$$\mathrm{S}_\lambda(X) = U\,\mathrm{diag}((d_1 - \lambda)_+, \ldots, (d_p - \lambda)_+)V^\top.$$

and $x_+ = \max(0, x)$ is the threshold operator. This leads us to the PGD update as follows:

$$X := \mathrm{S}_\lambda(W * M + (1 - W) * X). \tag{2.9}$$

Similar to standard gradient descent, it is possible to add a learning rate t to the PGD procedure leading to the hard and soft updates

$$\begin{aligned} X^{(i+1)} &:= \mathrm{SVD}_k(tW * M + (1 - tW) * X^{(i)}) \text{ and} \\ X^{(i+1)} &:= \mathrm{S}_{t\lambda}(tW * M + (1 - tW) * X^{(i)}) \end{aligned}$$

respectively. Although optimizing the learning rate may improve convergence, it can be quite expensive to run line search in the context of *WLRMA*, as a separate SVD is required for each value of t.

2.3 Acceleration Techniques

There exist several ways to accelerate proximal gradient descent. We discuss two in this section: Nesterov and Anderson acceleration.

2.3.1 Nesterov

PGD can be combined with Nesterov acceleration, which uses momentum from previous iterations (Nesterov, 1983). Specifically, if we denote

$$V^{(i)} = X^{(i)} + \frac{i-1}{i+2}\left(X^{(i)} - X^{(i-1)}\right) \tag{2.10}$$

then the PGD procedure for both non-convex and convex versions of the *WLRMA* is as follows:

$$X^{(i+1)} := \mathrm{SVD}_k(W * M + (1-W) * V^{(i)}) \text{ and} \tag{2.11}$$

$$X^{(i+1)} := \mathrm{S}_\lambda(W * M + (1-W) * V^{(i)}). \tag{2.12}$$

2.3.2 Anderson

Here we propose an alternative way to improve the convergence of the *WLRMA* baseline method via Anderson acceleration. To make it concise we present this technique for the non-convex *WLRMA* problem only. The same extension can be done for the convex relaxation as well.

Anderson acceleration (AA), or Anderson mixing, is a well-known method demonstrated to be very efficient for accelerating the convergence of fixed-point iterations (Anderson, 1965; Walker and Ni, 2011; Zhang et al., 2020). It is similar in flavor to most acceleration methods which combine the previous updates of the optimized parameter to improve the convergence speed. Given some function $f(x) : \mathbb{R}^d \to \mathbb{R}^d$, initialization $x^{(0)}$ and depth m, Anderson acceleration solves the fixed point equation $x = f(x)$ via the following steps.

Anderson acceleration (*AA*)

---

1. **[Initialize]** Set $x^{(1)} = f(x^{(0)})$.
2. *Repeat until convergence*:
   - 2.1 If $i < m$ we replace m by $m_* = \min\{m, i\}$. This step is required only for the first few iterations.
   - 2.2 Store updates in a matrix $F^{(i)} = \left(f^{(i-m)}, \ldots, f^{(i)}\right)$ where $f^{(j)} = f(x^{(j)})$.
   - 2.3 Store residuals in a matrix $R^{(i)} = \left(r^{(i-m)}, \ldots, r^{(i)}\right)$ where $r^{(j)} = f^{(j)} - x^{(j)}$.
   - 2.4 Find coefficients $\alpha = (\alpha_m, \ldots, \alpha_0)^\top$ such that $\sum_{j=0}^{m} \alpha_j = 1$ and that minimize the linear combination of residuals, i.e. $\|R^{(i)}\alpha\|_2$.
   - 2.5 Combine previous updates setting $x^{(i+1)} = F^{(i)}\alpha$.

Note that the optimization problem

$$\underset{\alpha=(\alpha_m,\ldots,\alpha_0)^\top}{\text{minimize}} \left\|R^{(i)}\alpha\right\|_2 \text{ subject to } \alpha^\top \mathbf{1} = 1 \tag{2.13}$$

has an explicit solution

$$\alpha^* = \frac{\left({R^{(i)}}^\top R^{(i)}\right)^{-1}\mathbf{1}}{\mathbf{1}^\top\left({R^{(i)}}^\top R^{(i)}\right)^{-1}\mathbf{1}} \tag{2.14}$$

which can be found in two steps. First we solve equation ${R^{(i)}}^\top R^{(i)}\theta = \mathbf{1}$ and then normalize the solution $\alpha = \frac{\theta}{\mathbf{1}^\top\theta}$.

To combine *WLRMA* with Anderson acceleration we first need to restate the PGD procedure as a fixed point equation. Following the idea suggested in the recent paper by Mai and Johansson (2020) we restate the *WLRMA* update as

$$Y^{(i+1)} = W * M + (1 - W) * X^{(i)} \text{ and } X^{(i+1)} = \mathrm{SVD}_k(Y^{(i+1)}).$$

Note that $Y^{(i)}$ is an update after the gradient step and in the case of binary weights it will simply represent the imputed matrix M. Then the convergence for the baseline *WLRMA* is guaranteed when the following fixed-point equation (written in terms of the auxiliary variable Y) holds

$$Y = f(Y) \text{ where } f(Y) = W * M + (1 - W) * \mathrm{SVD}_k(Y).$$

Note that in Anderson acceleration x is assumed to be a vector and the *WLRMA* problem is stated in matrix form, so in order to run Anderson acceleration in the context of *WLRMA* we need to vectorize all the matrices. We will use lower case letters to denote flattened matrices, e.g. $y = \mathrm{vec}(Y)$ and $f = \mathrm{vec}(f(Y))$. This leads us to the following implementation of the *WLRMA-AA* algorithm. In order to build similar algorithm for the convex relaxation one replace $\mathrm{SVD}_k(\cdot)$ by $\mathrm{S}_\lambda(\cdot)$.

Anderson weighted low-rank matrix approximation (*WLRMA-AA*)

---

1. **[Initialize]** Set $Y^{(0)}$ (e.g. column mean imputation) and $y^{(0)} = \mathrm{vec}(Y^{(0)})$, then compute $X^{(0)} = \mathrm{SVD}_k(Y^{(0)})$.

2. *Repeat until convergence*:

   2.1 Compute $f^{(i)} = \mathrm{vec}(W * M + (1 - W) * X^{(i)})$.

   2.2 Set $r^{(i)} = f^{(i)} - y^{(i)}$.

   2.3 Concatenate $R^{(i)} = [R^{(i-1)} : r^{(i)}], \quad F^{(i)} = [F^{(i-1)} : f^{(i)}]$.

   2.4 Solve OLS.

   - Solve ${R^{(i)}}^\top R^{(i)}\theta = \mathbf{1}$ w.r.t. θ.
   - Normalize $\alpha = \frac{\theta}{\mathbf{1}^\top\theta}$.

   2.5 Compute $y^{(i+1)} = F^{(i)}\alpha$ and set $Y^{(i+1)} = \mathrm{matrix}(y^{(i+1)})$.

2.6 Update $X^{(i+1)} = \text{SVD}_k(Y^{(i+1)})$.

2.7 Drop the first column of both $F^{(i)}$ and $R^{(i)}$ if they contain more than m columns.

One can find a link between the proposed algorithm and Nesterov acceleration. First note that since $\sum_{i=0}^{m} \alpha_i = 1$ the update for the auxiliary variable can be rewritten as:

$$Y^{(i+1)} = W * M + (1 - W) * \sum_{j=0}^{m} \alpha_j X^{(i-j)}. \tag{2.15}$$

Thus if we denote

$$V^{(i)} = \sum_{j=0}^{m} \alpha_j X^{(i-j)}, \tag{2.16}$$

which is nothing but some linear combination of previous X updates, then step 2.6 in the above algorithm will coincide with the second step of the Nesterov update (2.12).

Several tricks are usually applied to improve the performance of Anderson acceleration.

- Since the loss function is not guaranteed to decrease after each Anderson update, one can run guarded Anderson acceleration. In other words, for each step we compare the loss improvement between the simple PGD and accelerated one and choose the update with better improvement. Alternatively, since the most unstable behavior of Anderson acceleration is demonstrated in the first few iterations where the gradient has the largest norm, one can delay the acceleration by a few iterations and start after the most significant drop in the gradient was passed.

- Note that the only information required for computing the Anderson coefficients α is the covariance matrix of the residuals $R^{(i)}$, which costs $O(npm^2)$. Although in most applications depth is usually chosen to be relatively small ($3 - 30$, see Anderson, 1965), this computation can be quite expensive if X is high-dimensional and can exceed the SVD cost if one of n or p is less than m^2. Since only one column of $R^{(i)}$ is updated at each iteration, one can keep and successively update the residual covariance matrix instead of re-computing it after each loop. This reduces the cost to $O(npm)$.

2.4 Simulation

In this section we compare the proposed three methods, i.e. baseline, Nesterov and Anderson *WLRMA*, on a small simulation example. We generate the data as follows. First we draw $A \in \mathbb{R}^{n \times r}$ and $B \in \mathbb{R}^{p \times r}$ from standard normal distribution and compute $M = AB^\top + E$. Here $E \in \mathbb{R}^{n \times p}$ is some matrix of errors drawn from $\mathcal{N}(0, \sigma^2)$. We further generate the matrix of weights $W \in \mathbb{R}^{n \times p}$ via uniform distribution. In our experiments we set $n = 1000$, $p = 100$, $r = 70$ and $\sigma = 1$. We

initialize $X^{(0)}$ at zero matrix and using the three suggested algorithms, i.e baseline, Nesterov and Anderson *WLRMA*, we solve problems (2.6) and (2.7). For the latter approach, we set depth equal to three. For non-convex *WLRMA* we test $k = 20, 50, 70$. To make the results compatible, for the convex relaxation we search for λ that results in approximately the same value of the unpenalized loss (see optimization problem (2.6)). Thus we vary λ in the grid $\lambda = 100, 30, 5$. We track the relative change in the loss, i.e.

$$\Delta^{(i)} = \left| \frac{\ell(X^{(i+1)}) - \ell(X^{(i)})}{\ell(X^{(i)})} \right|. \tag{2.17}$$

Here

$$\ell(X) = \|\sqrt{W} * (M - X)\|_F^2$$

for the *WLRMA* problem and

$$\ell(X) = \frac{1}{2}\|\sqrt{W} * (M - X)\|_F^2 + \lambda\|X\|_*$$

for the convex relaxation. We stop the algorithm as soon as the relative difference is less than some threshold, i.e. $\Delta^{(i)} < \epsilon$. In our experiments, the maximum number of iterations is limited to 300 and the convergence threshold is set to $\epsilon = 10^{-8}$.

The results are presented in Figure 2.1. According to our simulation experiments, both acceleration methods reach the convergence points faster than the baseline. Moreover, Anderson improves the convergence speed much more than Nesterov in the case of the convex relaxation. Note also that the convergence of accelerated methods for non-convex *WLRMA* is more erratic comparing to the baseline.

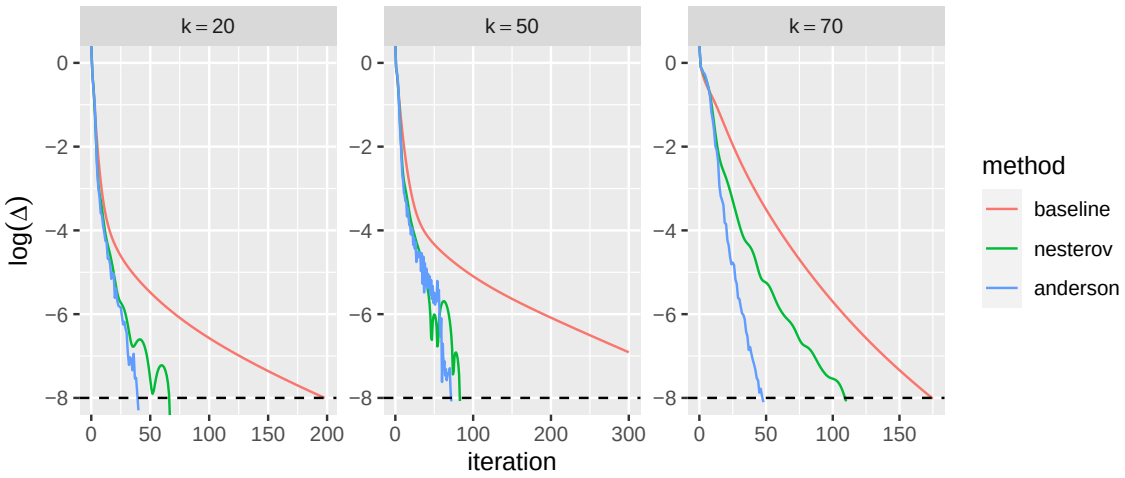
2.5 Stabilizing Anderson acceleration coefficients

In this section we propose a way to stabilize the behavior of Anderson acceleration in the case of non-convex *WLRMA*. First, let us plot the coefficient paths, i.e. α vs. iteration. In the left panel of Figure 2.2 we present the paths produced while solving the *WLRMA* problem with $k = 50$. We observe some significant oscillation in the coefficients, which results in the oscillation of the convergence path of Anderson *WLRMA* in Figure 2.1a (blue curve on the middle panel).

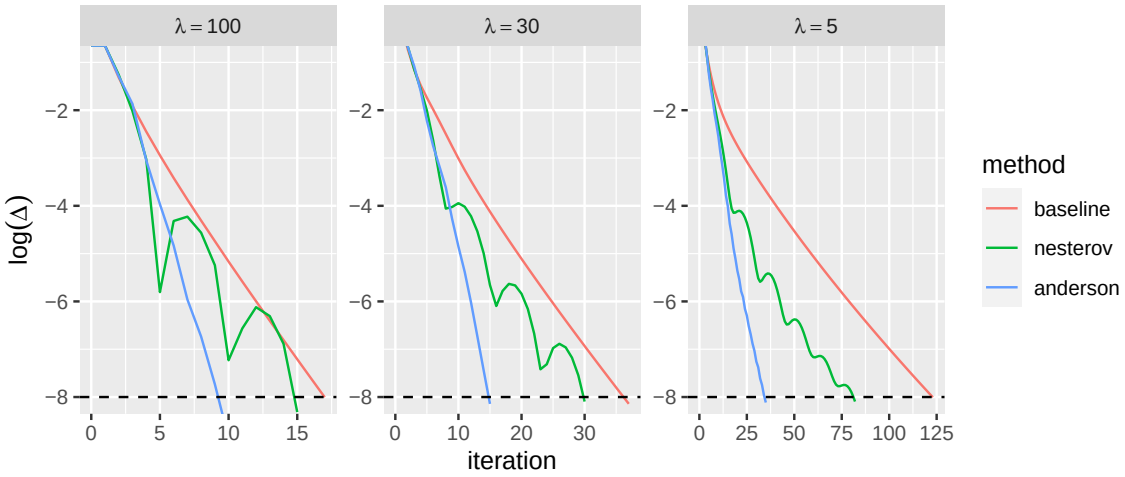
To stabilize the coefficient paths we add regularization and restate problem (2.13) as

$$\underset{\alpha=(\alpha_m,\ldots,\alpha_0)^\top}{\text{minimize}} \|R^{(i)}\alpha\|_2^2 + \gamma\|\alpha - \alpha_{prev}\|_2^2 \quad \text{subject to } \sum_{i=0}^{m} \alpha_i = 1. \tag{2.18}$$

Here α_{prev} is the coefficient vector obtained on the previous iteration (regularization depth is equal to one), or the average of d coefficient vectors obtained on several previous iterations (regularization



(a) Solving the non-convex *WLRMA* problem with different ranks k.



(b) Solving the *WLRMA* convex relaxation problem with different penalty factors λ.

Figure 2.1: Simulation example. Relative objective change vs. iteration number plotted for baseline, Nesterov and Anderson $m = 3$ algorithms. We match the rank and penalty factor of non-convex and convex problems so that top and bottom panels have similar value of the unpenalized objective.

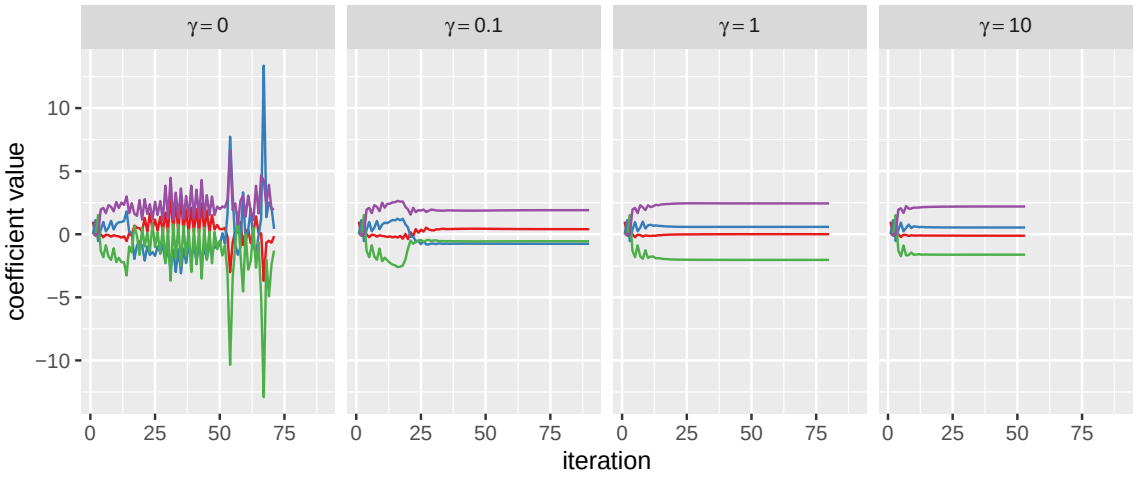


Figure 2.2: Coefficient paths obtained via Andersen acceleration with $m = 3$ while solving the *WLRMA* problem for $k = 50$. Each of the $m + 1$ coefficients is represented in different color. Left panel corresponds to acceleration without stabilization (blue curve on the middle panel in Figure 2.1a). The remaining panels represent acceleration with different smoothing penalties γ and α_{prev} equal to the average of three previous coefficient vectors.

depth is equal to d). The penalty in the above problem works in a similar way as acceleration methods: it balances the new update of the optimized variable with the previous guesses thereby smoothing the coefficient paths. On can derive the solution for problem (2.18).

$$\alpha^*_\gamma = (I + \gamma K)\alpha^* \text{ where } K = \left({R^{(i)}}^\top R^{(i)} + \gamma I\right)^{-1}\left(\alpha_{prev}\mathbf{1}^\top - \mathbf{1}\alpha_{prev}^\top\right)$$

$$\alpha^* = \frac{\left({R^{(i)}}^\top R^{(i)} + \gamma I\right)^{-1}\mathbf{1}}{\mathbf{1}^\top\left({R^{(i)}}^\top R^{(i)} + \gamma I\right)^{-1}\mathbf{1}}$$

Note that computing α^*_γ involves some simple manipulations with the small $(m + 1) \times (m + 1)$ covariance matrix of the residuals $R^{(i)}$. Since this matrix is also required to be computed while deriving the unpenalized solution (2.14), the cost for regularized and non-regularized Anderson is equivalent.

Now we test the proposed regularization technique on the same *WLRMA* problem. We vary γ, the smoothing penalty factor, in the grid $\gamma = 0.1, 1, 10$ and consider the average of three previous coefficient vectors for α_{prev}. The resulting coefficient paths are presented in Figure 2.2. As expected, increasing γ smooths the coefficient paths. We compare the convergence speed for different γ values. According to Figure 2.3 regularization also makes the convergence curves more smooth. Moreover, for large enough values of γ stabilizing Anderson coefficients allows us to reach the convergence point faster.

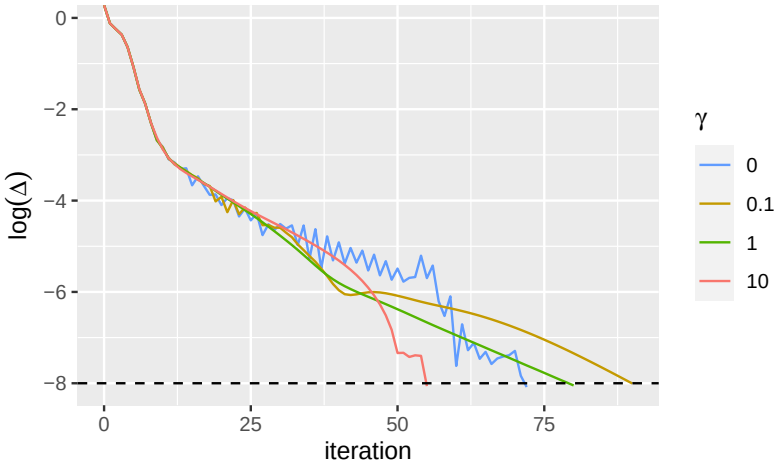


Figure 2.3: Convergence comparison for Anderson ($\gamma = 0$) and stabilized Anderson ($\gamma = 0.1, 1, 10$). We set α_{prev} to the average of the previous three coefficient vectors. We solve the *WLRMA* problem for $k = 50$, thus blue curve on this plot corresponds to the blue curve on the middle plot of Figure 2.1a.

2.6 High dimensions

2.6.1 Alternating least squares

In some applications, where n and/or p are extremely large the SVD part of the proximal operator is not plausible. For example, in the Netflix competition the training data matrix was approximately $400K$ by $20K$. In this section we borrow an idea from Hastie et al. (2015), proposed for the low-rank matrix approximation problem with missing values, that allows us to avoid the SVD in high dimensions.

Similar to (2.4) it is possible to restate the *WLRMA* problem as

$$\underset{A \in \mathbb{R}^{n \times k},\ B \in \mathbb{R}^{p \times k}}{\text{minimize}} \|W * (M - AB^\top)\|_F^2 \tag{2.19}$$

This alternative view of the *WLRMA* problem is well-studied in literature (Razenshteyn et al., 2016; Dutta et al., 2018), where the authors propose simple iterative least squares algorithm that alternates two steps: fix A and solve a weighted least squares problem to find B; then do the same for A fixing B. Although the algorithm avoids the most expensive SVD part of the PGD computations, for general W, it is still only moderately efficient as multiple weighted regressions are required for each step (one for each row of A and B).

It is easy to show that the proximal step for (2.6) is equivalent to solving (2.19) with $M = X - \nabla g(X) = Y$ and unit weights, i.e. $W_{ij} = 1$. Thus, the alternating least squares update can be done via solving an unweighted multiresponse regression problem. To be precise, we get the

following updates for non-convex *WLRMA*

$$B = Y^\top A(A^\top A)^{-1} \text{ and } A = YB(B^\top B)^{-1}, \tag{2.20}$$

which leads us to the following SVD-free alternating algorithm.

High-dimensional weighted low-rank matrix approximation (*WLRMA-ALS*)

---

1. **[Initialize]** Set $A^{(0)} \in \mathbb{R}^{n\times k}$ and $B^{(0)} \in \mathbb{R}^{p\times k}$ at random.

2. *Repeat until convergence*:

   2.1 Compute $X^{(i)} = A^{(i)}{B^{(i)}}^\top$ and $Y^{(i+1)} = W * M + (1 - W) * X^{(i)}$.

   2.2 Set $B^{(i+1)} := {Y^{(i+1)}}^\top A^{(i)}\big({A^{(i)}}^\top A^{(i)}\big)^{-1}$

   2.3 Update $X^{(i)} = A^{(i)}{B^{(i+1)}}^\top$ and $Y^{(i+1)} = W * M + (1 - W) * X^{(i)}$.

   2.4 Set $A^{(i+1)} := Y^{(i+1)}B^{(i+1)}\big({B^{(i+1)}}^\top B^{(i+1)}\big)^{-1}$.

Note that quite many variations of the above algorithm can be derived. For example, one can skip an extra update of $X^{(i)}$ and $Y^{(i+1)}$ (step 2.3). To recover the exact result of the projection step one should alternate 2.2 and 2.4 until the convergence.

It is not hard to build similar algorithm for the *WLRMA* convex relaxation. We first claim that for each $\lambda > 0$ there exists r such that for all $k > r$ problem (2.7) is equivalent to solving penalized problem

$$\underset{A\in\mathbb{R}^{n\times k},\ B\in\mathbb{R}^{p\times k}}{\text{minimize}} \frac{1}{2}\|W * (M - AB^\top)\|_F^2 + \frac{\lambda}{2}\|A\|_F^2 + \frac{\lambda}{2}\|B\|_F^2 \tag{2.21}$$

Next, note that if one matrix (A or B) is fixed and the weights are unit, then the other matrix can be found via solving a ridge regression problem. Thus replacing the ALS updates at step 2.2 of the above algorithm by

$$B = Y^\top A(A^\top A + \lambda I)^{-1} \text{ and } A = YB(B^\top B + \lambda I)^{-1} \tag{2.22}$$

we get a simple alternating procedure solving the *WLRMA* convex relaxation (2.7). To check the convergence of the algorithm one needs to compute the loss functions of (2.19) or (2.21). This, again, can be done efficiently using the sparse structure of W.

2.6.2 Sparse weights

In some applications the matrix of weights W can be very sparse. For instance, in the Netflix application, where almost all movie ratings are unobserved, W will mostly contain zeros. It turns

out that for sparse weight matrices the storage and computational cost of the ALS algorithm can be significantly reduced. The trick we propose below is especially useful when the high dimensionality of the problem makes it impossible to store $n \times p$ matrices.

Denote Ω the set of non-zero element of W. Following Hastie et al. (2015) we first note that matrix $Y^{(i+1)}$ at step 2.1 and 2.3 can be represented in a *sparse + low-rank* form as follows:

$$Y^{(i+1)} = \underbrace{W * (M - X^{(i)})}_{sparse} \quad + \quad \underbrace{X^{(i)}}_{low-rank} = S^{(i)} + X^{(i)}.$$

Here the second term $X^{(i)}$ is a high-dimensional but low-rank matrix, that can be stored in the form of (A, B) pair. Furthermore, the first term $S^{(i)}$ is a sparse matrix, which takes only $O(k \times |\Omega|)$ to compute (using the low-rank structure of $X^{(i)}$) and which can be stored in the form of a $|\Omega|$-dimensional vector. This leads us to the implementation of ALS in the following algorithm.

Sparse high-dimensional weighted low-rank matrix approximation (*WLRMA-SALS*)

---

1. [**Initialize**] Set $A^{(0)} \in \mathbb{R}^{n \times k}$ and $B^{(0)} \in \mathbb{R}^{p \times k}$ at random.

2. *Repeat until convergence*:

   2.1 Compute sparse $S^{(i)} = W * (M - A^{(i)} {B^{(i)}}^\top)$.

   2.2 Compute $H_A = A^{(i)} \big({A^{(i)}}^\top A^{(i)} \big)^{-1}$ and set $B^{(i+1)} = {B^{(i)}}^\top A^{(i)} H_A + {S^{(i)}}^\top H_A$.

   2.3 Update sparse $S^{(i)} = W * (M - A^{(i)} {B^{(i+1)}}^\top)$.

   2.4 Compute $H_B = B^{(i+1)} \big({B^{(i+1)}}^\top B^{(i+1)} \big)^{-1}$ and set $A^{(i+1)} = A^{(i)} {B^{(i+1)}}^\top H_B + S^{(i)} H_B$

Note that H_A and H_B are cheap to calculate via multiplying skinny matrices. Moreover, step 2.2 and 2.4 can be simplified to

$$B^{(i+1)} = B^{(i)} + {S^{(i)}}^\top H_A \text{ and } A^{(i+1)} = A^{(i)} + S^{(i)} H_B,$$

respectively. Then steps 2.2. and 2.4 are easy to handle using the sparse structure of $S^{(i)}$, thereby making the computations for the ALS algorithm very efficient. Finally, for the convex relaxation we can simply use

$$H_A = A^{(i)} \big({A^{(i)}}^\top A^{(i)} + \lambda I \big)^{-1} \text{ and } H_B = B^{(i+1)} \big({B^{(i+1)}}^\top B^{(i+1)} + \lambda I \big)^{-1}$$

which still reduces to "cheap" multiplication of skinny matrices and inverting a small $r \times r$ matrix.

2.6.3 Acceleration in high dimensions

It is possible to extend all the tricks proposed in Section 2.6 to accelerated methods as well. For both Nesterov and Anderson acceleration we exploit the following idea. Note that each iteration of the *WLRMA-ALS* (and its sparse modification) algorithm can bed as an operator on a pair of matrices (A, B). Specifically, at step 2.1 the algorithm takes $(A^{(i)}, B^{(i)})$ and after some transformation at steps 2.1–2.4 it returns the updated pair $(A^{(i+1)}, B^{(i+1)})$. For simplicity, let us denote this operator by $\Phi(A, B)$; this operator will be the main building block of our acceleration algorithm. Further, to introduce acceleration to the algorithm we concatenate matrices A and B an denote the result by Z, thus $Z = \binom{A}{B}$. To build our accelerated ALS approaches we apply acceleration to Z. For example, in the Nesterov case this will be equivalent to applying acceleration separately to A and B.

High-dimensional Nesterov weighted low-rank matrix approximation (*WLRMA-NALS*)

1. [**Initialize**] Set $A^{(0)} \in \mathbb{R}^{n\times k}$ and $B^{(0)} \in \mathbb{R}^{p\times k}$ at random.
2. *Repeat until convergence*:
   2.1 Update $V_A^{(i)} = A^{(i)} + \frac{i-1}{i+2}\left(A^{(i)} - A^{(i-1)}\right)$ and $V_B^{(i)} = B^{(i)} + \frac{i-1}{i+2}\left(B^{(i)} - B^{(i-1)}\right)$.
   2.2 Update $\left(A^{(i+1)}, B^{(i+1)}\right) := \Phi\left(V_A^{(i)}, V_B^{(i)}\right)$

To build Anderson acceleration we consider our baseline-ALS algorithm as a successive application of the update $Z^{(i+1)} = \Phi(Z^{(i)})$. This leads us to the fixed-point equation $Z = \Phi(Z)$, which can be solved via Anderson acceleration outline discussed in Section 2.3.2. Note that the resulting method will operate in terms of skinny $(n + m) \times k$ matrices which makes it very efficient in terms of both computational and storage cost.

2.6.4 Find the solution rank

Sometimes it can be useful to track the solution rank while updating A and B. For example, one can consider the delayed Anderson acceleration (see Section 2.3.2) and initialize the acceleration as soon as the solution rank becomes less than k. To calculate the solution rank for low-rank $X = AB^\top$ one can apply the following simple steps:

- Decompose A, i.e. $A = U_A D_A V_A^\top$.
- Multiply $\tilde{B} = BV_A D_A$.
- Decompose $\tilde{B} = U_B D_B V_B^\top$.

Thus $X = (U_A V_B) D_B U_B^\top$ is the SVD of X and the number of non-zero elements in D_B will represent the solution rank. Again, since we work in terms of skinny matrices only, the SVD is cheap to compute.

2.7 MovieLens data

In this section we test the proposed methods on a real dataset. The dataset describes a 5-star rating and from MovieLens, a movie recommendation service. It contains 1 million ratings from 6000 users on 4000 movies that we store in a large 6000×4000 matrix M. Note that M is very sparse: some of the movies were rated by just a few users (if any) and only 5% of values are observed. Although the SVD-based approaches proposed in Sections 2.2–2.3 could be applied to this data, each iteration takes several minutes to complete, which makes the convergence process extremely slow.

In Figure 2.4 we present the results for three ALS methods: baseline, Nesterov and Anderson with depth $m = 3$. To initialize the algorithms we use a warm start, which is the ALS solution for non-weighted problem with unobserved values of X replaced by zeroes. To guarantee that loss decreases at each iteration for Anderson Acceleration we use the guarded modification (see Section 2.3.2). Using the three algorithms under consideration we solve the *WLRMA* convex relaxation problem varying λ in the grid $\lambda = 40, 50, 100$ which results in the solutions of rank $70, 20$ and 5, respectively. We limit the maximum number of iterations to 200 and set the convergence threshold to $\epsilon = 10^{-8}$.

According to the Figure, all three methods achieve reasonable accuracy of $\epsilon = 10^{-4}$ in just a few minutes. However, both acceleration methods demonstrate better convergence speed comparing to the baseline ALS technique. Note that for $\lambda = 100, 50$ Anderson achieves the threshold $\epsilon = 10^{-8}$ much faster than the Nesterov. Although for $\lambda = 40$ Nesterov is slightly faster than the competitor the rank plot (green curve on the bottom right panel) suggests that it stopped too early. Finally, we observe that for less accurate solutions, e.g. the ones with $\epsilon = 10^{-4}$, Nesterov acceleration could outperform the Anderson approach.

2.8 Discussion

In this chapter we proposed a weighted generalization of the low-rank matrix approximation technique. Both classical *LRMA* problem, which has an explicit solution, and the famous *LRMA* problem with missing values, which can be solved via iterative soft- or hard-impute algorithms, are special cases of this weighted generalization. To build the baseline algorithm solving the weighted problem we utilize proximal gradient descent. We further suggest two ways to accelerate the baseline method. The first technique is based on the Nesterov acceleration which uses the momentum from the previous iteration. To develop the second acceleration method we restate the *WLRMA* problem as a fixed-point equation and subsequently utilize Anderson acceleration.

There is still much scope for future work. One interesting direction is to test the proposed methods on some real-data applications with sparse but non-binary weights. For example, one can consider a GLM model with a working response having some low-rank matrix structure. Then each step of the Newton gradient descent will be equivalent to solving a weighted low-rank matrix

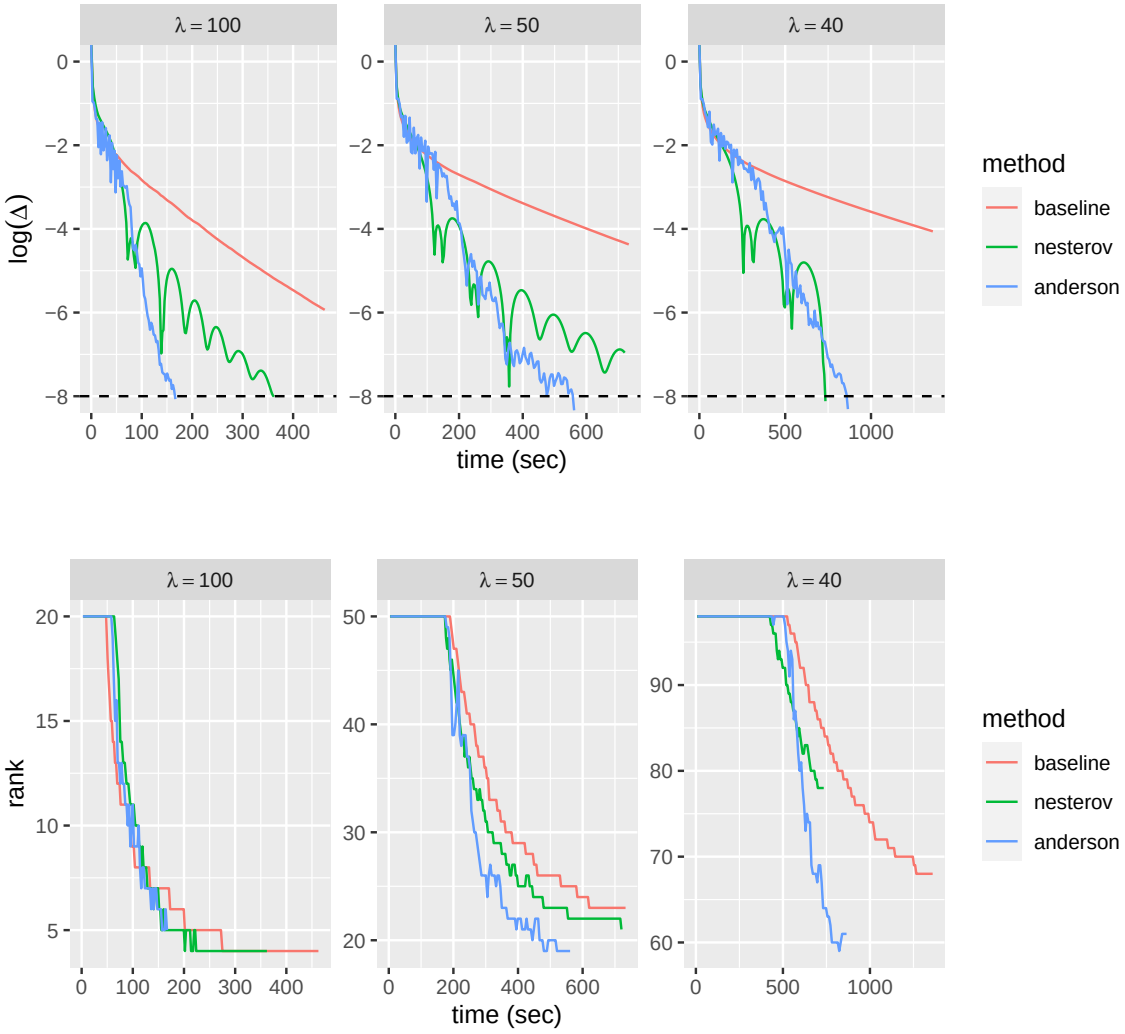


Figure 2.4: MovieLens example. Comparison of three ALS algorithms: baseline, Nesterov and Anderson with $m = 3$. The convex relaxation with different λ is solved. Top panel represents the objective relative change vs. time. Bottom panel represents the solution rank vs. time.

approximation problem. Furthermore, if a huge portion of data is missing, it becomes a *WLRMA* problem with a sparse weight matrix. Another direction could be to explore the ways to improve the stability of the Anderson acceleration. There are many tricks beyond the ones discussed in this chapter that could help us with this goal; for example, we can test Powell regularization or restart checking proposed in Zhang et al. (2020).

Chapter 3

Canonical correlation analysis and brain studies

3.1 Introduction and background

Canonical correlation analysis (*CCA*) is a classic method commonly used in statistics for studying complex multivariate data. *CCA* was first introduced by Hotelling (1936) as a tool for finding relationships between two sets of variables. It remains relevant in many domains including, but not limited to, genetics (Waaijenborg et al., 2008; Parkhomenko et al., 2009; Cao et al., 2009) and neuroscience (Wang et al., 2020; Zhuang et al., 2020). In many applications the number of available observations is significantly smaller than the number of features under consideration, so some form of regularization is essential. Numerous regularized *CCA* extensions have been proposed (Lykou and Whittaker, 2010; Hardoon and Shawe-Taylor, 2011; Witten and Tibshirani, 2009). The most popular existing approach, called *regularized CCA* (*RCCA*), imposes an ℓ_2-penalty on the canonical coefficients (Vinod, 1976; Leurgans et al., 1993). Like any other standard regularization method based on the ℓ_2 penalty, it has the property of treating all the coefficients equally and shrinking them toward zero. Although *RCCA* is well suited to data with general structure, in some applications the structure of the data can play an important role when investigating the association between variables. In this chapter we develop several regularized extensions of *CCA*. These extensions were originally motivated by brain imaging applications, but the scope of applications can readily be extended to other fields.

The chapter is organized as follows. In Section 3.3 we introduce the necessary background for both *CCA* and *RCCA* methods. Next, we propose several approaches to regularization that account for the underlying structure of the data. In particular, in Section 3.5 we introduce *partially regularized canonical correlation analysis* (*PRCCA*) that imposes an ℓ_2 penalty only on a subset of

the features. Although both *RCCA* and *PRCCA* problems have simple explicit solutions that can be computed via singular value decomposition, they require us to work in terms of sample covariance matrices. This can be infeasible when the number of features is very large. In Sections 3.3.3 and 3.5.2 we cover the "kernel" trick that allows to escape excessive computations while conducting *CCA* with regularization. In Section 3.6 we introduce *group regularized CCA* (*GRCCA*), a novel method that exploits group structure in the data. Since all the methods under consideration have similar structure, they can be considered as special cases of *CCA* with a general regularization penalty discussed in Sections 3.7. We illustrate the proposed methods on our motivating study example involving functional brain imaging data in Section 3.4 as well as on a small simulation example in Section 3.10. We conclude with a discussion, including some ideas for future work.

3.2 Canonical correlation analysis

Consider two random vectors $X \in \mathbb{R}^p$ and $Y \in \mathbb{R}^q$. The goal of *canonical correlation analysis (CCA)* is to find a linear combination of X variables and a linear combination of Y variables with the maximum possible correlation. Typically we find a sequence of such linear combinations. Namely, for $i = 1, \ldots, \min(p, q)$ define a sequence of pairs of random variables (U_i, V_i) as follows (Härdle and Simar, 2007)

1. Random variables U_i and V_i are linear combinations of X and Y, respectively, i.e.

$$U_i = \alpha_i^\top X \quad \text{and} \quad V_i = \beta_i^\top Y.$$

2. Coefficient vectors $\alpha_i \in \mathbb{R}^p$ and $\beta_i \in \mathbb{R}^q$ maximize the correlation

$$\rho(\alpha_i, \beta_i) = \text{cor}(\alpha_i^\top X, \beta_i^\top Y).$$

3. Pair (U_i, V_i) is uncorrelated with previous pairs, i.e

$$\text{cor}(U_i, U_j) = \text{cor}(V_i, V_j) = 0 \text{ for } j < i.$$

The pair (U_i, V_i) is called *i-th pair of canonical variates*; the corresponding optimal correlation value $\rho_i = \rho(\alpha_i, \beta_i)$ is called *i-th canonical correlation.*

Note that correlation coefficient $\rho(\alpha, \beta)$ can be rewritten as

$$\rho_{CCA}(\alpha, \beta) = \frac{\alpha^\top \Sigma_{XY} \beta}{\sqrt{\alpha^\top \Sigma_{XX} \alpha}\, \sqrt{\beta^\top \Sigma_{YY} \beta}}, \tag{3.1}$$

where Σ_{XX}, Σ_{YY} and Σ_{XY} refer to the covariance matrices $\text{cov}(X)$, $\text{cov}(Y)$ and $\text{cov}(X, Y)$, respectively. It is easy to restate maximization of $\rho_{CCA}(\alpha, \beta)$ w.r.t. α and β in terms of a constrained

optimization problem

$$\underset{\alpha\in\mathbb{R}^p,\ \beta\in\mathbb{R}^q}{\text{maximize}}\ \alpha^\top\Sigma_{XY}\beta \text{ subject to } \alpha^\top\Sigma_{XX}\alpha = 1 \text{ and } \beta^\top\Sigma_{YY}\beta = 1. \tag{3.2}$$

Thus, finding the i-th canonical pair is equivalent to solving the problem:

$$\underset{\alpha_i\in\mathbb{R}^p,\ \beta_i\in\mathbb{R}^q}{\text{maximize}}\ \alpha_i^\top\Sigma_{XY}\beta_i \text{ subject to } \alpha_i^\top\Sigma_{XX}\alpha_i = 1 \text{ and } \beta_i^\top\Sigma_{YY}\beta_i = 1$$
$$\alpha_i^\top\Sigma_{XX}\alpha_j = 0 \text{ and } \beta_i^\top\Sigma_{YY}\beta_j = 0 \text{ for } j < i.$$

One can show that the canonical variates can be found via a singular value decomposition of the matrix $\Sigma_{XX}^{-\frac{1}{2}}\Sigma_{XY}\Sigma_{YY}^{-\frac{1}{2}}$, and that the canonical correlations coincide with the singular values of this matrix (Mardia et al., 1979).

3.3 Regularized canonical correlation analysis

3.3.1 Dealing with high dimensions

In practice, we replace covariance matrices Σ_{XX}, Σ_{YY} and Σ_{XY} by the sample covariance matrices $\widehat{\Sigma}_{XX}$, $\widehat{\Sigma}_{YY}$ and $\widehat{\Sigma}_{XY}$. Specifically, suppose $\mathbf{X} \in \mathbb{R}^{n\times p}$ and $\mathbf{Y} \in \mathbb{R}^{n\times q}$ refer to matrices of n observations for random vectors X and Y, respectively. Without loss of generality, assume that the columns of $\mathbf{X}$ and $\mathbf{Y}$ are centered (mean 0), then

$$\widehat{\Sigma}_{XX} = \tfrac{1}{n}\mathbf{X}^\top\mathbf{X},\ \ \widehat{\Sigma}_{YY} = \tfrac{1}{n}\mathbf{Y}^\top\mathbf{Y} \ \text{ and } \ \widehat{\Sigma}_{XY} = \tfrac{1}{n}\mathbf{X}^\top\mathbf{Y}.$$

If the number of observations n is smaller than p and/or q, the corresponding sample covariance matrices are singular and the inverses $\widehat{\Sigma}_{XX}^{-\frac{1}{2}}$ and/or $\widehat{\Sigma}_{YY}^{-\frac{1}{2}}$ do not exist. *Regularized canonical correlation analysis (RCCA)* resolves this problem by adding diagonal matrices to the sample covariance matrices of X and Y (Leurgans et al., 1993; Gonzalez et al., 2008):

$$\widehat{\Sigma}_{XX}(\lambda_1) = \widehat{\Sigma}_{XX} + \lambda_1 I_p \ \text{ and } \ \widehat{\Sigma}_{YY}(\lambda_2) = \widehat{\Sigma}_{YY} + \lambda_2 I_q. \tag{3.3}$$

Here I_p refers to the $p \times p$ identity matrix. The *modified correlation coefficient* that is maximized while seeking pairs of canonical variates is, hence,

$$\rho_{RCCA}(\alpha,\beta;\lambda_1,\lambda_2) = \frac{\alpha^\top\widehat{\Sigma}_{XY}\beta}{\sqrt{\alpha^\top(\widehat{\Sigma}_{XX}+\lambda_1 I)\alpha}\,\sqrt{\beta^\top(\widehat{\Sigma}_{YY}+\lambda_2 I)\beta}}. \tag{3.4}$$

By analogy with *CCA*, it is easy to show that the *RCCA* variates can be found via the singular value decomposition of the matrix $(\widehat{\Sigma}_{XX}+\lambda_1 I)^{-\frac{1}{2}}\widehat{\Sigma}_{XY}(\widehat{\Sigma}_{YY}+\lambda_2 I)^{-\frac{1}{2}}$ and that *RCCA* modified correlations are equal to the singular values of this matrix.

3.3.2 Shrinkage property

Similar to ridge regression, regularization shrinks the *CCA* coefficients α and β towards zero, where the *penalty parameters* λ_1 and λ_2 control the strength of the shrinkage of α and β, respectively. This can be supported by the following reasoning. As in the case of *CCA*, maximization of the modified correlation $\rho_{RCCA}(\alpha, \beta; \lambda_1, \lambda_2)$ w.r.t. α and β can be restated as a constrained optimization problem

$$\underset{\alpha \in \mathbb{R}^p,\ \beta \in \mathbb{R}^q}{\text{maximize}}\ \alpha^\top \widehat{\Sigma}_{XY} \beta \ \text{ subject to } \alpha^\top (\widehat{\Sigma}_{XX} + \lambda_1 I)\alpha = 1 \text{ and } \beta^\top (\widehat{\Sigma}_{YY} + \lambda_2 I)\beta = 1.$$

Note that the constraints can be rewritten as

$$\alpha^\top \widehat{\Sigma}_{XX} \alpha + \lambda_1 \|\alpha\|^2 = 1 \ \text{ and } \ \beta^\top \widehat{\Sigma}_{XX} \beta + \lambda_1 \|\beta\|^2 = 1$$

where $\|\cdot\|$ refers to the vector Euclidean norm. Finally, one can interpret λ_1 and λ_2 as Lagrangian dual variables for constraints $\|\alpha\|^2 \le t_1$ and $\|\beta\|^2 \le t_2$ which brings us to the optimization problem

$$\underset{\alpha \in \mathbb{R}^p,\ \beta \in \mathbb{R}^q}{\text{maximize}}\ \alpha^\top \widehat{\Sigma}_{XY} \beta \text{ subject to } \alpha^\top \widehat{\Sigma}_{XX} \alpha = 1 \text{ and } \beta^\top \widehat{\Sigma}_{YY} \beta = 1$$
$$\|\alpha\|^2 \le t_1 \text{ and } \|\beta\|^2 \le t_2. \tag{3.5}$$

One can show that for some appropriately chosen t_1 and t_2 this problem is equivalent to maximizing objective (3.4). Moreover, increasing λ_1 and λ_2 is equivalent to decreasing thresholds t_1 and t_2 which leads us to the shrinkage property. Finally, increasing λ_1 and λ_2 increases the denominator of (3.4) thereby shrinking the modified correlation coefficient to zero as well.

3.3.3 The *RCCA* kernel trick

In some applications we need to deal with a very high-dimensional feature space. For instance, analyzing functional magnetic resonance imaging (fMRI) data, where the dimension refers to the number of brain regions (or voxels), the number of features can reach hundreds of thousands. If one of p and q is very large it can be problematic to store matrices $\widehat{\Sigma}_{XX}$ and $\widehat{\Sigma}_{YY}$. In this section we illustrate a simple trick based on the invariance of the *RCCA* problem under orthogonal transformations, that allows one to handle high-dimensional data when computing the *RCCA* solution. The idea to reduce a high dimensional *CCA* problem to a low-dimensional one via the kernel trick was previously introduced by Kuss (2003) and Hardoon et al. (2005). Below we demonstrate the practical application of this idea to the *RCCA* problem that we subsequently use in the implementation.

For simplicity, we assume that regularization is imposed on the X part only, i.e. we assume $q < n$ and set $\lambda_2 = 0$. The same reasoning applies if we regularize Y part as well. First we use the fact that any $n \times p$ matrix $\mathbf{X}$ with $p \gg n$ can be decomposed (e.g. via SVD) into a product $\mathbf{X} = \mathbf{R}V^\top$, where $\mathbf{R} \in \mathbb{R}^{n \times n}$ is a square matrix, and $V \in \mathbb{R}^{p \times n}$ is a matrix with orthonormal columns, i.e. $V^\top V = I$.

Lemma 3 (The *RCCA* kernel trick)**.** *The original RCCA problem stated for* $\mathbf{X}$ *and* $\mathbf{Y}$ *can be reduced to solving the RCCA problem for* $\mathbf{R}$ *and* $\mathbf{Y}$*. The resulting canonical correlations and variates for these two problems coincide. The canonical coefficients for the original problem can be recovered via the linear transformation* $\alpha_X = V\alpha_R$*.*

Proof. Denote $V^\perp$ an orthogonal complement of matrix V, i.e. the matrix $V^\perp \in \mathbb{R}^{p\times(p-n)}$ such that $\widetilde{V} = \left(V, V^\perp\right) \in \mathbb{R}^{p\times p}$ is a full-rank orthogonal matrix. Then $V^\top\widetilde{V} = (I_n, 0)$. Denote also

$$\gamma = \widetilde{V}^\top\alpha = \begin{pmatrix} V^\top\alpha \\ (V^\perp)^\top\alpha \end{pmatrix} = \begin{pmatrix} \gamma_1 \\ \gamma_2 \end{pmatrix}.$$

Note that since there is a one-to-one correspondence between α and γ, the optimization of $\rho_{RCCA}(\alpha, \beta; \lambda_1, \lambda_2)$ w.r.t. to α is equivalent to optimization w.r.t. γ. Further, the following relation is true

$$\begin{aligned}
&\alpha^\top\frac{\mathbf{X}^\top\mathbf{Y}}{n}\beta = \gamma_1^\top\frac{\mathbf{R}^\top\mathbf{Y}}{n}\beta \\
&\alpha^\top\left(\frac{\mathbf{X}^\top\mathbf{X}}{n} + \lambda_1 I\right)\alpha = \gamma_1^\top\frac{\mathbf{R}^\top\mathbf{R}}{n}\gamma_1 + \lambda_1\gamma_1^\top\gamma_1 + \lambda_1\gamma_2^\top\gamma_2
\end{aligned}$$

Therefore, the correlation coefficient (3.4) with $\lambda_2 = 0$ can be rewritten in terms of R and Y as

$$\rho_{RCCA}(\gamma, \beta; \lambda_1) = \frac{\gamma_1^\top\widehat{\Sigma}_{RY}\beta}{\sqrt{\gamma_1^\top(\widehat{\Sigma}_{RR} + \lambda_1 I)\gamma_1 + \lambda_1\|\gamma_2\|^2}\sqrt{\beta^\top\widehat{\Sigma}_{YY}\beta}}.$$

It is easy to show that the maximum value of $\rho_{RCCA}(\gamma, \beta; \lambda_1)$ is attained when $\gamma_2 = 0$, so the above correlation coefficient is nothing but the *RCCA* correlation coefficient computed for $\mathbf{R}$ and $\mathbf{Y}$. Furthermore, the optimal value of $\alpha_X = \alpha$ can be recovered from $\alpha_R = \gamma_1$ by $\alpha = V\gamma_1$ and, since $\mathbf{X}\alpha = \mathbf{R}\gamma_1$, the canonical variates computed for $\mathbf{X}$ coincide with the ones computed for $\mathbf{R}$. □

Note that for $p \gg n$ the above trick allows us to avoid manipulating large $p \times p$ and $p \times q$ covariance matrices $\widehat{\Sigma}_{XX}$ and $\widehat{\Sigma}_{XY}$ and to operate in terms of smaller $n \times n$ and $n \times q$ matrices $\widehat{\Sigma}_{RR}$ and $\widehat{\Sigma}_{RY}$. Of course, exactly the same trick can be applied to the Y part if $q \gg n$.

3.3.4 Hyperparameter tuning

Before proceeding to our first example, let us discuss how one can tune the hyperparemeters. There are two hyperparameters for *RCCA*, i.e. λ_1 and λ_2. Let us denote the vector of hyperparameters by θ. The values for these hyperparameters can be chosen via cross-validation. Below we present the outline for hold-out cross-validation; however, it can be naturally extended to the case of k-fold cross-validation.

First we split all available observations into train $(\mathbf{X}_{train}, \mathbf{Y}_{train})$ and validation $(\mathbf{X}_{val}, \mathbf{Y}_{val})$ sets. We use the former set to fit the model and compute canonical coefficients $\alpha(\theta)$ and $\beta(\theta)$. Further, we use the latter set to estimate the model performance, i.e. we calculate $\rho_{val}(\theta) =$

$\text{cor}\left(\mathbf{X}_{val}\alpha(\theta), \mathbf{Y}_{val}\beta(\theta)\right)$. Note that here we utilize simple correlation instead of the modified correlation as a measure of performance. We pick the values of the hyperparameters maximizing the validation correlation, i.e. $\theta^* = \text{argmax}_\theta\left(\rho_{val}(\theta)\right)$, which can be done by means of grid search.

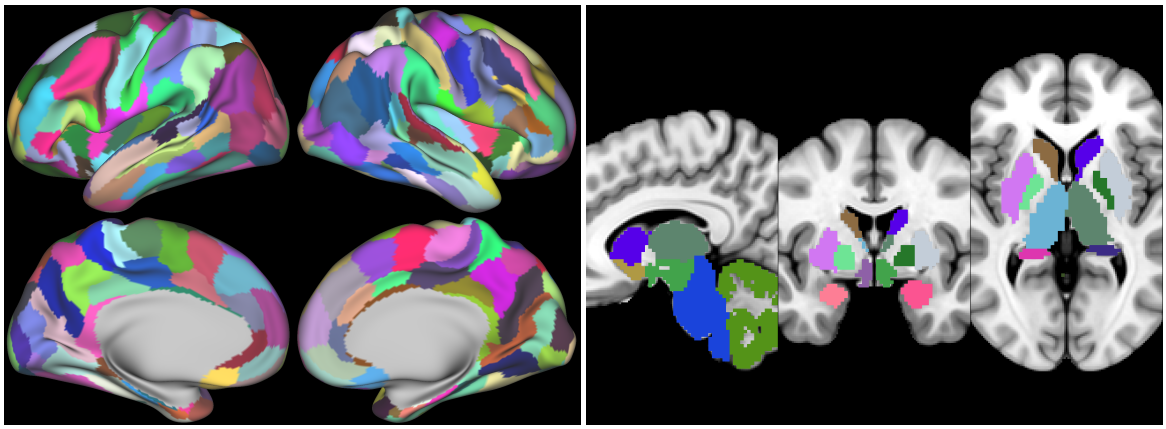
3.4 Human Connectome data study

In this section we present an application of regularized *CCA* to data from a neuroscience study: the Human Connectome Project for Disordered Emotional States (HCP-DES; Tozzi et al., 2020). One aim of HCP-DES is to link the function of macroscopic human brain circuits to self-reports of emotional well-being using magnetic resonance imaging. Here, we focused on brain activations during a Gambling task designed to probe the brain circuits underlying reward (described in detail in (Barch et al., 2013; Tozzi et al., 2020). We linked this neuroimaging data with self-reports assessing various aspects of reward-related behaviors (Behavioral Approach System/Behavioral Inhibition Scale (BIS/BAS); Carver and White, 1994), depression symptoms (Mood and Anxiety Symptom Questionnaire (MASQ); Wardenaar et al., 2010) and positive as well as negative affective states (Positive and Negative Affect Schedule (PANAS); Watson et al., 1988). We selected participants who had complete self-report and imaging data as well as no quality control issues, for a total of 153 participants (94 females, 59 males, mean age 25.91, sd 4.85). For details on the preprocessing and subject-level modeling used to derive brain activations in response to the Gambling task, see Section 3.12.1 of the Appendix. We used for our analysis the activations for the monetary reward compared to monetary loss during the task. For each subject, the activation at each greyordinate (grey matter coordinate) in the brain was extracted, yielding a matrix $\mathbf{X}$ of $n = 153$ rows (subjects) and $p = 90\ 368$ columns (greyordinates). The self-report data consisted of 9 variables: drive, fun seeking, reward responsiveness (from the BAS), total behavioral inhibition (from the BIS), distress, anhedonia, anxious arousal (from the MASQ), positive and negative affective states (from the PANAS). These were entered in a matrix $\mathbf{Y}$ of $n = 153$ rows (subjects) and $q = 9$ columns.

To test our structured methods, the 90 368 greyordinates were grouped into 229 regions depending on their functional and anatomical properties, corresponding to the 210 regions of the Brainnetome atlas (Fan et al., 2016) with the addition of the 19 subcortical regions of the Desikan-Killiany Atlas (Desikan et al., 2006). The resulting brain regions are presented in Figure 3.1.

3.4.1 Correlation analysis via *RCCA*

We start with a relatively simple model averaging activation inside each brain region and using the averaged values as features. Thus $\mathbf{X}$ becomes of dimension 153×229. To remove the effect of sex on the resulting correlation we adjusted both activation and behavioral data for the binary sex variable by means of simple linear regression (mean adjustment). Since the $\mathbf{Y}$ matrix is relatively small we imposed no penalty on $\mathbf{Y}$ ($\lambda_2 = 0$). To pick the hyperparameter λ_1 we ran 10-fold cross-validation on



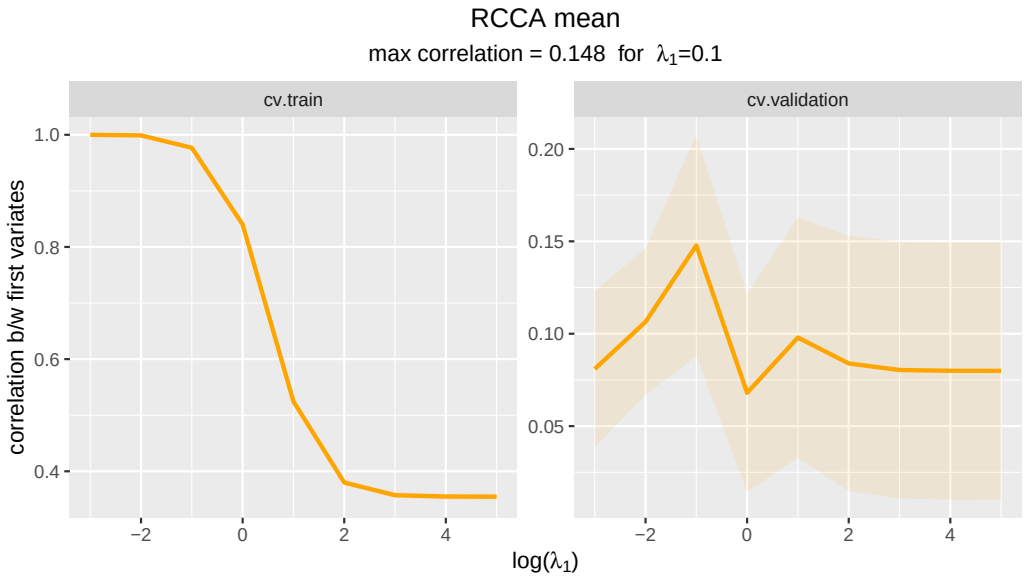
(a) Cortical parcellation. (b) Subcortical parcellation.

Figure 3.1: 229 brain regions: 210 cortical regions of the Brainnetome atlas and 19 subcortical regions of the Desikan-Killiany Atlas.

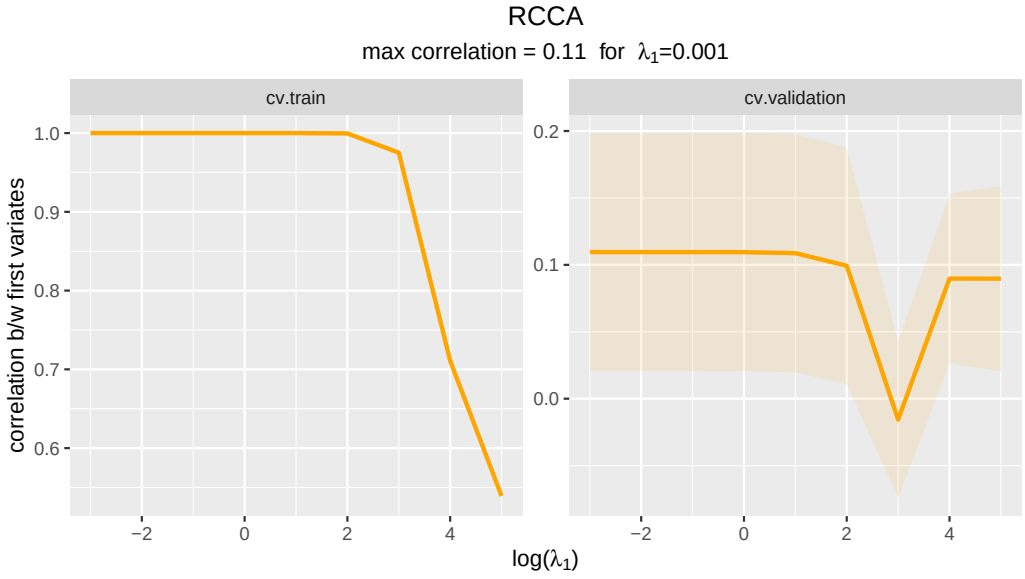
the adjusted data pairs with the penalty factor varying over the grid $\lambda_1 = 10^{-3}, 10^{-2}, \ldots, 10^4, 10^5$.

The resulting cross-validation curves represent the unpenalized correlation between canonical variates (computed on the 9 folds of train set and one fold of validation set) averaged across 10 folds (see Figure 3.2a). Note that although larger λ_1 shrinks the modified correlation ρ_{RCCA} toward zero, the unpenalized correlation is not guaranteed to be monotonically decreasing in λ_1. According to the plot, the highest score is achieved for $\lambda_1 = 0.1$ with the corresponding test correlation equal to 0.148. Using the kernel trick we now run *RCCA* for the original activation data (90 368 features adjusted for the sex effect). According to Figure 3.2b, the maximum score is equal to 0.11 ($\lambda_1 = 0.001$). In general, to compare two models and check that the cross-validation score does not reflect spurious findings we should validate the performance of the models on an independent test set. However, the small sample size of the data (only 153 observations) makes the test correlation estimates unreliable. We use nested cross-validation (NCV) to overcome the problem of overfitting to the dataset that we use for tuning. Specifically, we split the data in 11 folds. Each of the 11 folds is given an opportunity to be used as an independent test set, while all other 10 folds folds are used to tune the hyperparameters via 10-fold cross-validation. Therefore, we report 11 cross-validation scores along with 11 test scores, and we present the average for both as well as 1SE confidence intervals (see Figure 3.7). According to the NCV scores, the cross-validation procedure for the full *RCCA* model discovered significant correlation (independent test set correlation averaged across 11 folds is 0.105). However, the correlation value obtained by the smaller mean *RCCA* model was way too optimistic (average test score is 0.044 with wide error bands).

It is worth noting the computational speed of the proposed method. Unlike the `rcc()` function from the popular `CCA` R package (Gonzalez et al., 2008), which is not able to handle such a large number of features ($\approx$ 90K for the X side), our implementation of *RCCA* with the kernel trick



(a) Averaged data



(b) Original data

Figure 3.2: The cross-validation curves obtained via *RCCA* with 10-fold cross-validation for the Human Connectome dataset. Left panel: (unpenalized) correlation between train canonical variates. Right panel: (unpenalized) correlation between validation canonical variates.

completes the calculations in 20 seconds.

3.4.2 Interpretability of canonical coefficients

In this section we visualize the *RCCA* coefficients α corresponding to the hyperparameters chosen by cross-validation. Recall that the original X features represent the brain activation detected at each brain greyordinate, so we can map the resulting *RCCA* coefficients back to the brain surface (see Figure 3.3). There is an apparent trade-off between the interpretability and flexibility of the model. Namely, although full data *RCCA* is more flexible, there is quite a lot of variation in the resulting canonical coefficients. This makes the corresponding brain image harder to interpret. On the other hand, the reduced model allows us to identify the brain regions that have the highest impact on the resulting correlation. However, it loses in terms of flexibility (and, potentially, performance). In what follows, we aim to develop the model that links these two extremes enabling us to control the interpretability vs. flexibility trade-off.

3.5 Partially regularized canonical correlation analysis

3.5.1 Penalizing a subset of canonical coefficients

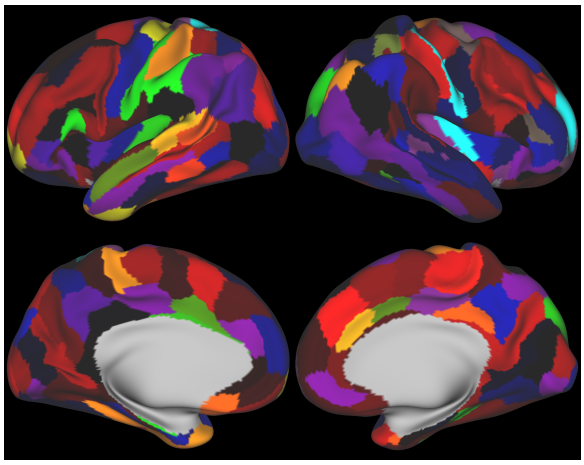
Suppose you are interested in the influence of a specific brain region on the resulting *CCA* correlation, however, you do not want to completely eliminate the remaining brain regions from the data. Recall that the inequality constraints in the *RCCA* optimization problem (3.5) control the deviation of all canonical coefficients from zero. *PRCCA* is a modification of *RCCA* that allows one to shrink only a subset of the *CCA* coefficients leaving the complement unpenalized. The proposed *PRCCA* method is a key building block for our final group *RCCA* approach and also admits some independent interesting applications.

Suppose that both α and β are split into two parts

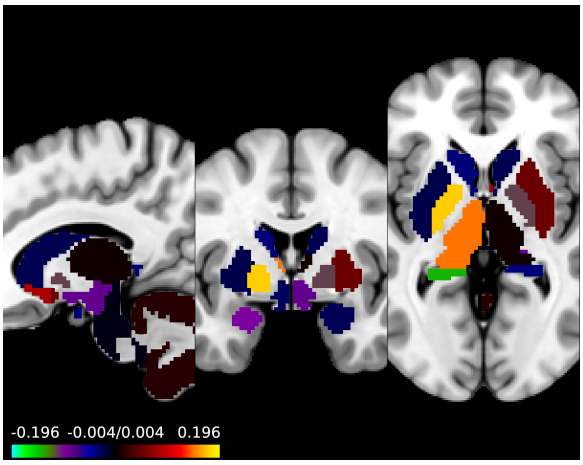
$$\begin{aligned}\alpha &= \left(\begin{smallmatrix}\alpha_1\\ \alpha_2\end{smallmatrix}\right) \text{ where } \alpha_1 \in \mathbb{R}^{p_1} \text{ and } \alpha_2 \in \mathbb{R}^{p_2} \text{ with } p_1 + p_2 = p,\\ \beta &= \begin{pmatrix}\beta_1\\ \beta_2\end{pmatrix} \text{ where } \beta_1 \in \mathbb{R}^{q_1} \text{ and } \beta_2 \in \mathbb{R}^{q_2} \text{ with } q_1 + q_2 = q.\end{aligned}$$

Replacing the constraints $\|\alpha\|^2 \leq t_1$ and $\|\beta\|^2 \leq t_2$ in the optimization problem (3.5) by $\|\alpha_1\|^2 \leq t_1$ and $\|\beta_1\|^2 \leq t_2$, respectively, we get the *PRCCA* optimization problem

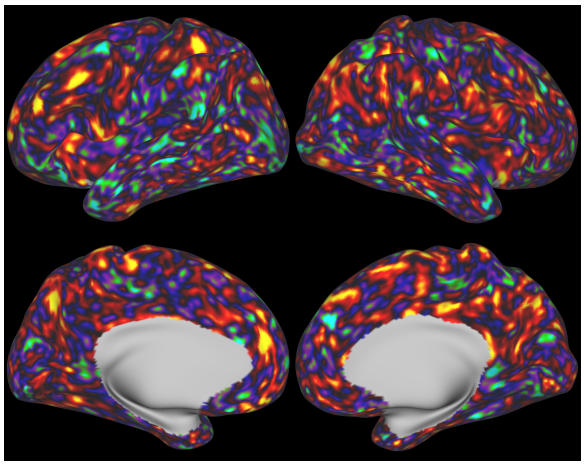
$$\begin{aligned}\underset{\alpha\in\mathbb{R}^p,\ \beta\in\mathbb{R}^q}{\text{maximize}}\ \alpha^\top \widehat{\Sigma}_{XY}\beta \text{ subject to } \ & \alpha^\top \widehat{\Sigma}_{XX}\alpha = 1 \text{ and } \beta^\top \widehat{\Sigma}_{YY}\beta = 1\\ & \|\alpha_1\|^2 \leq t_1 \text{ and } \|\beta_1\|^2 \leq t_2. \qquad (3.6)\end{aligned}$$



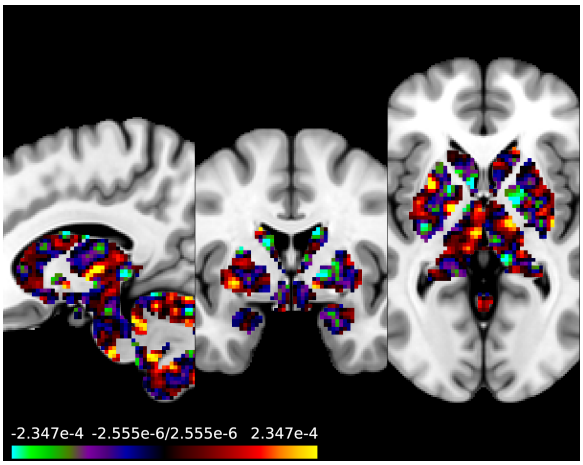
(a) Cortical coefficients, averaged data.



(b) Subcortical coefficients, averaged data.



(c) Cortical coefficients, original data.



(d) Subcortical coefficients, original data

Figure 3.3: The *RCCA* coefficients computed for the averaged data (229 features) with hyperparameter $\lambda_1 = 0.1$ chosen by cross-validation and for the original data (90 368 features) with hyperparameter $\lambda_1 = 10$ chosen by cross-validation.

Re-expressing the constraints for α and β in terms of dual variables we get

$$\alpha^\top \widehat{\Sigma}_{XX}\alpha + \lambda_1\|\alpha_1\|^2 = \alpha^\top \left(\widehat{\Sigma}_{XX} + \lambda_1 \left(\begin{smallmatrix} I_{p_1} & 0 \\ 0 & 0 \end{smallmatrix}\right)\right)\alpha$$
$$\beta^\top \widehat{\Sigma}_{YY}\beta + \lambda_2\|\beta_1\|^2 = \beta^\top \left(\widehat{\Sigma}_{YY} + \lambda_2 \left(\begin{smallmatrix} I_{q_1} & 0 \\ 0 & 0 \end{smallmatrix}\right)\right)\beta.$$

This leads us to the *PRCCA* modification of the correlation coefficient as follows

$$\rho_{PRCCA}(\alpha,\beta;\lambda_1,\lambda_2) = \frac{\alpha^\top \widehat{\Sigma}_{XY}\beta}{\sqrt{\alpha^\top(\widehat{\Sigma}_{XX} + \lambda_1 \left(\begin{smallmatrix} I_{p_1} & 0 \\ 0 & 0 \end{smallmatrix}\right))\alpha}\sqrt{\beta^\top(\widehat{\Sigma}_{YY} + \lambda_2 \left(\begin{smallmatrix} I_{q_1} & 0 \\ 0 & 0 \end{smallmatrix}\right))\beta}}. \tag{3.7}$$

As usual, the *PRCCA* variates and coefficients obtained by maximizing (3.7) can be found by means of SVD of the matrix $\left(\widehat{\Sigma}_{XX} + \lambda_1 \left(\begin{smallmatrix} I_{p_1} & 0 \\ 0 & 0 \end{smallmatrix}\right)\right)^{-\frac{1}{2}} \widehat{\Sigma}_{XY} \left(\widehat{\Sigma}_{YY} + \lambda_2 \left(\begin{smallmatrix} I_{q_1} & 0 \\ 0 & 0 \end{smallmatrix}\right)\right)^{-\frac{1}{2}}$.

3.5.2 The *PRCCA* kernel trick

In this section we extend the kernel trick to the *PRCCA* problem set up. Note that, because I was replaced by block matrix $\left(\begin{smallmatrix} I & 0 \\ 0 & 0 \end{smallmatrix}\right)$ in the denominator of the modified correlation coefficient (3.4), the *PRCCA* problem does not preserve the property of invariance under orthogonal transformations. Thus, the mathematics used in Section 3.3.3 does not work anymore. There are two main ingredients for the *PRCCA* kernel trick. First, if the feature matrix consists of two orthogonal blocks, then the kernel trick can be applied to each block independently. Second, there exists a non-orthogonal transformation of the feature matrix making the two blocks orthogonal to each other, while resulting in an equivalent *PRCCA* problem.

We again assume for simplicity that the regularization penalty is imposed on the X part only. Suppose $X = \left(\begin{smallmatrix} X_1 \\ X_2 \end{smallmatrix}\right)$, where random vectors $X_1 \in \mathbb{R}^{p_1}$ and $X_2 \in \mathbb{R}^{p_2}$ correspond to penalized coefficients part α_1 and unpenalized part α_2, respectively. Let $\mathbf{X}_1 \in \mathbb{R}^{n\times p_1}$ and $\mathbf{X}_2 \in \mathbb{R}^{n\times p_2}$ represent the corresponding matrices of observations, so $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2)$. To make the *PRCCA* solution identifiable we require $\mathbf{X}_2$ to be tall and full rank, i.e. $p_2 < n$ and $\operatorname{rank}(\mathbf{X}_2) = p_2$. We can also assume that $p_1 \gg n$.

As the first step we find a linear transformation $A \in \mathbb{R}^{p\times p}$ such that matrix $\widetilde{\mathbf{X}} = \mathbf{X}A = (\widetilde{\mathbf{X}}_1, \widetilde{\mathbf{X}}_2) \in \mathbb{R}^{n\times p}$ has orthogonal blocks $\widetilde{\mathbf{X}}_1 \in \mathbb{R}^{n\times p_1}$ and $\widetilde{\mathbf{X}}_2 \in \mathbb{R}^{n\times p_2}$, i.e. $\widetilde{\mathbf{X}}_1^\top \widetilde{\mathbf{X}}_2 = 0$, and that preserves the second block, i.e. $\widetilde{\mathbf{X}}_2 = \mathbf{X}_2$. This can be easily done by linear regression. This linear transformation maps the original *PRCCA* problem to an equivalent one (equivariant in terms of the coefficients and invariant in terms of the objective).

Note that the above transformation forces the sample covariance matrix $\widehat{\Sigma}_{\widetilde{X}\widetilde{X}}$ to be block-diagonal with blocks $\widehat{\Sigma}_{\widetilde{X}_1\widetilde{X}_1}$ and $\widehat{\Sigma}_{\widetilde{X}_2\widetilde{X}_2}$, which enables us to apply the kernel trick to the first and the second block of $\widetilde{\mathbf{X}}$ independently. Specifically, consider the decomposition of $\widetilde{\mathbf{X}}_1$, as in the *RCCA* kernel lemma, i.e. $\widetilde{\mathbf{X}}_1 = \mathbf{R}_1 V_1^\top$ where $V_1 \in \mathbb{R}^{p_1\times n}$ is a matrix with orthonormal columns

and $\mathbf{R}_1 \in \mathbb{R}^{n\times n}$ is some square matrix. Then the following lemma holds.

Lemma 4 (*PRCCA* kernel trick)**.** *The original PRCCA problem stated for* $\mathbf{X}$ *and* $\mathbf{Y}$ *can be reduced to solving the smaller PRCCA problem for* $\mathbf{R} = \left(\begin{smallmatrix}\mathbf{R}_1\\ \mathbf{X}_2\end{smallmatrix}\right)$ *and* $\mathbf{Y}$*. The resulting canonical correlations and variates for these two problems coincide. The canonical coefficients for the original problem can be recovered via the linear transformation* $\alpha_X = A\left(\begin{smallmatrix}V_1 & 0\\ 0 & I\end{smallmatrix}\right)\alpha_R$*.*

Proof. To find the required linear transformation A we first regress $\mathbf{X}_2$ from $\mathbf{X}_1$. We denote the matrix of regression coefficients by $B = (\mathbf{X}_2^\top\mathbf{X}_2)^{-1}\mathbf{X}_2^\top\mathbf{X}_1 \in \mathbb{R}^{p_2\times p_1}$ and set $A = \left(\begin{smallmatrix}I & 0\\ -B & I\end{smallmatrix}\right)$. This transformation leads to the following transformed matrix

$$\widetilde{\mathbf{X}} = \mathbf{X}A = (\widetilde{\mathbf{X}}_1, \widetilde{\mathbf{X}}_2) = (\mathbf{X}_1 - \mathbf{X}_2 B, \mathbf{X}_2)\,.$$

It is easy to check that matrix A is invertible and that the following relations hold

$$A^{-1} = \left(\begin{smallmatrix}I & 0\\ B & I\end{smallmatrix}\right) \quad \text{and} \quad A^{-T}\left(\begin{smallmatrix}I_{p_1} & 0\\ 0 & 0\end{smallmatrix}\right)A^{-1} = \left(\begin{smallmatrix}I_{p_1} & 0\\ 0 & 0\end{smallmatrix}\right).$$

Thus, if we denote $\widetilde{\alpha} = A^{-1}\alpha$, then

$$\begin{aligned}
&\alpha^\top\frac{\mathbf{X}^\top\mathbf{Y}}{n}\beta = \widetilde{\alpha}^\top\frac{\widetilde{\mathbf{X}}^\top\mathbf{Y}}{n}\beta\\
&\alpha^\top\left(\frac{\mathbf{X}^\top\mathbf{X}}{n} + \lambda_1\left(\begin{smallmatrix}I_{p_1} & 0\\ 0 & 0\end{smallmatrix}\right)\right)\alpha = \widetilde{\alpha}^\top\left(\frac{\widetilde{\mathbf{X}}^\top\widetilde{\mathbf{X}}}{n} + \lambda_1\left(\begin{smallmatrix}I_{p_1} & 0\\ 0 & 0\end{smallmatrix}\right)\right)\widetilde{\alpha}
\end{aligned}$$

The above equations imply that the *PRCCA* correlation coefficient (3.7) with $\lambda_2 = 0$ can be rewritten in terms of $\widetilde{\mathbf{X}}$ and $\mathbf{Y}$

$$\rho_{PRCCA}(\widetilde{\alpha}, \beta; \lambda_1) = \frac{\widetilde{\alpha}^\top\widehat{\Sigma}_{\widetilde{X}Y}\beta}{\sqrt{\widetilde{\alpha}^\top\left(\widehat{\Sigma}_{\widetilde{X}\widetilde{X}} + \lambda_1\left(\begin{smallmatrix}I_{p_1} & 0\\ 0 & 0\end{smallmatrix}\right)\right)\widetilde{\alpha}}\,\sqrt{\beta^\top\widehat{\Sigma}_{YY}\beta}}.$$

Next, let $\widetilde{\alpha} = \left(\begin{smallmatrix}\widetilde{\alpha}_1\\ \widetilde{\alpha}_2\end{smallmatrix}\right)$, where $\widetilde{\alpha}_1 \in \mathbb{R}^{p_1}$ and $\widetilde{\alpha}_2 \in \mathbb{R}^{p_2}$ correspond to blocks $\widetilde{\mathbf{X}}_1$ and $\widetilde{\mathbf{X}}_2$, respectively. Denote the orthogonal complement of V_1 by $V_1^\perp \in \mathbb{R}^{p_1\times(p_1-n)}$ and consider the following transformation of the *PRCCA* coefficients

$$\gamma_1 = V_1^\top\widetilde{\alpha}_1 \text{ and } \gamma_2 = \left(V_1^\perp\right)^\top\widetilde{\alpha}_1$$

as well as the concatenation $\gamma = \left(\begin{smallmatrix}\gamma_1\\ \widetilde{\alpha}_2\end{smallmatrix}\right)$. Then, by analogy one can show that

$$\widetilde{\alpha}^\top \frac{\widetilde{\mathbf{X}}^\top \mathbf{Y}}{n}\beta = \gamma_1^\top \frac{\mathbf{R}_1^\top \mathbf{Y}}{n}\beta + \widetilde{\alpha}_2^\top \frac{\widetilde{\mathbf{X}}_2^\top \mathbf{Y}}{n}\beta = \gamma^\top \frac{\mathbf{R}^\top \mathbf{Y}}{n}\beta$$

$$\widetilde{\alpha}^\top \left(\frac{\widetilde{\mathbf{X}}^\top \widetilde{\mathbf{X}}}{n} + \lambda_1 \left(\begin{smallmatrix} I_{p_1} & 0\\ 0 & 0\end{smallmatrix}\right)\right)\widetilde{\alpha} = \gamma_1^\top \left(\frac{\mathbf{R}_1^\top \mathbf{R}_1}{n} + \lambda_1 I\right)\gamma_1 + \lambda_1 \gamma_2^\top \gamma_2 + \widetilde{\alpha}_2^\top \frac{\widetilde{\mathbf{X}}_2^\top \widetilde{\mathbf{X}}_2}{n}\widetilde{\alpha}_2 =$$

$$= \gamma^\top \left(\frac{\mathbf{R}^\top \mathbf{R}}{n} + \lambda_1 \left(\begin{smallmatrix} I_{p_1} & 0\\ 0 & 0\end{smallmatrix}\right)\right)\gamma + \lambda_1 \gamma_2^\top \gamma_2$$

where the last equation holds since $\mathbf{R}_1^\top \widetilde{\mathbf{X}}_2 = V_1^\top \widetilde{\mathbf{X}}_1^\top \widetilde{\mathbf{X}}_2 = 0$.

Again, we can ignore $\lambda_1 \gamma_2^\top \gamma_2$ term as it is present only in the denominator of *PRCCA* correlation coefficient, which, therefore, can be rewritten in terms of $\mathbf{R}$ and $\mathbf{Y}$ as

$$\rho_{PRCCA}(\gamma, \beta; \lambda_1) = \frac{\gamma^\top \widehat{\Sigma}_{RY}\beta}{\sqrt{\gamma^\top \left(\widehat{\Sigma}_{RR} + \lambda_1 \left(\begin{smallmatrix} I_{p_1} & 0\\ 0 & 0\end{smallmatrix}\right)\right)\gamma}\ \sqrt{\beta^\top \widehat{\Sigma}_{YY}\beta}},$$

which is exactly the *PRCCA* correlation coefficient computed for $\mathbf{R}$ and $\mathbf{Y}$. The optimal value of $\alpha_X = \alpha$ for the original problem can be recovered from $\alpha_R = \gamma$ by

$$\alpha = A\widetilde{\alpha} = A\left(\begin{smallmatrix} V_1\gamma_1\\ \widetilde{\alpha}_2\end{smallmatrix}\right) = A\left(\begin{smallmatrix} V_1 & 0\\ 0 & I\end{smallmatrix}\right)\gamma.$$

Moreover,

$$\mathbf{X}\alpha = \widetilde{\mathbf{X}}\widetilde{\alpha} = (\mathbf{R}_1, \widetilde{\mathbf{X}}_2)\left(\begin{smallmatrix} V_1^\top \widetilde{\alpha}_1\\ \widetilde{\alpha}_2\end{smallmatrix}\right) = \mathbf{R}\gamma,$$

so the canonical variates computed for $\mathbf{X}$ coincide with the ones computed for $\mathbf{R}$. □

Note that, according to the lemma, instead of working with large matrices $\widehat{\Sigma}_{XX} \in \mathbb{R}^{p\times p}$ and $\widehat{\Sigma}_{XY} \in \mathbb{R}^{p\times q}$ we can operate in terms of smaller matrices $\widehat{\Sigma}_{RR} \in \mathbb{R}^{(n+p_2)\times(n+p_2)}$ and $\widehat{\Sigma}_{RY} \in \mathbb{R}^{(n+p_2)\times q}$ thereby avoiding excessive computations.

3.5.3 Testing *PRCCA* on the Human Connectome data

First we chose the brain region of interest that we aim to release from the regularization penalty. To do so for each column of averaged activation data we compute *Cohen's d*, which measures the effect size for a one-sample t-test comparing the population mean to zero, and pick the regions with at least a medium effect ($d > 0.3$). The resulting 26 regions demonstrate the largest activation during the Gambling task. Then we run *PRCCA* on the averaged data imposing the penalty on all but these 26 regions. We consider the same grid of values $\lambda_1 = 10^{-3}, 10^{-2}, \ldots, 10^4, 10^5$ and choose the hyperparameter according to the maximum cross-validation score (see Section 3.3.4 for the details). The highest validation correlation is equal to 0.073 and is achieved when $\lambda_1 = 1$

(see Figure 3.4). Figure 3.5 represents the canonical coefficients computed for λ_1 chosen by cross-validation. As expected, the coefficients for most regions were shrunk to zero leaving only a few standing out. Again, it is the kernel trick which enables running cross-validation for the extremely high-dimensional feature matrix in just a few minutes. Although *PRCCA* does not perform well in this application, it will play an important role in developing subsequent methods.

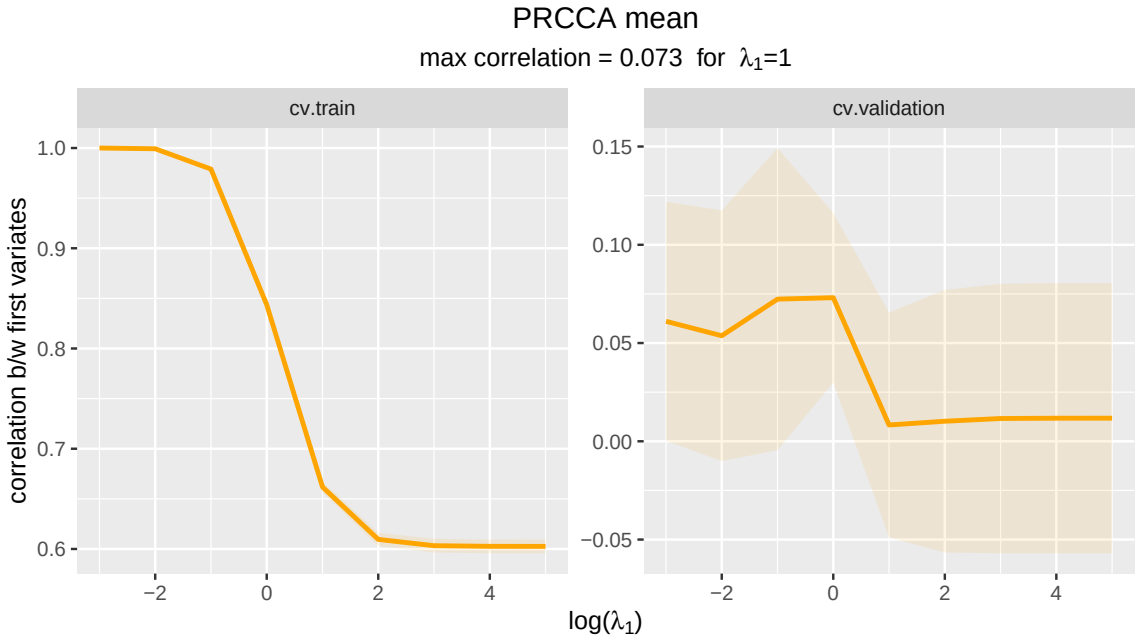


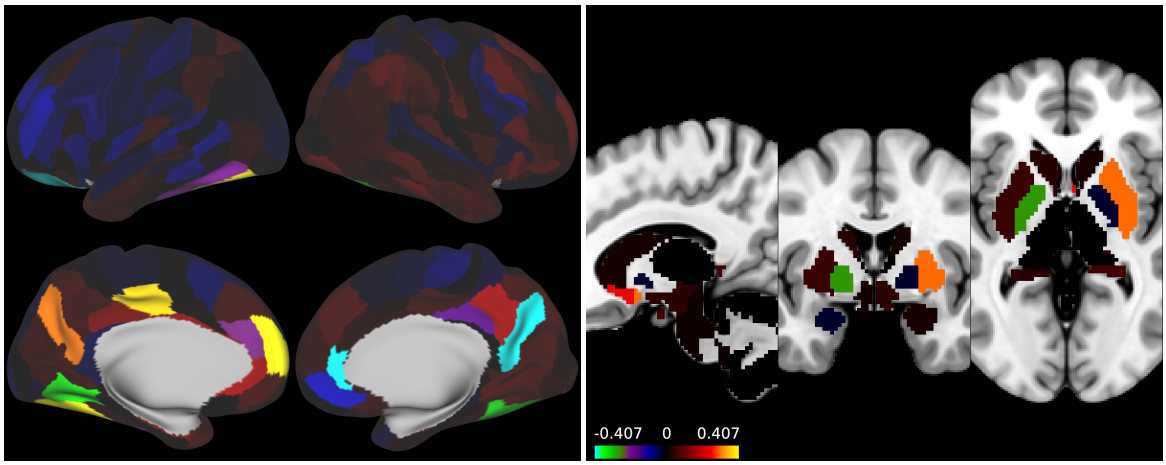
Figure 3.4: The cross-validation curves obtained via *PRCCA* with 10-fold cross-validation for the Human Connectome dataset. Left panel: (unpenalized) correlation between train canonical variates. Right panel: (unpenalized) correlation between validation canonical variates.

3.6 Group regularized canonical correlation analysis

3.6.1 Handling data with a group structure

The main critique of applying standard *RCCA* approach to the fMRI data is that, in fact, *RCCA* completely ignores the brain geometry treating all the features equally. Recall that in the Human Connectome data the features representing particular greyordinates are grouped into macro regions according to the function and anatomy. The goal of the *group regularized canonical correlation analysis (GRCCA)* is to incorporate this underlying data structure into the regularization penalty.

There are some group extensions of *CCA* based on elastic net and group lasso penalizations suggested in the literature (Chen et al., 2012; Lin et al., 2013). Unlike the existing methods the proposed *GRCCA* approach does not require an iterative algorithm and has a simple explicit solution.



(a) Cortical coefficients. (b) Subcortical coefficients.

Figure 3.5: *PRCCA* coefficients computed for the averaged data (229 features) with hyperparameter $\lambda_1 = 1$ chosen by cross-validation.

Equipped with the kernel trick it also allows working with data in a very high-dimensional feature space.

GRCCA solves the *CCA* problem under the following two natural assumptions. First, we assume *homogeneity of groups* and expect that the features within each group have approximately equal contribution to the canonical variates. In other words, the corresponding *CCA* coefficients do not vary significantly inside each group. Second, we assume the *differentiating sparsity on a group level* and expect that the coefficients will be shrunk toward zero all together for some groups. In terms of brain imaging applications these two assumptions mean that greyordinates "act in concert" within each macro region and that some regions have a weaker effect on the studied phenomenon.

To state the *GRCCA* optimization problem rigorously we need to introduce some further notation. Suppose that the elements of random vectors X and Y are known to come in K and L groups, respectively. For each $k = 1, \dots, K$ let p_k be the number of elements from X that belong to group k, so $\sum_{k=1}^{K} p_k = p$, and let $X_k \in \mathbb{R}^{p_k}$ be the random vector that consists of these elements. Without loss of generality, we can assume that $X = \begin{pmatrix} X_1 \\ \dots \\ X_K \end{pmatrix}$. Similarly, one can divide the *CCA* coefficients α into blocks corresponding to different groups, i.e. $\alpha = \begin{pmatrix} \alpha_1 \\ \dots \\ \alpha_K \end{pmatrix}$. If $\bar{\alpha}_k = \frac{\mathbf{1}^\top \alpha_k}{p_k}$ denotes the mean of the *CCA* coefficients in group k, then the group homogeneity assumption implies that all values of α_k do not deviate significantly from the mean value $\bar{\alpha}_k$, whereas, the differentiating sparsity on a group level implies that the average deviation of $\bar{\alpha}_k$ from zero is small. This can be characterized by the following two constraints: $\sum_{k=1}^{K} \|\alpha_k - \mathbf{1}\bar{\alpha}_k\|^2 \leq t_1$ and $\sum_{k=1}^{K} p_k \bar{\alpha}_k^2 \leq s_1$. Note that these two equations can be interpreted as bounds on within- and between- group variation, respectively. One can derive similar constraints for β coefficients as well. Adding all the constraints to the *CCA*

optimization problem we end up with the *GRCCA* optimization problem

$$\begin{aligned}\underset{\alpha\in\mathbb{R}^p,\ \beta\in\mathbb{R}^q}{\text{maximize}}\ \alpha^\top\widehat{\Sigma}_{XY}\beta \text{ subject to } \alpha^\top\widehat{\Sigma}_{XX}\alpha=1 \text{ and } \beta^\top\widehat{\Sigma}_{YY}\beta=1\\ \sum_{k=1}^{K}\|\alpha_k-\mathbf{1}\bar{\alpha}_k\|^2\le t_1 \text{ and } \sum_{\ell=1}^{L}\|\beta_\ell-\mathbf{1}\bar{\beta}_\ell\|^2\le t_2\\ \sum_{k=1}^{K}p_k\bar{\alpha}_k^2\le s_1 \text{ and } \sum_{\ell=1}^{L}q_\ell\bar{\beta}_\ell^2\le s_2. \qquad (3.8)\end{aligned}$$

Next, denote $C_m=\frac{\mathbf{1}\mathbf{1}^\top}{m}\in\mathbb{R}^{m\times m}$. Let $C_X=C_{p_1}\oplus\ldots\oplus C_{p_K}\in\mathbb{R}^{p\times p}$ refer to the block diagonal matrix with blocks $C_{p_1},\ldots,C_{p_K}$. Thus, the constraints on α can be rewritten in terms of a regularization penalty as

$$\begin{aligned}&\alpha^\top\widehat{\Sigma}_{XX}\alpha+\lambda_1\sum_{k=1}^{K}\|\alpha_k-\mathbf{1}\bar{\alpha}_k\|^2+\mu_1\sum_{k=1}^{K}p_k\bar{\alpha}_k^2=\\ &\alpha^\top\widehat{\Sigma}_{XX}\alpha+\lambda_1\sum_{k=1}^{K}\alpha_k^\top\left(I-C_{p_k}\right)\alpha_k+\mu_1\sum_{k=1}^{K}\alpha_k^\top C_{p_k}\alpha_k=\\ &\alpha^\top\left(\widehat{\Sigma}_{XX}+\lambda_1(I-C_X)+\mu_1C_X\right)\alpha=\alpha^\top\left(\widehat{\Sigma}_{XX}+K_X(\lambda_1,\mu_1)\right)\alpha.\end{aligned}$$

Here $K_X(\lambda_1,\mu_1)=\lambda_1(I-C_X)+\mu_1C_X$ is the penalty matrix. The constraints for β can be combined in a similar way leading to the *GRCCA* modified correlation coefficient

$$\rho_{GRCCA}(\alpha,\beta;\lambda_1,\mu_1,\lambda_2,\mu_2)=\frac{\alpha^\top\widehat{\Sigma}_{XY}\beta}{\sqrt{\alpha^\top(\widehat{\Sigma}_{XX}+K_X(\lambda_1,\mu_1))\alpha}\sqrt{\beta^\top(\widehat{\Sigma}_{YY}+\ K_Y(\lambda_2,\mu_2))\beta}}.$$

Note that this correlation coefficient has similar structure to the *RCCA* coefficient (3.4) and *PRCCA* coefficient (3.7), but now the covariance matrices in the denominator are adjusted by block diagonal matrices $K_X(\lambda_1,\mu_1)$ and $K_Y(\lambda_2,\mu_2)$.

Similar to *RCCA* and *PRCCA*, the explicit solution to the *GRCCA* problem can be found via the SVD of matrix $\left(\widehat{\Sigma}_{XX}+K_X(\lambda_1,\mu_1)\right)^{-\frac{1}{2}}\widehat{\Sigma}_{XY}\left(\widehat{\Sigma}_{YY}+K_Y(\lambda_2,\mu_2)\right)^{-\frac{1}{2}}$, which can be problematic in high dimensions. It turns out that there is a simple linear transformation that converts the *GRCCA* problem to an equivalent *RCCA*/*PRCCA* problem; but before building the bridge between these three methods we need to touch base on general regularization.

3.7 General regularized canonical correlation analysis

3.8 General penalty matrix

It turns out that all *RCCA*, *PRCCA* and *GRCCA* methods are similar in nature: they perform regularization by means of adjusting covariance matrices $\widehat{\Sigma}_{XX}$ and/or $\widehat{\Sigma}_{YY}$ in the denominator of the modified correlation coefficient. In this section we consider the class of *CCA* problems with general weighted ℓ_2 regularization.

If $K_X, K_Y \in \mathbb{R}^{p\times p}$ are some positive semi-definite *penalty matrices*, then the general modified correlation coefficient can be written as

$$\rho(\alpha,\beta;K_X,K_Y) = \frac{\alpha^\top \widehat{\Sigma}_{XY}\beta}{\sqrt{\alpha^\top(\widehat{\Sigma}_{XX}+K_X)\alpha}\,\sqrt{\beta^\top(\widehat{\Sigma}_{YY}+K_Y)\beta}}. \tag{3.9}$$

The accompanying *general RCCA* optimization problem is therefore

$$\begin{aligned}\underset{\alpha\in\mathbb{R}^p,\ \beta\in\mathbb{R}^q}{\text{maximize}}\ \alpha^\top\widehat{\Sigma}_{XY}\beta \text{ subject to } \alpha^\top\widehat{\Sigma}_{XX}\alpha = 1 \text{ and } \beta^\top\widehat{\Sigma}_{YY}\beta = 1\\ \alpha^\top K_X\alpha \le t_1 \text{ and } \beta^\top K_Y\beta \le t_2.\end{aligned} \tag{3.10}$$

Note that the inequality constraints in (3.10) can be rewritten as $\|\alpha\|_{K_X} \le t_1$ and $\|\beta\|_{K_Y} \le t_2$, where $\|\cdot\|_A$ is weighted Euclidean norm defined as $\|x\|_A = x^\top Ax$. The resulting canonical variates and coefficients can be found via the singular value decomposition of the matrix $\left(\widehat{\Sigma}_{XX}+K_X\right)^{-\frac{1}{2}}\widehat{\Sigma}_{XY}\left(\widehat{\Sigma}_{YY}+K_Y\right)^{-\frac{1}{2}}$. To handle *general RCCA* in high dimensions, one can link it to the two methods for which we already established the kernel trick.

Lemma 5 (General *RCCA* to *RCCA*/*PRCCA*)**.** *If both K_X and K_Y are positive definite then, by some proper change of basis, the general RCCA problem can be reduced to the RCCA one. Alternatively, if one of K_X and K_Y has zero eigenvalues then general RCCA boils down to solving the PRCCA problem with number of unpenalized coefficients equal to the multiplicity of the zero eigenvalue.*

Proof. As usual, we do not penalize Y part assuming $K_Y = 0$. Consider the eigendecomposition of matrix K_X, i.e. $K_X = UDU^\top$ with orthogonal $U \in \mathbb{R}^{p\times p}$ and diagonal $D \in \mathbb{R}^{p\times p}$. Since K_X is supposed to be a positive semi-definite matrix, D has non-negative diagonal elements. Applying transformation $\widetilde{\mathbf{X}} = \mathbf{X}U$ and $\widetilde{\alpha} = U^\top\alpha$, and using the same reasoning as in 3.3.3 and 3.5.2 we obtain

$$\begin{aligned}\alpha^\top\frac{\mathbf{X}^\top\mathbf{Y}}{n}\beta &= \widetilde{\alpha}^\top\frac{\widetilde{\mathbf{X}}^\top\mathbf{Y}}{n}\beta\\ \alpha^\top\left(\frac{\mathbf{X}^\top\mathbf{X}}{n}+K_X\right)\alpha &= \widetilde{\alpha}^\top\left(\frac{\widetilde{\mathbf{X}}^\top\widetilde{\mathbf{X}}}{n}+D\right)\widetilde{\alpha}\end{aligned}$$

thus the equivalent modified correlation coefficient in the new basis is

$$\rho(\widetilde{\alpha}, \beta; D) = \frac{\widetilde{\alpha}^\top \widehat{\Sigma}_{\widetilde{X}Y} \beta}{\sqrt{\widetilde{\alpha}^\top \left(\widehat{\Sigma}_{\widetilde{X}\widetilde{X}} + D\right) \widetilde{\alpha}} \sqrt{\beta^\top \widehat{\Sigma}_{YY} \beta}}.$$

Further, we decompose remaining diagonal matrix as $D = SLS$ as follows. We have two cases:

1. If all diagonal elements of D are positive then we put $S = D^{\frac{1}{2}}$ and $L = I$.

2. Suppose the first p_1 elements of D are positive and the rest $p_2 = p - p_1$ elements are zero. Let $D = \left(\begin{smallmatrix} D_{11} & 0 \\ 0 & 0 \end{smallmatrix}\right)$, where $D_{11} \in \mathbb{R}^{p_1 \times p_1}$ is the block containing all positive diagonal elements of matrix D. Then we can set

$$S = \left(\begin{smallmatrix} D_{11}^{\frac{1}{2}} & 0 \\ 0 & I \end{smallmatrix}\right) \quad \text{and} \quad L = \left(\begin{smallmatrix} I_{p_1} & 0 \\ 0 & 0 \end{smallmatrix}\right)$$

Note that, unlike D, matrix S is non-singular, so the change of basis $\widetilde{\widetilde{\mathbf{X}}} = \widetilde{\mathbf{X}} S^{-1}$ and $\widetilde{\widetilde{\alpha}} = S\widetilde{\alpha}$ is well-defined and leads to the following equalities

$$\widetilde{\alpha}^\top \frac{\widetilde{\mathbf{X}}^\top \mathbf{Y}}{n} \beta = \widetilde{\widetilde{\alpha}}^\top \frac{\widetilde{\widetilde{\mathbf{X}}}^\top \mathbf{Y}}{n} \beta$$
$$\widetilde{\alpha}^\top \left(\frac{\widetilde{\mathbf{X}}^\top \widetilde{\mathbf{X}}}{n} + D\right) \widetilde{\alpha} = \widetilde{\widetilde{\alpha}}^\top \left(\frac{\widetilde{\widetilde{\mathbf{X}}}^\top \widetilde{\widetilde{\mathbf{X}}}}{n} + L\right) \widetilde{\widetilde{\alpha}}.$$

Therefore, the equivalent modified correlation coefficients

$$\rho(\widetilde{\widetilde{\alpha}}, \beta; L) = \frac{\widetilde{\widetilde{\alpha}}^\top \widehat{\Sigma}_{\widetilde{\widetilde{X}}\widetilde{\widetilde{X}}} \widetilde{\widetilde{\alpha}}}{\sqrt{\widetilde{\widetilde{\alpha}}^\top \left(\widehat{\Sigma}_{\widetilde{\widetilde{X}}\widetilde{\widetilde{X}}} + L\right) \widetilde{\widetilde{\alpha}}} \sqrt{\beta^\top \widehat{\Sigma}_{YY} \beta}}.$$

Is it easy to see that in the first case when $L = I$ the above correlation coefficient coincides with the *RCCA* correlation coefficient with $\lambda_1 = 1$. Alternatively, if $L = \left(\begin{smallmatrix} I_{p_1} & 0 \\ 0 & 0 \end{smallmatrix}\right)$ it is equal to the *PRCCA* correlation coefficient with $\lambda_1 = 1$ and p_2 unpenalized coefficients. Thus, we conclude that the General *RCCA* solution computed for X and Y can be found by means of either *RCCA* or *PRCCA* method applied to $\widetilde{\widetilde{X}}$ and Y. The canonical variates stay the same regardless the basis as $\widetilde{\widetilde{\mathbf{X}}}\widetilde{\widetilde{\alpha}} = \widetilde{\mathbf{X}}\widetilde{\alpha} = \mathbf{X}\alpha$. The corresponding inverse transform for the coefficients is $\alpha = SU\widetilde{\widetilde{\alpha}}$. □

3.8.1 Link to the flexibility vs. performance trade-off

There are several important properties of the proposed penalty matrix explaining the motivation for the *GRCCA* method. First, one can show that for $K_X(\lambda_1, \lambda_1) = \lambda_1 I$, so *RCCA* is the special case of *GRCCA*. Second, increasing λ_1 restrains the variability of coefficients within each brain region

and, in limit, makes all the coefficients that belong to the same brain region equal to each other (and equal to the region mean). This is essentially equivalent to replacing features in each brain region by the average. Therefore, when $\lambda_1 \to \infty$ the *GRCCA* problem becomes equivalent to the *RCCA* problem solved for the reduced data (see Section 3.4.1 for the details). To sum up, varying λ_1 and μ_1 allows us to approach the *RCCA* method conducted for either full or reduced data thereby controlling the flexibility vs interpretability trade-off described in Section 3.4.2.

3.9 Link between *GRCCA* and *RCCA/PRCCA*

In this section we discuss how one can reduce the *group regularized CCA* problem to either *RCCA* or *PRCCA*. This link can be subsequently used to establish the kernel trick for group-structured data thereby reducing computations in high dimensions.

3.9.1 Approach 1: via the SVD of the penalty matrix

Since *GRCCA* is just a special case of general *RCCA*, one can use previous lemma to map the *GRCCA* problem to either *RCCA* or *PRCCA* and, subsequently, find the canonical coefficients via the kernel trick. However, to find this transformation it is required to do an extra step: the eigendecomposition of the kernel matrices $K_X(\lambda_1, \mu_1)$ and $K_Y(\lambda_2, \mu_2)$. Although this can be infeasible in high dimensions for general penalty matrices it turns out that one can use the specific structure of the *GRCCA* penalty matrix to get around this eigendecomposition.

We again assume that the regularization was imposed on the X part only (i.e. $\lambda_2 = \mu_2 = 0$); however, it is not difficult to derive similar results for the general case. Recall that matrix $C_m = \frac{\mathbf{1}\mathbf{1}^\top}{m} \in \mathbb{R}^{m \times m}$ has

- unit eigenvalue with corresponding eigenvector $\frac{\mathbf{1}}{\sqrt{m}} \in \mathbb{R}^m$,
- zero eigenvalue with corresponding eigenspace $\left[\frac{\mathbf{1}}{\sqrt{m}}\right]^\perp \in \mathbb{R}^{m \times (m-1)}$.

Here $[A]^\perp$ refers to the orthogonal complement of matrix A. The resulting eigendecomposition is therefore

$$C_m = U_m \left(\begin{smallmatrix} 1 & 0 \\ 0 & 0 \end{smallmatrix}\right) U_m^\top \quad \text{with} \quad U_m = \left(\tfrac{\mathbf{1}}{\sqrt{m}} \left[\tfrac{\mathbf{1}}{\sqrt{m}}\right]^\perp \right).$$

It is easy to show the following eigendecomposition as well

$$\lambda_1(I - C_m) + \mu_1 C_m = U_m D_m U_m^\top \quad \text{with} \quad D_m = \begin{pmatrix} \mu_1 & 0 \\ 0 & \lambda_1 I_{m-1} \end{pmatrix}.$$

Thus the penalty matrix $K_X(\lambda_1, \mu_1)$ can be decomposed as

$$K_X(\lambda_1, \mu_1) = UDU^\top \quad \text{with} \quad U = U_{p_1} \oplus \ldots \oplus U_{p_K} \quad \text{and} \quad D = D_{p_1} \oplus \ldots \oplus D_{p_K}.$$

Using the lemma from Section 3.7 we conclude that if $\lambda_1, \mu_1 > 0$ then the *GRCCA* problem can be solved via the *RCCA* approach. Alternatively, if $\lambda_1 = 0$ or $\mu_1 = 0$ it can be reduced to the *PRCCA* problem with $n - K$ and K unpenalized coefficients, respectively. Hereafter we will assume $\lambda_1 > 0$, i.e. the presence of group homogeneity.

Note that due to the specific structure of $K_X(\lambda_1, \mu_1)$ the eigendecomposition can be calculated block-wise. Moreover, the resulting computational cost is equal to the cost of computing the orthogonal complements $\left[\frac{\mathbf{1}}{\sqrt{p_1}}\right]^\perp, \ldots, \left[\frac{\mathbf{1}}{\sqrt{p_K}}\right]^\perp$, which can be efficiently found via, for example, Helmert contrasts (c.f. `contr.helmert()` function in R) and without computing any eigendecomposition.

3.9.2 Approach 2: via the feature matrix extension

In Section 3.9 we already proposed the way to solve the *GRCCA* problem. If the number of groups K is rather small then penalty matrix $K_X(\lambda_1, \mu_1)$ will consist of a few large blocks. In this case it would be quite expensive to compute the orthogonal complements and to further apply the linear transformation mapping *general RCCA* to *RCCA/PRCCA*. It turns out that there is an alternative linear transformation that leads to equivalent *RCCA/PRCCA* problem while being less expensive.

We need to establish some notation first. Suppose that matrix $\mathbf{X}$ is divided into blocks according to groups, i.e. $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_K)$. Denote the column average matrix for group k by $\bar{\mathbf{X}}_k = \mathbf{X}_k \frac{\mathbf{1}}{p_k}$ and consider an extended matrix $\widetilde{\mathbf{X}}$ that consists of (scaled) mean centered blocks and extra K columns corresponding to (scaled) group means

$$\widetilde{\mathbf{X}}(a, b) = \left(\sqrt{\frac{1}{a}} \left(\mathbf{X}_1 - \bar{\mathbf{X}}_1 \right), \sqrt{\frac{p_1}{b}} \bar{\mathbf{X}}_1, \ldots, \sqrt{\frac{1}{a}} \left(\mathbf{X}_K - \bar{\mathbf{X}}_K \right), \sqrt{\frac{p_K}{b}} \bar{\mathbf{X}}_K \right) \in \mathbb{R}^{n \times (p+K)}.$$

Lemma 6 (*GRCCA* to *RCCA/PRCCA*)**.** *If $\mu_1 > 0$ the GRCCA problem for $\mathbf{X}$ and $\mathbf{Y}$ can be reduced to solving the RCCA problem for $\widetilde{\mathbf{X}}(\lambda_1, \mu_1)$ and $\mathbf{Y}$. If $\mu_1 = 0$, then GRCCA boils down to the PRCCA problem for $\widetilde{\mathbf{X}}(\lambda_1, 1)$ and $\mathbf{Y}$ with K unpenalized coefficients.*

Proof. Let us prove the statement for $K = 1$, one can easily extend the proof for arbitrary K. For $K = 1$ the penalty matrix is

$$K_X(\lambda_1, \mu_1) = \lambda_1 \left(I - \frac{\mathbf{1}\mathbf{1}^\top}{n} \right) + \mu_1 \frac{\mathbf{1}\mathbf{1}^\top}{n}$$

which can be decomposed as

$$K_X(\lambda_1, \mu_1) = \widetilde{U} \widetilde{D} \widetilde{U}^\top \quad \text{with} \quad \widetilde{U} = \left(I - \frac{\mathbf{1}\mathbf{1}^\top}{n} \frac{\mathbf{1}}{\sqrt{n}} \right) \quad \text{and} \quad \widetilde{D} = \begin{pmatrix} \lambda_1 I_n & 0 \\ 0 & \mu_1 \end{pmatrix}.$$

This decomposition can be considered as an alternative to the eigendecomposition $K_X = UDU^\top$ discussed in the previous section. However, unlike matrix U which was square and orthogonal, $\widetilde{U}$ is a rectangular $n \times (n + 1)$ matrix with orthogonal rows, i.e. $\widetilde{U}\widetilde{U}^\top = I$.

Further, similar to the previous section, we do decomposition $\widetilde{D} = \widetilde{S}\widetilde{L}\widetilde{S}$. We again have two cases here. If $\mu_1 > 0$ then we have

$$\widetilde{S} = \begin{pmatrix} \sqrt{\lambda_1} I_n & 0 \\ 0 & \sqrt{\mu_1} \end{pmatrix} \text{ and } \widetilde{L} = I$$

and the following relation is true

$$\mathbf{X}\widetilde{U}\widetilde{S}^{-1} = \left(\sqrt{\tfrac{1}{\lambda_1}} \left(\mathbf{X} - \bar{\mathbf{X}} \right), \sqrt{\tfrac{n}{\mu_1}} \bar{\mathbf{X}} \right) = \widetilde{\mathbf{X}}(\lambda_1, \mu_1).$$

If, on the contrary, $\mu_1 = 0$ then

$$\widetilde{S} = \begin{pmatrix} \sqrt{\lambda_1} I_n & 0 \\ 0 & 1 \end{pmatrix} \text{ and } \widetilde{L} = \begin{pmatrix} I_n & 0 \\ 0 & 0 \end{pmatrix}$$

which leads us to

$$\mathbf{X}\widetilde{U}\widetilde{S}^{-1} = \left(\sqrt{\tfrac{1}{\lambda_1}} \left(\mathbf{X} - \bar{\mathbf{X}} \right), \sqrt{n} \bar{\mathbf{X}} \right) = \widetilde{\mathbf{X}}(\lambda_1, 1).$$

Note that the proof in Section 3.7 requires matrix $\widetilde{U}$ to have orthogonal rows only. Thus, following this proof, one can conclude that the original *GRCCA* problem is equivalent to the *RCCA* problem solved for $\widetilde{\mathbf{X}}(\lambda_1, \mu_1)$ and $\mathbf{Y}$ if $\mu_1 > 0$. Alternatively, if $\mu_1 = 0$ then *GRCCA* boils down to solving the *PRCCA* problem for $\widetilde{\mathbf{X}}(\lambda_1, 1)$ and $\mathbf{Y}$, and one unpenalized coefficient. Again, such change of basis does not influence the canonical variates, whereas the canonical coefficients are transformed according to $\alpha = \widetilde{U}\widetilde{S}^{-1}\widetilde{\alpha}$. □

Note that the corresponding data transformation boils down to computing the group means and adjusting the feature matrix by group means. Although this approach can be considered as convenient alternative to the one suggested in Section 3.9, there is a trade-off. On the one hand, we reduce the cost by getting around the eigendecomposition (computing the orthogonal complements). On the other hand, we increase the feature matrix dimension from p to $p + K$, which can be quite inefficient for large K.

3.9.3 Analysis of the Human Connectome dataset via *GRCCA*

In this section we apply the *GRCCA* method to the Human Connectome study data grouping activation features according to the brain regions. We again adjusted X and Y for the sex effect and ran 10-fold cross-validation on the adjusted data pair with the penalty factors varying in the grid $\lambda_1, \mu_1 = 10^{-3}, 10^{-2}, \ldots, 10^4, 10^5$ (see Section 3.3.4 for the details). The resulting cross-validation curves are presented in Figure 3.6. According to the plot, the highest cross-validation score is attained for $\lambda_1 = 100$ and $\mu_1 = 1$ leading to the correlation of 0.296. To validate these significant findings, we again run nested cross-validation (see Section 3.4.1). According to Figure 3.7, the NCV average test set score is equal to 0.236. Thus, although slightly optimistic (by a modest value of 0.06), the *GRCCA* cross-validation correlation is not a spurious finding and is a significant improvement comparing to the *RCCA* method.

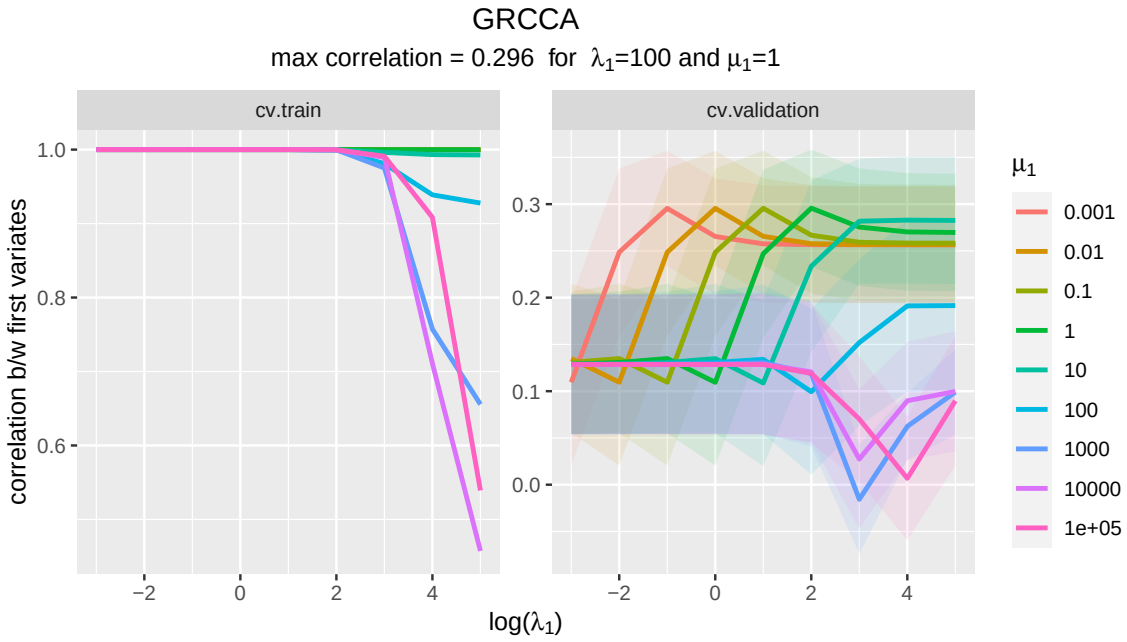


Figure 3.6: The cross-validation curves obtained via *GRCCA* with 10-fold cross-validation for the Human Connectome Project dataset. Left panel: (unpenalized) correlation between train canonical variates. Right panel: (unpenalized) correlation between validation canonical variates.

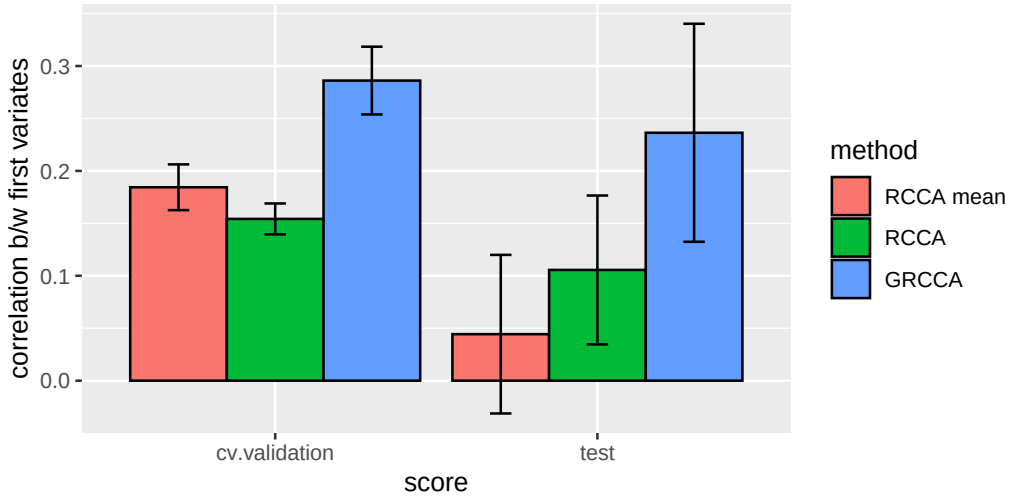


Figure 3.7: Nested cross-validation scores computed for three models: *RCCA* fitted on the averaged data (229 features), *RCCA* fitted on the original data (90 368 features), *GRCCA*. Two scores are reported: *cv.validation* = maximum score obtained via 10-fold cross-validation, averaged across 11 NCV folds; *test* = score computed on independent test set, averaged across 11 NCV folds.

In addition to better performance, the *GRCCA* method allows us to track the effect of the variation inside each brain region (controlled by the λ_1 penalty factor) on the resulting canonical correlation separately from the effect of the variation across the regions (controlled by μ_1 penalty factor). For example, the spikes for small μ_1 values suggest that it is more beneficial to reduce within group variation than between group one. In other words, shrinking all brain region coefficients toward the group means improves the performance more than shrinking group means toward zero. Moreover, for small μ_1 it is the ratio of hyperparameters that plays the key role: the highest score is always achieved when $\frac{\lambda_1}{\mu_1} = 100$. For large μ_1 this pattern disappears as we start to over-penalize both between and within group variations.

3.9.4 Using GRCCA for visualization

In this section we demonstrate another advantage of *GRCCA* in the context of visualization and interpretability. In Figure 3.8 we present the coefficient paths (α vs. λ_1) produced by the *RCCA* method as well as the group modification. Here different colors represent different brain regions. According to the plot, we observe the following behavior of the coefficients. For the *RCCA* method the canonical coefficients are shrunk toward zero all together with the growth of λ_1. On the contrary, for large λ_1 all the *GRCCA* coefficient paths become horizontal, which implies the convergence of the coefficients to the group means. Finally, increasing μ_1 shrinks all the group means toward zero encouraging differentiating sparsity on a group level.

In Figure 3.9 we present the brain images computed for $\mu_1 = 1$ and $\lambda_1 = 1, 10, 100, 1000$. Note that larger λ_1 makes the brain region pattern more obvious (similar to Figures 3.3a-3.3b). Moreover, for $\lambda_1 = 100$ we get the plot corresponding to the best *GRCCA* model from Section 3.9.3. Thus the *GRCCA* model chosen by cross-validation has not only better performance on the validation set than the *RCCA* model, but it is also more interpretable in the context of the importance of each brain region. Specifically, the canonical component had especially high positive loadings in subcortical regions involved in reward processing, such as the striatum (nucleus accumbens, putamen) and thalamus (Haber, 2017). It also loaded positively on a cortical network encompassing the temporal lobe, dorsolateral prefrontal, dorsomedial prefrontal, posterior cingulate and precentral cortices (see Figure 3.9 for an annotated visualization of these results). Most of these regions have been shown to be connected to the striatum and to be part of key reward-processing pathways as well (Haber, 2017).

3.10 Simulation study

In this section we set up a small simulation experiment where we compare performance of all the above methods on the data with group structure. We generate the data as follows. For random vector X we assume that it is grouped into K groups of equal size, thereby having $p_k = \frac{p}{K}$ variables

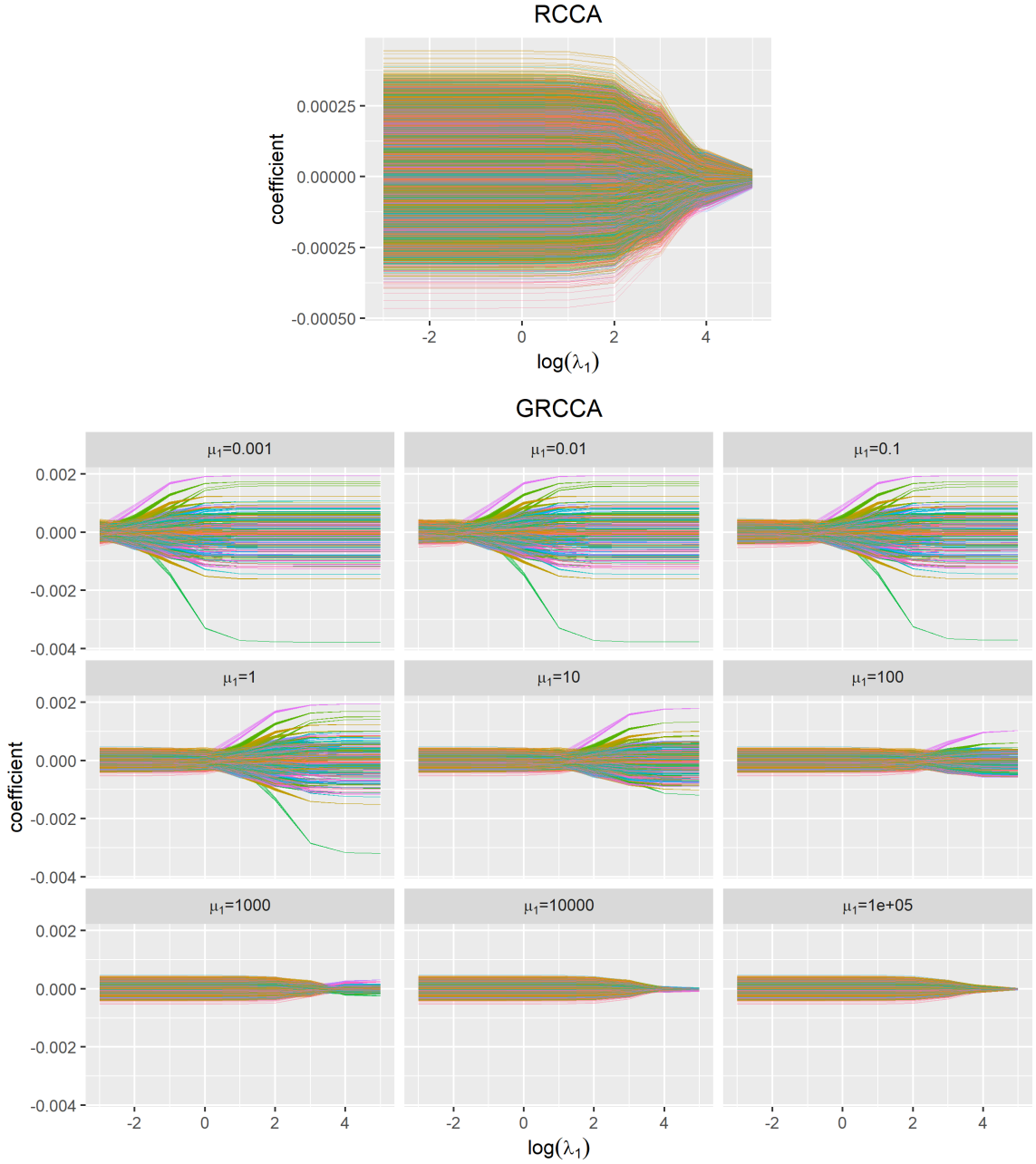


Figure 3.8: Coefficient paths for *RCCA* and *GRCCA* with colors representing the regions. The *RCCA* method shrinks canonical coefficients toward zero all together with the growth of λ_1. The *GRCCA* method shrinks them toward the group means with the growth of λ_1, whereas increasing μ_1 shrinks all the group means toward zero.

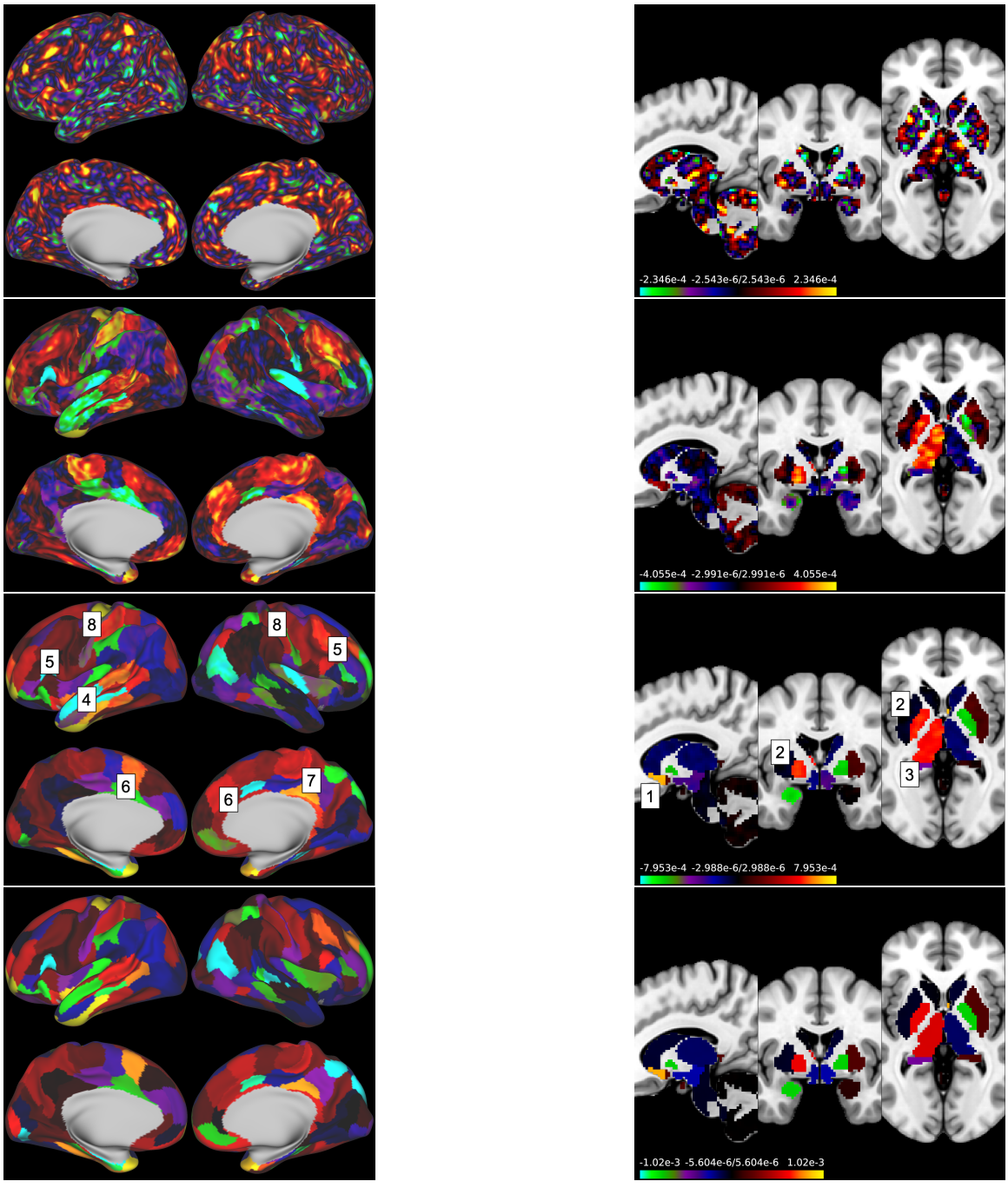


Figure 3.9: From top to bottom: the *GRCCA* coefficients for $\mu_1 = 1$ and $\lambda_1 = 1, 10, 100, 1000$. The third row represents the solution produced by the cross-validation procedure. Annotation of brain regions: [1] nucleus accumbens, [2] putamen, [3] thalamus, [4] temporal lobe, [5] dorsolateral prefrontal cortex, [6] dorsomedial prefrontal cortex, [7] posterior cingulate cortex, [8] precentral cortex.

in group k. Each group of X is generated by one of K centroid random variables and is obtained by adding some Gaussian noise to the centroid. Moreover, we assume the presence of some correlation between the centroids and Y. To be precise, to generate the data we exploit the multivariate normal distribution as a joint distribution of random vector $Y \in \mathbb{R}^q$ and random vector of centroids $X^c \in \mathbb{R}^K$:

$$(Y, X^c) \sim \mathcal{N}_{q+K}(0, \Sigma) \quad \text{with} \quad \Sigma = \begin{pmatrix} I_q & \mathbf{1}\mathbf{1}^\top \sigma_{XY}^2 \\ \mathbf{1}\mathbf{1}^\top \sigma_{XY}^2 & I_K \end{pmatrix}.$$

Next, we generate random vector $X_k \in \mathbb{R}^{p_k}$ corresponding to groups $k = 1, \ldots, K$ from the distribution $X_k | X_k^c \sim \mathcal{N}_{p_k}(\mathbf{1}X_k^c, \sigma_X^2 I)$, where X_k^c is the k-th component of the centroid vector. Finally, we obtain matrices $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_K) \in \mathbb{R}^{n \times p}$ and $\mathbf{Y} \in \mathbb{R}^{n \times q}$ by drawing n samples from the above distributions. In our experiments we use $n = 10$, $p = 15$ and $q = 3$, the number of groups is $K = 5$. We set $\sigma_X = 1$ and test two settings: $\sigma_{XY} = 0.5$ for correlated data and $\sigma_{XY} = 0$ for independent data.

As the next step, we run *RCCA*, *PRCCA* and *GRCCA* on the generated data imposing the regularization on the X part only and using the following hyperparameters. For all methods the penalty factor is chosen to be $\lambda_1 = 10^{-5}, 10^{-4}, \ldots, 10^4, 10^5$. For *PRCCA* we penalize $p_1 = 10$ variables only leaving $p_2 = 5$ variables untouched; these five unpenalized variables correspond to the first features in each group. For *GRCCA* we again try two versions. First we run *GRCCA* with $\mu_1 = 0$, i.e. without differentiating sparsity on a group level assumption. Next we add differentiating sparsity to *GRCCA* and vary the penalty factor in the range $\mu_1 = 10^{-4}, 10^{-3}, \ldots, 1, 10$.

We compare all methods in terms of resulting correlations. For this purpose, we generate 1000 train and test sets, fit models on train and evaluate canonical correlation value on test. We plot average train and test correlations as well as their one standard error intervals vs. penalty factor λ_1 (see Fig. 3.10). According to the plot, for the correlated data the best test score is achieved by sparse version of *GRCCA*, which significantly outperforms *RCCA*. Better performance can be explained by the presence of the groups structure in the data. Note that non-sparse *GRCCA* also looses in terms of the test score. The possible reason is that number of observations ($n = 10$) is only twice as large as the number of groups ($K = 5$), so regularization on a group level helps to prevent overfitting to train data. In the case of independent data, all the competitors perform in a similar way: the average test correlation is very close to zero regardless the hyperparameter value; the test correlation curves are almost flat.

Finally, for each model we pick the value of λ_1 according to the maximum test score and compare the *CCA* coefficients α for the chosen models. Figure 3.11 displays the main difference between the *RCCA* and *GRCCA* methods. Although both techniques aim to reduce the data dimensionality, the reduction is achieved in a different way. Unlike *RCCA*, which treats all the coefficients equally and confines their deviation from zero, *GRCCA* carries out the reduction treating equally the coefficients inside each group and removing the within group noise. To sum up, in the presence of a group structure the group modification of *RCCA* allows for dimensionality reduction in a more efficient

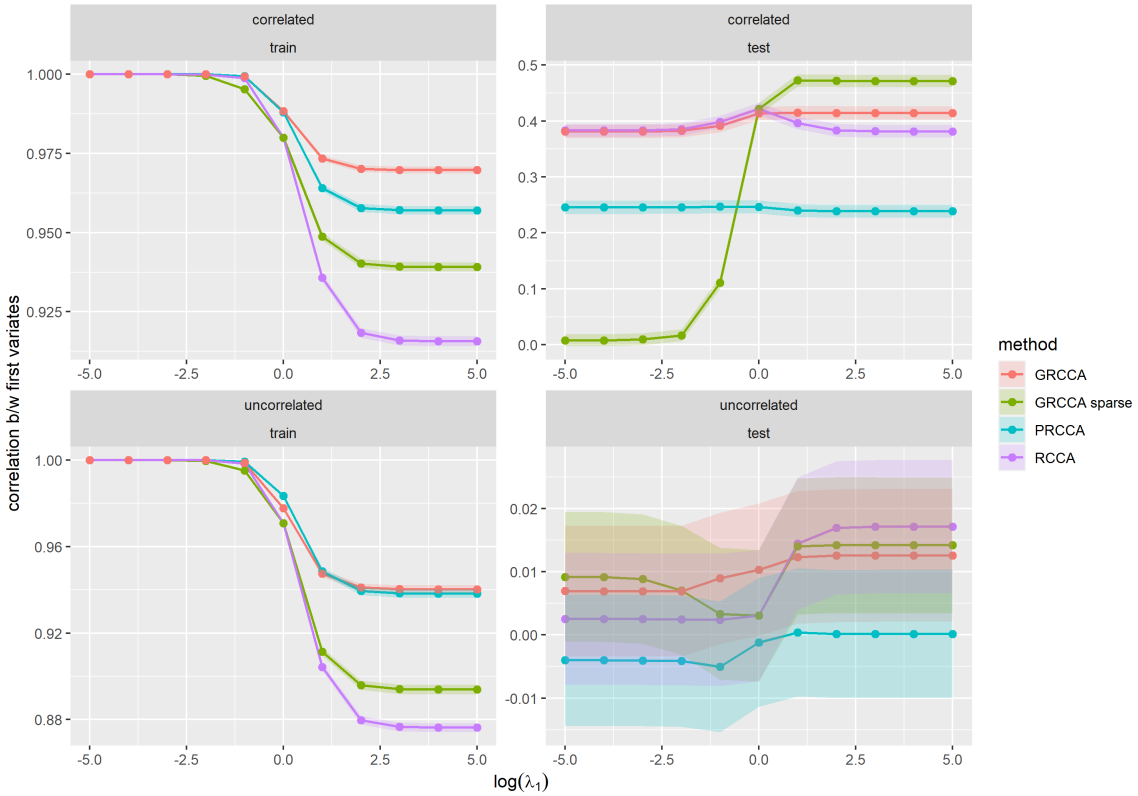


Figure 3.10: Train and test curves computed via simulation. Four models presented: *RCCA*, *PRCCA* and *GRCCA* with zero ($\mu_1 = 0$) and non-zero sparsity ($\mu_1 = 1$). First row: train and test correlation obtained for data with correlation ($\sigma_{XY} = 0.5$). Second row: train and test correlation obtained for uncorrelated data ($\sigma_{XY} = 0$).

and interpretable way.

3.11 Discussion

In this chapter we proposed several approaches to the *CCA* regularization. The introduced *PRCCA* technique has a similar flavor as *RCCA*, but it penalizes only a subset of canonical coefficients. Both of these methods combined with the proposed kernel trick allows us to find the *CCA* solution even in case of extremely high data dimensionality. We further present the *GRCCA* method, which is based on the underlying group structure of the data and which, therefore, can be useful in some applications, and extend regularization to the case of a more general regularization penalty thereby proposing *general RCCA*. The close connection between the latter techniques with *RCCA* and *PRCCA* methods enables to utilize the kernel trick in the general case thus providing a powerful tool for regularizing *CCA* in the high-dimensional framework.

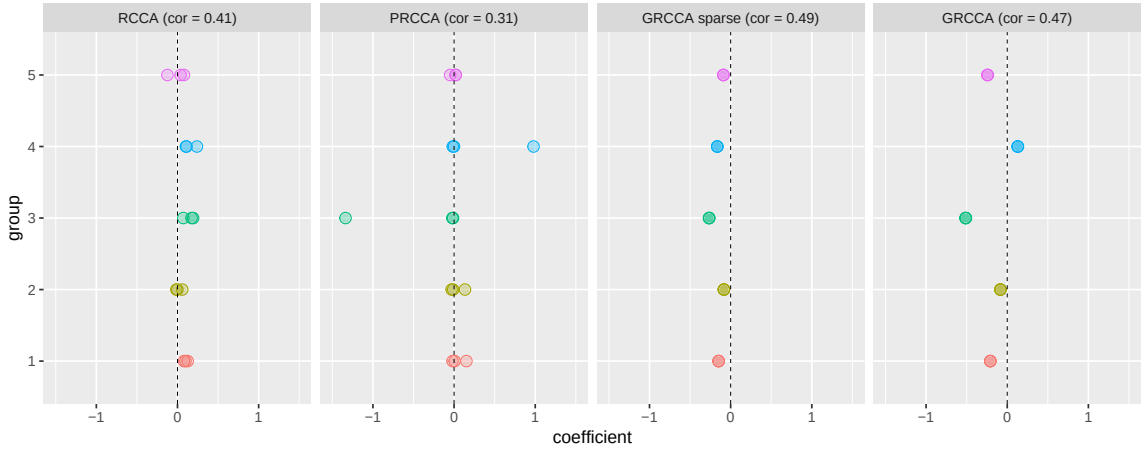


Figure 3.11: Coefficient values obtained via *RCCA*, *PRCCA* and two versions of *GRCCA* with hyperparameter λ_1 chosen to maximize the test correlation. In this plot color corresponds to the group number.

There is still much scope for future work. One interesting direction for further research is to consider other applications of the proposed group *RCCA* technique. For example, there are many problems in genetics where genes are grouped by functional similarity. Further, in this chapter we cover only two types of penalties: partial and group; although the proposed kernel trick can handle any ℓ_2-type penalty (see Section 3.7 for *general RCCA*). Thus we can study other structured modifications of *RCCA* that can be beneficial for applications. As an example, it may be interesting to explore hierarchical group structure, where not only brain loci are combined in some regions, but also regions are combined in some groups (e.g. we have cortical and subcortical groups in the HCP study).

From the computational point of view, it would be useful to investigate how one can optimize the choice of the hyperparameters. The following idea is inspired by ridge regression, which also uses the ℓ_2 penalty. Note that currently it is not necessary to apply any data normalization before running regularized *CCA* (all the fMRI features have the same scale). However, the overall scale of X influences the choice of the hyperparameters, i.e. multiplying X by some number a implies increasing the penalty factors by a^2 times as well. Therefore, it would be beneficial to develop some recommendations for the grid of hyperparameters the user should search through. For instance, we can use the ridge regression heuristic and introduce a concept of degrees-of-freedom (aka *df*) for *CCA* with regularization. Then we can base the hyperparameter recommendation on the *df* value.

3.12 Appendix

3.12.1 Neuroimaging analysis

Details about the protocol and measures collected by HCP-DES are outlined in Tozzi et al. (2020). Here, only the details relevant to this study are discussed.

Neuroimaging acquisition details Images were acquired at the Stanford Center for Cognitive and Neurobiological Imaging (CNI) on a GE Discovery MR750 3 T scanner using a Nova Medical 32-channel head coil. Two spin-echo fieldmaps were acquired at the beginning of each session, one with a posterior-anterior phase encoding direction, the other with an anterior-posterior direction. All fMRI scans were conducted using a blipped-CAIPI simultaneous multislice "multiband" acquisition (Setsompop et al., 2012).

1. Spin-echo fieldmaps: TE = 55.5 ms, TR = 6 s, FA = 90°, acquisition time = 18 s, field of view = 220.8 × 220.8 mm, 3D matrix size = 92 × 92 × 60, slice orientation = axial, angulation to anterior commissure - posterior commissure (AC-PC) line, phase encoding = AP and PA, receiver bandwidth = 250 kHz, readout duration = 49.14 ms, echo spacing = 0.54 ms, voxel size = 2.4 mm isotropic.

2. Single-band calibration: TE = 30 ms, TR = 4.4 s, FA = 90°, acquisition time = 13 s, field of view = 220.8 × 220.8 mm, 3D matrix size = 92 × 92 × 60, slice orientation = axial, angulation to AC-PC line, phase encoding = PA, receiver bandwidth = 250 kHz, readout duration = 49.14 ms, echo spacing = 0.54 ms, number of volumes = 4, voxel size = 2.4 mm isotropic.

3. Multiband fMRI: TE = 30 ms, TR = 0.71 s, FA = 54°, acquisition time = 3:44 (Gambling task), field of view = 220.8 × 220.8 mm, 3D matrix size = 92 × 92 × 60, slice orientation = axial, angulation to AC-PC line, phase encoding = PA, receiver bandwidth = 250 kHz, readout duration = 49.14 ms, echo spacing = 0.54 ms, number of volumes for Gambling task = 316, multiband factor = 6, calibration volumes = 2, voxel size = 2.4 mm isotropic.

4. T1-weighted: TE = 3.548 ms, MPRAGE TR = 2.84 s, FA = 8°, acquisition time = 8:33, field of view = 256 × 256 mm, 3D matrix size = 320 × 320 × 230, slice orientation = sagittal, angulation to AC-PC line, receiver bandwidth = 31.25 kHz, fat suppression = no, motion correction = PROMO, voxel size = 0.8 mm isotropic.

5. T2-weighted: TE = 74.4 ms, TR = 2.5 s, FA = 90°, acquisition time = 5:42, field of view = 240 × 240 mm, 3D matrix size = 320 × 320 × 216, slice orientation = sagittal, angulation to AC-PC line, receiver bandwidth = 125 kHz, fat suppression = no, motion correction = PROMO, voxel size = 0.8 mm isotropic.

Gambling task paradigm The HCP-DES adopts a version of the HCP Gambling task modified to allow comparison of small and large gain and loss outcomes (Somerville et al., 2018; Tozzi et al., 2020). A question mark is displayed on the screen and the participant must guess whether a number is greater than or less than five (and indicate their answer via button presses). If the participant identifies correctly, they win money, and if they guess incorrectly, they lose money. At the end of the task, 5 trials are randomly selected and summed together to determine the participant's payment.

Preprocessing Raw image files were converted to BIDS format and preprocessed using fMRIPrep (Esteban et al., 2019). Briefly, brain surfaces were reconstructed using recon-all (FreeSurfer 6.0.1; Dale et al., 1999). Susceptibility distortion for fMRI data were corrected using the two echo-planar imaging (EPI) references with opposing phase-encoding directions (Cox and Hyde, 1997). Surface data was registered to fsaverage space and subcortical data to MNI space. These were then merged to grayordinate CIFTI files. Automated labeling of noise components following ICA decomposition was performed using AROMA (Pruim et al., 2015). As final output, we down-sampled the preprocessed grey-ordinate functional CIFTI files to 32k FSLR space (Glasser et al., 2013). Then, we applied a 4 mm full-width half-maximum smoothing constrained to the grey matter boundaries. For the quantification of brain responses to the task, the following conditions were convolved with a canonical hemodynamic response function as implemented in FSL (Jenkinson et al., 2012): high win, low win, high loss, low loss, high cue, low cue. The regressors obtained were entered in a design matrix together with the confound regressors generated by AROMA. A GLM analysis was then performed using the HCP pipelines (Glasser et al., 2013) and the contrast win $>$ loss was estimated for each participant (coefficients of high and low wins and losses were averaged). The z-scores corresponding to this contrast in each greyordinate were the features entered in the CCA analyses.

Chapter 4

Multi-period forecasting and COVID-19 pandemic

4.1 Introduction and background

Time series forecasting techniques are used to predict events that occur over time by analyzing trends and patterns in past data. They are widely applicable across many fields of including finance, economics, politics, sports, meteorology and epidemiology. The latter area became especially important since the beginning of the COVID-19 pandemic in December 2019.

Several time-series forecasting techniques have been proposed in the literature. Standard statistical methods based on regressive models such as *autoregressive* (*AR*), *moving average* (*MA*), *autoregressive moving average* (*ARMA*), *autoregressive integrated moving average* (*ARIMA*) have been commonly used to forecast time-series (Box and Jenkins, 1976). These Box-Jenkins methods are particularly efficient when applied to a linear stationary time series; they can accommodate the non-linear case by applying some appropriate transformation first. More recent approaches are based on machine learning methods, in particular, artificial neural networks (Zhang et al., 1998; Nesreen et al., 2010; Makridakis et al., 2018). Compared to the *ARIMA*-type models, these often demonstrate better performance in forecasting non-linear signals.

The standard application of these techniques aims to predict the signal for a single forecast horizon (or "ahead"), most often one-step-ahead. However, in some applications such as epidemiology, where decisions are often based on the future trend of signal, simultaneous forecasts for multiple aheads can be of great interest. One of the popular methods for predicting several ahead values is *multi-stage prediction* (*MSP*) (Chen et al., 2004) or *multi-period forecasting* (*MFP*). This approach is usually based on a single output model which is applied recursively, i.e. the predicted value of the signal three weeks ahead is determined based on the already-produced predicted values for one and

two weeks ahead. The main disadvantages of such an iterative procedure is error propagation. An alternative method suggested in the literature is called the *multiple-input multiple-output approach* (*MIMO*), which aims to predict a vector of future values all at once (Bontempi, 2008; Ben Taieb et al., 2010). Detailed comparisons between different *MIMO* techniques can be found in Cheng et al. (2006) and An and Anh (2015).

In this study we introduce a novel approach for predicting multiple ahead values simultaneously which is based on the idea that the future signal can be well-approximated by a smooth curve. The rest of the paper is organized as follows. In Section 4.2 we introduce the general multi-period forecasting problem. In Sections 4.3.1–4.3.2 we describe two regression-based approaches for solving:

- a simple baseline method that predicts all aheads independently of each other (often termed "direct" forecasting);
- and a novel *MPF* method that enforces smoothness across aheads.

We extend the methodology to the case that some of the response signals are unobserved in Section 4.4 and propose an analogue based on quantile-regression in Section 4.7. Sections 4.5, 4.6 and 4.8 illustrate the *MPF* technique on a small simulation example as well as real COVID-19 case incidence data obtained from the Delphi Epidata CovidCast API (Reinhart et al., 2021). We conclude the paper with a Discussion where we suggest some future research directions.

4.2 Forecasting problem

In this section we state the general multi-period forecasting problem. The aim is to predict multiple future values of a time-dependant variable using a set of features (also depending on time). We begin by introducing some notation. Suppose that we measure a response variable $Y_i(t)$ and a vector of p covariates $X_i(t) = \big(X_{i1}(t), \ldots, X_{ip}(t)\big)$ at time t and location i. Denote by $A = \{a_1, \ldots, a_q\} \in \mathbb{R}^q_{\geq 0}$ the sorted set of target ahead values for the response variable; $L_k = \{\ell_{k1}, \ldots, \ell_{km_k}\} \in \mathbb{R}^{m_k}_{>0}$ a set of "lags" for the k-th predictor; and $L = \{L_1, \ldots, L_p\}$ a list of lags for all the covariates. Then the goal of *multi-period forecasting* (*MPF*) is to predict the response variable for all the aheads, i.e.

$$Y_i(t + A) = \big(Y_i(t + a_1), \ldots, Y_i(t + a_q)\big) \in \mathbb{R}^q,$$

using all the lagged features at location i, i.e.

$$X_i(t - L) = \big(X_{i1}(t - L_1), \ldots, X_{ip}(t - L_p)\big) \in \mathbb{R}^m.$$

Here, by analogy with the response,

$$X_{ik}(t - L_k) = \big(X_{ik}(t - \ell_{k1}), \ldots, X_{ik}(t - \ell_{km_k})\big) \in \mathbb{R}^{m_k}$$

represents the lagged values of the k-th predictor at location i and $m = \sum_{k=1}^{p} m_k$ corresponds to the total number of lagged predictors.

A simple example of an *MPF* problem is: on December 15, predict the expected number of newly reported of COVID-19 cases on December 15 and December 22 using the number of visits to the doctor on December 8 and December 1 across all the U.S. states. In this case,

- t is December 15, the forecast date;
- i represents a U.S. state;
- $Y_i(t)$ is the number of COVID-19 cases in state i on day t;
- $X_i(t) = \big(X_{i1}(t)\big)$ represents the number of doctor visits in state i on day t;
- $A = \{0, 7\}$ is the set of ahead values;
- $L_1 = \{7, 14\}$ is the set of lags.

Note that in many applications the response variable is also included in the set of predictors, thereby incorporating the historical values of the response into the feature set.

4.3 Multi-period forecaster

4.3.1 Baseline linear model

A straightforward (direct) multi-period forecaster is a linear model for each location i, timestamp t and ahead value a:

$$Y_i(t+a) = \sum_{k=1}^{p} \sum_{\ell \in L_k} X_{ik}(t-\ell)\ b_{k\ell}(a) + \epsilon_i(t+a). \tag{4.1}$$

Here $\epsilon_i(t+a) \sim \mathcal{N}(0, \sigma^2)$ are i.i.d errors and $b_{k\ell}(a)$ are unknown model coefficients. In what follows, we assume that the measurements are done at n locations and that multiple past values are available. If we denote the set of the available past timestamps by $T = \{t_1, \ldots, t_N\}$ then model (4.1) leads us to the following objective

$$\sum_{i=1}^{n} \sum_{t \in T} \sum_{a \in A} \left(Y_i(t+a) - \sum_{k=1}^{p} \sum_{\ell \in L_k} X_{ik}(t-\ell)\ b_{k\ell}(a) \right)^2 \tag{4.2}$$

that we aim to minimize w.r.t. the model coefficients. We note that the resulting optimization goal is nothing but a multivariate least-squares problem: the loss is separable in terms of ahead values, so $b_{k\ell}(a)$ can be found independently for each $a \in A$ via ordinary least squares with response $Y_i(t+a)$ and predictors $X_i(t-L)$.

For convenience we will restate the objective in matrix form. To do so, we first denote all the coefficients corresponding to the k-th predictor by

$$b_k(a) = \big(b_{k\ell_{k1}}(a), \ldots, b_{k\ell_{km_k}}(a)\big) \in \mathbb{R}^{m_k}$$

and form the coefficient matrix

$$B = \begin{pmatrix} b_1(a_1) & \cdots & b_p(a_1) \\ \vdots & \ddots & \vdots \\ b_1(a_q) & \cdots & b_p(a_q) \end{pmatrix} \in \mathbb{R}^{q\times m}.$$

Next, we denote the matrices of the response and the predictors measured at time t by

$$Y(t) = \begin{pmatrix} Y_1(t+A) \\ \vdots \\ Y_n(t+A) \end{pmatrix} \in \mathbb{R}^{n\times q} \quad \text{and} \quad X(t) = \begin{pmatrix} X_1(t-L) \\ \vdots \\ X_n(t-L) \end{pmatrix} \in \mathbb{R}^{n\times m}$$

and concatenate all the data row-wise into

$$Y = \begin{pmatrix} Y(t_1) \\ \vdots \\ Y(t_N) \end{pmatrix} \in \mathbb{R}^{Nn\times q} \quad \text{and} \quad X = \begin{pmatrix} X(t_1) \\ \vdots \\ X(t_N) \end{pmatrix} \in \mathbb{R}^{Nn\times m}.$$

Hence, the *MPF* optimization problem in Equation 4.2 can be stated in multi-response regression (MRR) form as

$$\underset{B\in\mathbb{R}^{m\times q}}{\text{minimize}} \, \|Y - XB^\top\|_F^2, \tag{4.3}$$

where $\|Z\|_F^2 = \sum_{ij} Z_{ij}^2$ is the squared Frobenius norm of a matrix Z. The explicit solution can be found via the formula

$$\widehat{B}^\top = (X^\top X)^{-1} X^\top Y.$$

We will refer to this forecaster as the *baseline MPF*.

4.3.2 Smoothing constraint

The main disadvantage of the baseline model (4.3) is that the coefficients for all the response columns are computed independently of each other. In other words, the model completely ignores the underlying data structure, i.e. that each column of Y represents the same signal measured for different ahead values. To incorporate this information into the *MPF* problem we desire some smoothness in the model coefficients.

Specifically, we desire that each $b_{k\ell}(a)$ is a smooth function of ahead values. Such smoothness can be enforced by requiring B to be representable as a linear combination of smooth basis functions

$h_1(a), \ldots, h_d(a)$, e.g. a spline or polynomial. This suggests the representation

$$b_{k\ell}(a) = \sum_{j=1}^{d} \theta_{jk\ell} h_j(a) \text{ for some } \theta_{jk\ell} \in \mathbb{R}. \tag{4.4}$$

Here d is a hyperparameter that controls the flexibility of $b_{k\ell}(a)$. In what follows, we refer to d as the *degrees-of-freedom*. Combining (4.2) with (4.4) leads us to the *smooth multi-period forecasting* (*SMPF*) objective

$$\underset{\theta_{jk\ell},\ \forall j,k,\ell}{\text{minimize}} \sum_{i=1}^{n} \sum_{t \in T} \sum_{a \in A} \left(Y_i(t+a) - \sum_{k=1}^{p} \sum_{\ell \in L_k} X_{ik}(t-\ell) \sum_{j=1}^{d} \theta_{jk\ell} h_j(a) \right)^2. \tag{4.5}$$

Note that the second term in (4.5) involves all the unknown parameters $\theta_{jk\ell}$ of the model, so the resulting loss function is no longer separable. However, since the predicted values

$$\hat{Y}_i(t+a) = \sum_{k=1}^{p} \sum_{\ell \in L_k} X_{ik}(t-\ell) \sum_{j=1}^{d} \theta_{jk\ell} h_j(a) \tag{4.6}$$

is a linear function of the coefficients it is still possible to find the explicit solution via regression.

Again, it is convenient to rewrite the loss function in matrix form. To do so, we first store all the coefficients in a matrix

$$\Theta = \begin{pmatrix} \theta_{11} & \ldots & \theta_{1p} \\ \ldots & \ldots & \ldots \\ \theta_{d1} & \ldots & \theta_{dp} \end{pmatrix} \in \mathbb{R}^{d \times m}, \text{ where } \theta_{jk} = (\theta_{jk\ell_{k1}}, \ldots, \theta_{jk\ell_{km_k}}) \in \mathbb{R}^{m_k}.$$

Next, we introduce the basis matrix

$$H = \begin{pmatrix} h_1(a_1) & \ldots & h_d(a_1) \\ \ldots & \ldots & \ldots \\ h_1(a_q) & \ldots & h_d(a_q) \end{pmatrix} \in \mathbb{R}^{q \times d},$$

where each column represents a function from the basis evaluated at all ahead values in A. As a result, one can restate constraint (4.4) in matrix form as $B = H\Theta$ and, together with (4.3), this implies the *SMPF* optimization can be written as

$$\underset{\Theta \in \mathbb{R}^{d \times m}}{\text{minimize}} \|Y - X\Theta^\top H^\top\|_F^2. \tag{4.7}$$

Note that in this problem the basis H is considered to be fixed, so the only unknown parameter is Θ. The degrees-of-freedom d, which controls the size of the basis, is the model's hyperparameter and can be chosen from a grid of values via cross-validation.

Similar to the baseline model, it is possible to find an explicit solution to (4.7). First, without

loss of generality, we assume that H has orthogonal columns. Otherwise, one can take the QR-decomposition $H = QR$ and apply the change of variables $\widetilde{H} = Q$ and $\widetilde{\Theta} = R\Theta$. Next, since the Frobenius norm is invariant under orthogonal transformations we can restate problem (4.7) as

$$\underset{\Theta \in \mathbb{R}^{d \times m}}{\text{minimize}} \|YH - X\Theta^\top\|_F^2,$$

which is, again, a multi-response regression problem with solution

$$\widehat{\Theta}^\top = (X^\top X)^{-1} X^\top Y H.$$

4.4 Missing values

This section extends the *SMPF* methodology proposed in Section 4.3.2 to the case when only part of the response matrix Y is observed. In forecasting applications, missing values often occur. For example, for a recent time t and location i we may not have observed response values $Y_i(t+a)$ for all ahead values $a \in A$ as some of them have not occurred yet. Moreover, the data can be updated at different times for different locations; thus, $Y_i(t+a)$ may not have been collected yet for some i.

To handle unobserved values we allow the set of ahead values to depend on the timestamp t and location i and denote it by $A_i(t)$. We also assume that each $A_i(t)$ is a subset of original $A = \{a_1, \ldots, a_q\}$. One can derive the new loss function as follows

$$\sum_{i=1}^{n} \sum_{t \in T} \sum_{a \in A_i(t)} \left(Y_i(t+a) - \sum_{k=1}^{p} \sum_{\ell \in L_k} X_{ik}(t-\ell) \sum_{j=1}^{d} \theta_{jk\ell} h_j(a) \right)^2. \tag{4.8}$$

Similar to Sections 4.3.1–4.3.2, it is not hard to restate the *SMPF* optimization problem in matrix form. Defining

$$W_i(t+a) = \begin{cases} 1 & \text{if } a \in A_i(t), \\ 0 & \text{otherwise,} \end{cases}$$

to be a binary weight matrix representing the missingness of the response, then minimizing Equation (4.8) is equivalent to solving

$$\underset{\Theta \in \mathbb{R}^{d \times m}}{\text{minimize}} \|W * (Y - X\Theta^\top H^\top)\|_F^2, \tag{4.9}$$

where $*$ refers to the element-wise Hadamard matrix product and W is the matrix containing all the weights.

Unlike the unweighted case, weighted *SMPF* cannot be reduced to a multi-response regression by simple manipulations with Frobenius norm. However, since the second term in (4.9) is a linear function of Θ it is still possible to restate it as an expanded ordinary least squares problem. Denote

$w, y \in \mathbb{R}^{Nnq}$ and $\theta \in \mathbb{R}^{dm}$ the vectors obtained by the concatenation of columns of matrices W, Y and $\Theta^\top$, respectively. Writing $\widetilde{X} = H \otimes X$ as the Kronecker product between H and X, then Equation (4.9) is equivalent to solving

$$\underset{\theta \in \mathbb{R}^{dm}}{\text{minimize}} \, \|w * (y - \widetilde{X}\theta)\|_2^2. \tag{4.10}$$

Note that for general w the solution can be found by means of the weighted regression with weights w, response y and feature matrix $\widetilde{X}$. However, if the weights are binary one can simply remove the rows in y and $\widetilde{X}$, that correspond to the zero weights, and use simple linear regression.

4.5 Simulation experiment

In this section we test the *SMPF* model from Section 4.4 on a small simulation example. For simplicity we use only one forecast date t and denote it as $t = 0$. We fix the number of locations at $n = 1000$ and the number of predictors at $p = 10$. We also assume no lags for this model, i.e. $L_k = \{0\}$ for $k = 1, \ldots, 10$. We first generate the matrix of covariates $X \in \mathbb{R}^{n \times p}$ with elements $X_{ik} \sim \mathcal{N}(0, 1)$. Further, we set the number of ahead values to $q = 30$ and the set of ahead values to $A = \{0, 1, \ldots, 29\}$. To create B we evaluate orthogonal quadratic polynomial basis at all elements in A and store them column-wise as $H \in \mathbb{R}^{q \times d}$. Here $d = 3$ and each column of H represents a basis function, including the intercept. Next, we draw the elements of the coefficient matrix $\Theta \in \mathbb{R}^{d \times m}$ from standard normal distribution. Finally, we generate the matrix of errors $E \in \mathbb{R}^{n \times q}$ with elements $\epsilon_{i,j} \sim \mathcal{N}(0, \sigma^2)$ and compute the response matrix as $Y = X\Theta^\top H^\top + E$. We randomly sample 10% of the Y matrix elements and treat them as unobserved.

We use half of the locations to fit the smooth multi-period forecasting model and the remaining half to evaluate the model performance. We vary the error variance σ^2 such that the signal-to-noise ratio is SNR $= 0.1, 0.5, 1, 2$, and we use mean absolute error (MAE) as the performance metric. Since, in practice, the true degrees-of-freedom is unknown, we it to vary over the grid $d = 1, 2, \ldots, 6$. For instance, $d = 1$ corresponds to the "null" constant model and $d = 2$ represents straight line forecasts. Thus for each value of SNR we produce a curve (MAE vs. degrees-of-freedom). The results are presented in Figure 4.1, where we also add the baseline multi-response regression solution as a reference (dashed red line).

According to the figure, for all SNR values the best smooth model outperforms the baseline, although the amount of improvement degrades slightly as SNR increases. Regardless of the signal-to-noise ratio, the minimum test score is achieved for the *SMPF* degrees-of-freedom around the true model value $d = 3$. Note that as the degrees-of-freedom increases, the *SMPF* still outperforms the Baseline, though setting $d = 30$ would necessarily result in identical performance. Therefore, in the simulation experiment the smooth multi-period forecasting model not only demonstrates the superior performance to the baseline method, but also is able to recover the true degrees-of-freedom.

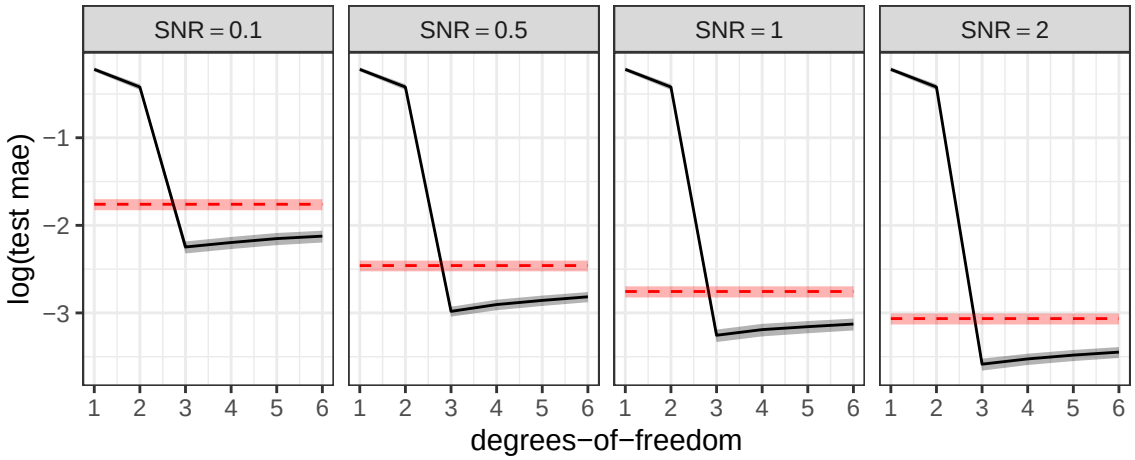


Figure 4.1: Simulation results. The solid black line represents the test MAE vs degrees-of-freedom computed by means of the *smooth MPF* model. The red dashed line corresponds to the MRR test score. Shaded regions represent 1SE interval computed across ten repeated simulations. Each panel corresponds to the simulated data with different SNR levels.

4.6 COVIDCast data experiments

Now we apply the multi-period forecasting approaches on the real data obtained from the Delphi COVIDCast API (Reinhart et al., 2021). This open-source dataset, which is updated daily, tracks multiple signals related to the spread and impact of the COVID-19 pandemic across the United States on both county and state levels. It contains a wide variety of typical COVID-19 metrics such as incident cases, deaths, and hospitalizations, as well as many unique indicators derived from mobility data, internet symptom searches, healthcare utilization reports, and sample surveys. For our experiments, we use three signals:

- `confirmed_7dav_incidence_prop`: the daily number of new confirmed COVID-19 cases (computed per 100,000 people);
- `smoothed_cli`: the estimated percentage of people with COVID-like illness, as measured by The Delphi Group at Carnegie Mellon University U.S. COVID-19 Trends and Impact Survey (CTIS), in partnership with Facebook (Salomon et al., 2021);
- `smoothed_hh_cmnty_cli`: the estimated percentage of people reporting illness in their local community, also measured by the Delphi US CTIS.

The latter two indicators were obtained from a voluntary survey conducted by Facebook. In order to reduce the weekly variability, all three signals are smoothed by taking the trailing average across a seven-day window. We consider the following forecast task:

- each location i represents a U.S. county;
- the response $Y_i(t)$ is the value of `confirmed_7dav_incidence_prop` at county i;
- three predictive features are used, i.e. $X_i(t) = (X_{i1}(t), X_{i2}(t), X_{i3}(t))$ represents the values of `confirmed_7dav_incidence_prop` as well as `smoothed_cli` and `smoothed_hh_cmnty_cli` at location i;
- ahead values $A = \{0, 1, \ldots, 27\}$ target daily forecast targets over four weeks;
- lag values $L = \{1, 2, \ldots, 28\}$ track the signal for four weeks preceding the forecast date.

The training set contains twelve weeks of daily data prior to 1 October 2021, that is

$$T_{train} = \{\text{10-Jul-2021, 11-Jul-2021, } \ldots \text{1-Oct-2021}\}.$$

To make the experiment more realistic, the data was downloaded "as reported on" 1 October 2021, thereby making all the signals after this date to be unobserved. In other words, $Y_i(t + a)$ is unobserved, or equivalently, $W_i(t + a) = 0$, if $t + a$ is any date after October 1. This practice also means that any revisions that would eventually be made after October 1 are not available. The distribution of missing response values for the training set is shown in blue in Figure 4.2. To test both *SMPF* models with and without missingness (the solutions to Equations (4.7) and (4.9)) we explore two scenarios:

Scenario 1: we remove all data for dates that would result in at least one unobserved ahead value, i.e. we use only the data from July 10 to September 4. In this case, the data is complete and we can use non-weighted *SMPF* for prediction.

Scenario 2: we include all the data from July 10 to October 1. Since the response matrix is only partially observed, we fit the weighted modification of *SMPF* with binary weights.

To make the solution more robust, among 581 counties with available survey data, we select the 300 with the highest average (across all the times) level of cases; we also remove all the observations containing missing values in the predictors. This results in 23079 training observations and 84 predictors.

We fit both *baseline* and *smooth MPF* models on the training set. For the smooth approach we use the orthogonal polynomial basis with intercept and vary the degrees-of-freedom in the grid $d = 1, 2, \ldots, 6$. To evaluate the models' performance we download the response values for the same 300 counties and including four weeks of observations following October 1. In other words, the new dataset contains the timestamps

$$T_{test} = \{\text{2-Oct-2021, 3-Oct-2021, } \ldots, \text{ 29-Oct-2021}\},$$

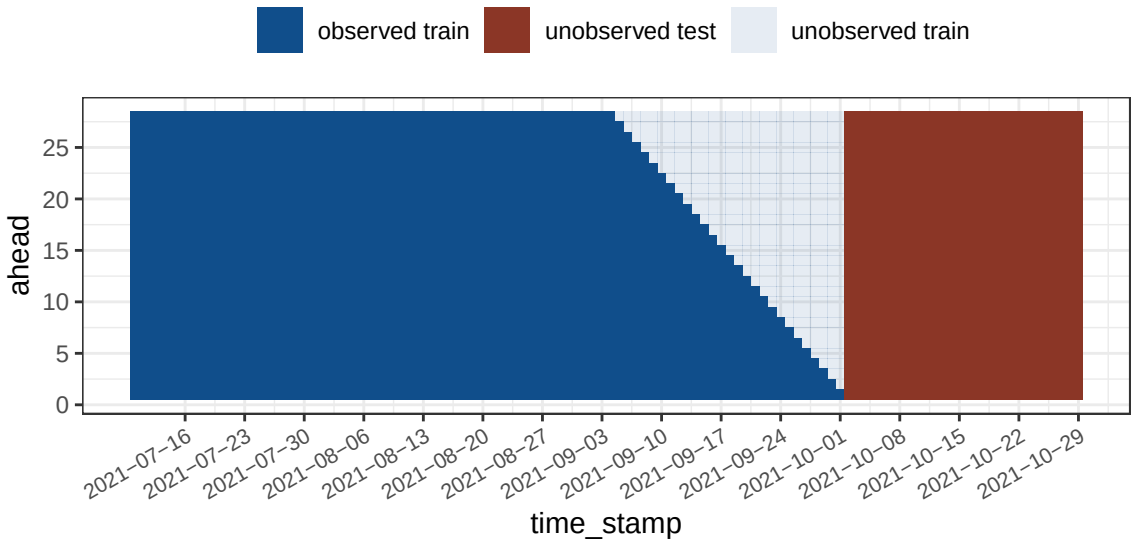


Figure 4.2: Schematic representation of missing values in response matrix when the "as of" date is set to October 1. Each column represents a timestamp; each row represents an ahead value; the element in row a column t corresponds to the $t + a$ time point. Blue and red colors represent train and test sets, respectively; light blue color corresponds the the time points after the "as of" date, which are treated as unobserved in the training phase. If $n = 1$, i.e. only one location is considered, then the picture represents exactly the distribution of missing values in train $Y^\top$ (the blue block) joined with test $Y^\top$ (the red block).

which results in 4780 test observations. Since we are interested in estimating how well the model will do at forecasting the future cases, the test set is downloaded "as of" 27 January 2022 and therefore there are no missing responses.

In Figure 4.3 we show test mean absolute error (MAE) for *smooth MPF* models with different degrees-of-freedom (solid line). We also include baseline MAE as a reference (dashed line). Here, the test MAE is averaged across all the locations, timestamps and ahead values. We start by comparing two data scenarios (blue and red colors in the figure). According to the plot, using all the data available before the "as of" date implies better test performance. This can be explained by the fact that COVID data is quite volatile, so including more recent observations allows the model to more accurately predict the future trend. This, however, comes at a price of increased computational cost. For a fully-observed response matrix the solution can be found via pre-multiplying Y by H and fitting the multi-response regression with feature matrix $X \in \mathbb{R}^{Nn\times m}$ and response matrix $YH \in \mathbb{R}^{Nn\times d}$. At the same time, the partially observed case requires us to solve a much larger regression problem with feature matrix $H \otimes X \in \mathbb{R}^{Nnq\times md}$ and response $y \in \mathbb{R}^{Nnq}$. Next, by comparing the *smooth* and *baseline MPF* test scores we conclude that smoothing improves the performance of multi-period forecaster. From the red and blue curves in Figure 4.3 one can infer that, for both scenarios, the

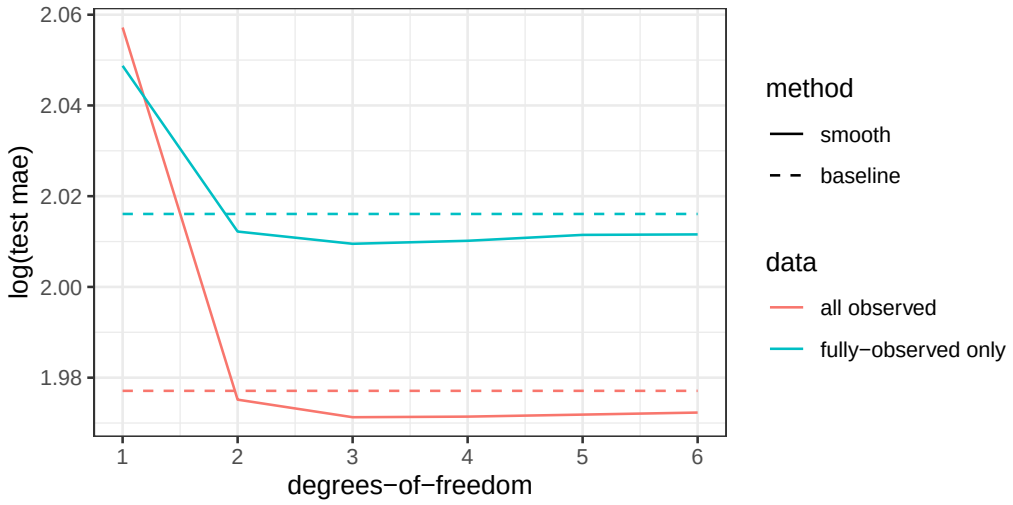


Figure 4.3: Comparing the test performance of the *baseline* and *smooth MPF* models while forecasting COVID-19. The data is downloaded "as of" October 1 and two scenarios are considered. Red color: the training data contains only the timestamps with fully-observed response vector (from July 10 to October 1), thus, the response matrix has no missing values. Blue color: the response matrix includes all the available timestamps; thus, it has some missing values (blue curve). The solid line shows the test MAE scores computed for the *smooth MPF* models with different degrees-of-freedom, which vary in the grid $d = 1, 2, \ldots, 6$. The dashed line represents the baseline model MAE. The plot demonstrates the superior performance of the smooth model to the baseline in both scenarios.

optimal value for the degrees-of-freedom is $d = 3$. The remaining results in this section are presented for the second data scenario, where the response matrix is partially observed.

To get more granular information on the model performance, we compute MAE separately for each column of Y and plot the dependence of test error on the ahead value. In Figure 4.4 we observe that, as one would expect, the accuracy decreases for larger ahead values for all models under consideration. In other words, forecasting is more challenging for time points that are farther into the future.

Finally, we compare *baseline MPF* with the best smooth model, i.e. the one that attains the lowest test score. Note that $d = 3$ gives quadratic dependence of the regression coefficients on time. Thus, the most promising approach is to predict some quadratic trend for cases at each timestamp. In Figure 4.5 each thin bright line starts at a timestamp and represents the predicted cases for the coming four weeks (28 ahead values). Here, the top row shows the baseline predictions, and the bottom row corresponds to those obtained by the optimal smooth model. To visualize and compare the *MPF* performance on the train and test sets, we include both train (blue color) and test (red color) fits to the plot. We also add the ground truth cases as a reference (dark bold line). To make the figure more readable, we present the results only for the five counties with the highest average case values and display each county in a separate panel. By analyzing this plot, we can

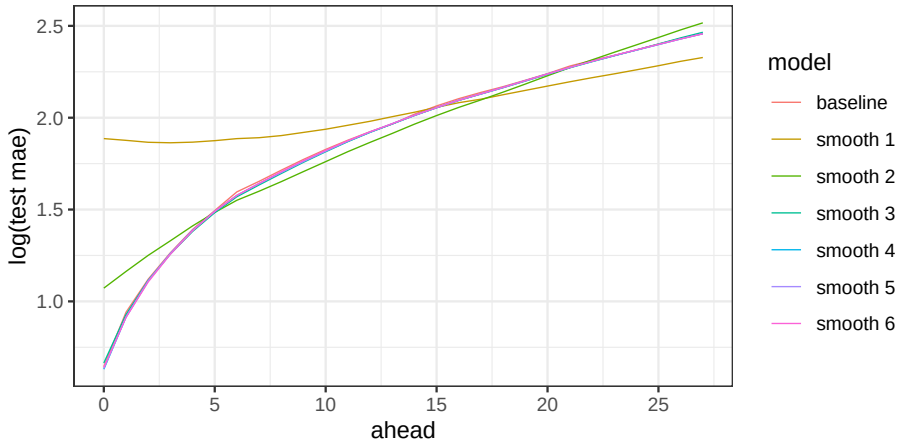


Figure 4.4: Comparing the test performance of the *baseline* and *smooth MPF* models while forecasting COVID-19. The result is presented for the second scenario, i.e. when the all the timestamps from July 10 to October 1 are included even if the response vector is partially observed. In this plot the test MAE is calculated for each ahead value separately and each line corresponds to different models (either baseline or smooth with $d = 1, 2, \ldots, 6$). The plot demonstrates that forecasting is more challenging for times which are further in the future.

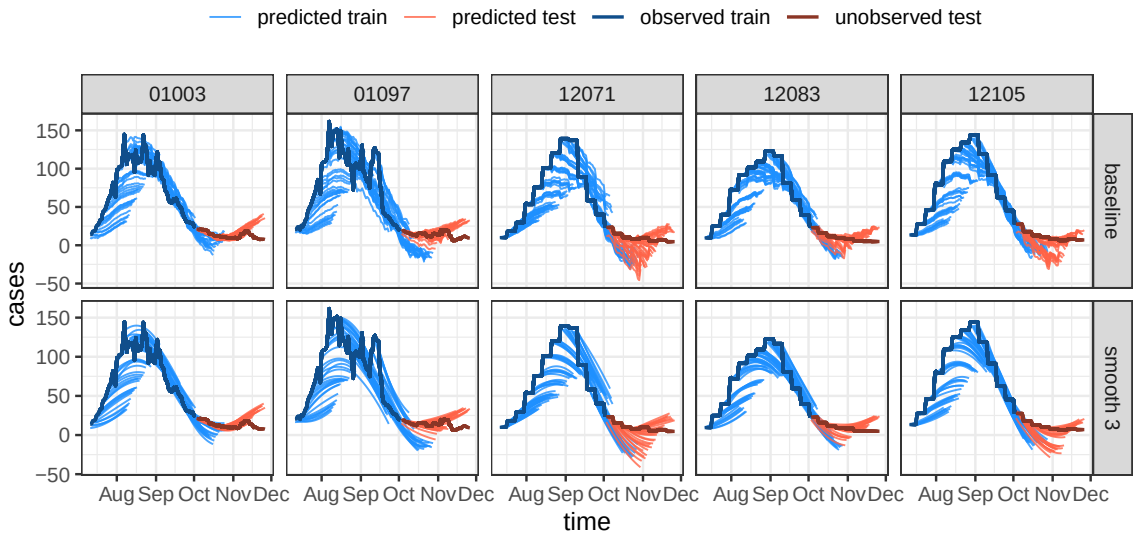


Figure 4.5: The plot displays the fits produced by the baseline and the optimal smooth model (with $d = 3$). The bold dark line shows the true value whereas predicted values are represented by bright thin lines (one line - one timestamp). Blue and red colors correspond to the train and test sets, respectively. The *baseline MPF* fit demonstrates irregular behavior which is moderated by smoothing.

see that the baseline model produces fits which look more wiggly, or noisy, relative to the *smooth MPF* prediction. This extra noise in the regression coefficients results in higher test MAE of the baseline compared to the competitor. Note that when true cases are close to zero, *MPF* may predict (impossible) negative values. One can easily fix this either by taking a log-transform of cases or by imposing a constraint on the predicted values.

4.7 Quantile forecasting

Now we shift from the point estimation task, which we handled by means of least squares regression, to interval prediction. In this section, we employ quantile regression (QR) to estimate intervals within which signals have a high probability of occurring (Koenker, 2005). We begin by introducing the *baseline quantile multi-period forecasting* (*QMPF*) method. For a quantile $\tau \in [0, 1]$ consider the pinball loss function

$$\rho_\tau(y, \hat{y}) = \begin{cases} \tau(y - \hat{y}) & \text{if } y \geq \hat{y}, \\ (1 - \tau)(\hat{y} - y) & \text{otherwise.} \end{cases}$$

Then goal is to solve the following objective

$$\underset{b_{k\ell}(a)}{\text{minimize}} \sum_{i=1}^{n} \sum_{t \in T} \sum_{a \in A_i(t)} \rho_\tau \Big(Y_i(t+a),\ \sum_{k=1}^{p} \sum_{\ell \in L_k} X_{ik}(t-\ell)\ b_{k\ell}(a) \Big). \tag{4.11}$$

Note that the above optimization task is stated in general form, where the set of ahead values can vary for each timestamp t and location i. We again assume $A_i(t) \subseteq A$.

Similar to Section 4.4, the solution to the *QMPF* problem can be found separately for each ahead value. Namely, for each a it amounts to fitting quantile regression with feature matrix X and the response vector which includes all the observed elements from Y that corresponds to a. As a result, each ahead value can be handled very efficiently by linear programming methods (Koenker, 2004).

Incorporating the smoothness into the coefficients leads us immediately to the smooth version of the *QMPF* objective

$$\sum_{i=1}^{n} \sum_{t \in T} \sum_{a \in A_i(t)} \rho_\tau \Big(Y_i(t+a),\ \sum_{k=1}^{p} \sum_{\ell \in L_k} X_{ik}(t-\ell) \sum_{j=1}^{d} \theta_{jk\ell} h_j(a) \Big), \tag{4.12}$$

which we aim to minimize w.r.t. $\theta_{jk\ell}$. By analogy with Section 4.4, the smooth problem can be reduced to fitting a weighted QR through some simple manipulations with X, Y, H and Θ. Specifically, one can show that minimizing (4.12) is equivalent to solving

$$\underset{\theta \in \mathbb{R}^{dm}}{\text{minimize}} \sum_{i=1}^{Nnq} w_i \cdot \rho_\tau \big(y_i,\ \widetilde{X}_i^\top \theta \big). \tag{4.13}$$

Here, $y, w \in \mathbb{R}^{Nnq}$ and $\theta \in \mathbb{R}^{dm}$ correspond to the vectors obtained by the concatenation of columns of matrices Y, W and $\Theta^\top$, respectively; W is the matrix of binary weights representing the the missing responses in Y; and $\widetilde{X}_i$ is the i-th row of $\widetilde{X} = H \otimes X$.

Note that, unlike the multiple least squares case, where the computations can be significantly simplified for fully-observed responses by pre-multiplying Y by H, the QR loss is not invariant under the orthogonal transformations. Thus, computing the extended feature matrix $\widetilde{X}$ is necessary for the *smooth QMPF* technique, regardless of the missingness pattern.

4.8 Quantile forecasting in COVIDCast study

We test both *baseline* and *smooth QMPF* techniques on the same COVIDCast data. We restrict our investigation only to the second scenario with partially observed responses. In our experiments we use three quantiles: $\tau = 0.5$ that corresponds to the predicted median value of cases and $\tau = 0.2, 0.8$ that we use to compute lower and upper bounds for the predicted intervals. For each τ we solve the *QMPF* optimization problem and calculate the resulting fit according to (4.6), which we hereafter denote by $\widehat{Y}_i^\tau(t+a)$. We denote by M the number of observed responses, i.e. $M = \sum_{i=1}^n \sum_{t \in T} |A_i(t)|$, and track three performance metrics:

$$\text{mean absolute error (MAE)} = \frac{1}{M} \sum_{i=1}^{n} \sum_{t \in T} \sum_{a \in A_i(t)} \left| Y_i(t+a) - \widehat{Y}_i^{0.5}(t+a) \right|,$$

$$\text{lower miscoverage rate (LMR)} = \frac{1}{M} \sum_{i=1}^{n} \sum_{t \in T} \sum_{a \in A_i(t)} \mathbf{1}\left\{ Y_i(t+a) < \widehat{Y}_i^{0.2}(t+a) \right\},$$

$$\text{upper miscoverage rate (UMR)} = \frac{1}{M} \sum_{i=1}^{n} \sum_{t \in T} \sum_{a \in A_i(t)} \mathbf{1}\left\{ Y_i(t+a) > \widehat{Y}_i^{0.8}(t+a) \right\}.$$

Here $\mathbf{1}\{\mathcal{B}\}$ refers to the indicator function, taking the value 1 on the event $\mathcal{B}$ and 0 otherwise. We evaluate these three metrics on the test set and present the results in Figure 4.6. According to the upper left panel, the smooth model with the lowest MAE score has $d = 3$ degrees-of-freedom. Despite implying that cases should be forecast in a simplistic quadratic fashion, it outperforms the baseline model in terms of MAE. In the bottom left panel of the plot we show the miscoverage rates obtained by 0.2 (green) and 0.8 (orange) quantiles. From this plot we can conclude that smoothing not only decreases the mean absolute error, but also can be helpful in improving the *QMPF* coverage, though this improvement is slight.

Analogously to Figure 4.5, we also examine the fitted values obtained by the baseline and the *smooth QMPF* model with three degrees-of-freedom. For simplicity, in Figure 4.6 we present the forecasted values for one timestamp (i.e. October 2) and the twenty counties with the highest average rate of cases. From the plot we can infer that for some counties, e.g. 01003 or 01097, smoothing can improve the prediction accuracy, although for others, e.g. 45035 or 45063, the difference is not

considerable.

4.9 Conformal calibration

Note that for both $\tau = 0.2, 0.8$ quantiles we expect to observe miscoverage of about 20%. Thus, *QMPF* models demonstrate mild undercoverage by the lower bound and more sever overcoverage by the upper one (see the left bottom panel of Figure 4.6). In this section we apply calibration to the QR model which allows us to improve the coverage on the test set.

Conformal quantile regression is a method for constructing prediction intervals that, without making distributional assumptions, helps achieve proper coverage in finite samples (Romano et al., 2019). The idea of this technique is to perform calibration of predicted values on some independent set. Thus, as a first step we split out training data into two parts: we refit the model on the first part and use the second one to calibrate the predicted cases. To reduce the correlation between these parts, we hold out four weeks of the most recent timestamps from T_{train} for calibration, i.e.

$$
\begin{aligned}
T_{\text{train}} &= T_{\text{train}}^{\text{fit}} \bigcup T_{\text{train}}^{\text{cal}}, \\
T_{\text{train}}^{\text{fit}} &= \{\text{10-Jul-2021, 11-Jul-2021, } \ldots, \text{ 3-Sep-2021}\}, \\
T_{\text{train}}^{\text{cal}} &= \{\text{4-Sep-2021, 5-Sep-2021, } \ldots, \text{ 1-Oct-2021}\}.
\end{aligned}
$$

After fitting *QMPF* models on $T_{\text{train}}^{\text{fit}}$ we use the resulting coefficients to evaluate the fits $\widehat{Y}_i^{\tau}(t+a)$ as well as the upper and lower errors

$$
\begin{aligned}
E_i^{0.2}(t+a) &= \widehat{Y}_i^{0.2}(t+a) - Y_i(t+a), \\
E_i^{0.8}(t+a) &= Y_i(t+a) - \widehat{Y}_i^{0.8}(t+a).
\end{aligned}
$$

Then, we use $T_{\text{train}}^{\text{cal}}$ to calculate the margins

$$
\begin{aligned}
Q^{0.2} = &\ \text{0.8-th empirical quantile of } \{E_i^{0.2}(t+a) : i \in [n],\ a \in A,\ t \in T_{\text{train}}^{\text{cal}}\}, \\
Q^{0.8} = &\ \text{0.8-th empirical quantile of } \{E_i^{0.8}(t+a) :\ i \in [n],\ a \in A,\ t \in T_{\text{train}}^{\text{cal}}\},
\end{aligned}
$$

and replace the original prediction interval $[\widehat{Y}_i^{0.2}(t+a),\ \widehat{Y}_i^{0.8}(t+a)]$ with its calibrated version $[\widehat{Y}_i^{0.2}(t+a) - Q^{0.2},\ \widehat{Y}_i^{0.8}(t+a) + Q^{0.8}]$.

We display the performance of *QMPF* after calibration in the right panel of Figure 4.6. As one can see from the bottom right panel of the plot, the procedure considerably improves the coverage, which is now much closer to the reference 20%. According to the upper right panel, the optimal smooth model has $d = 2$ degrees-of-freedom, suggesting forecasting a linear trend for cases. Finally, analyzing both panels, we conclude that, even for calibrated models, the smoothing technique still outperforms the baseline method on the test set.

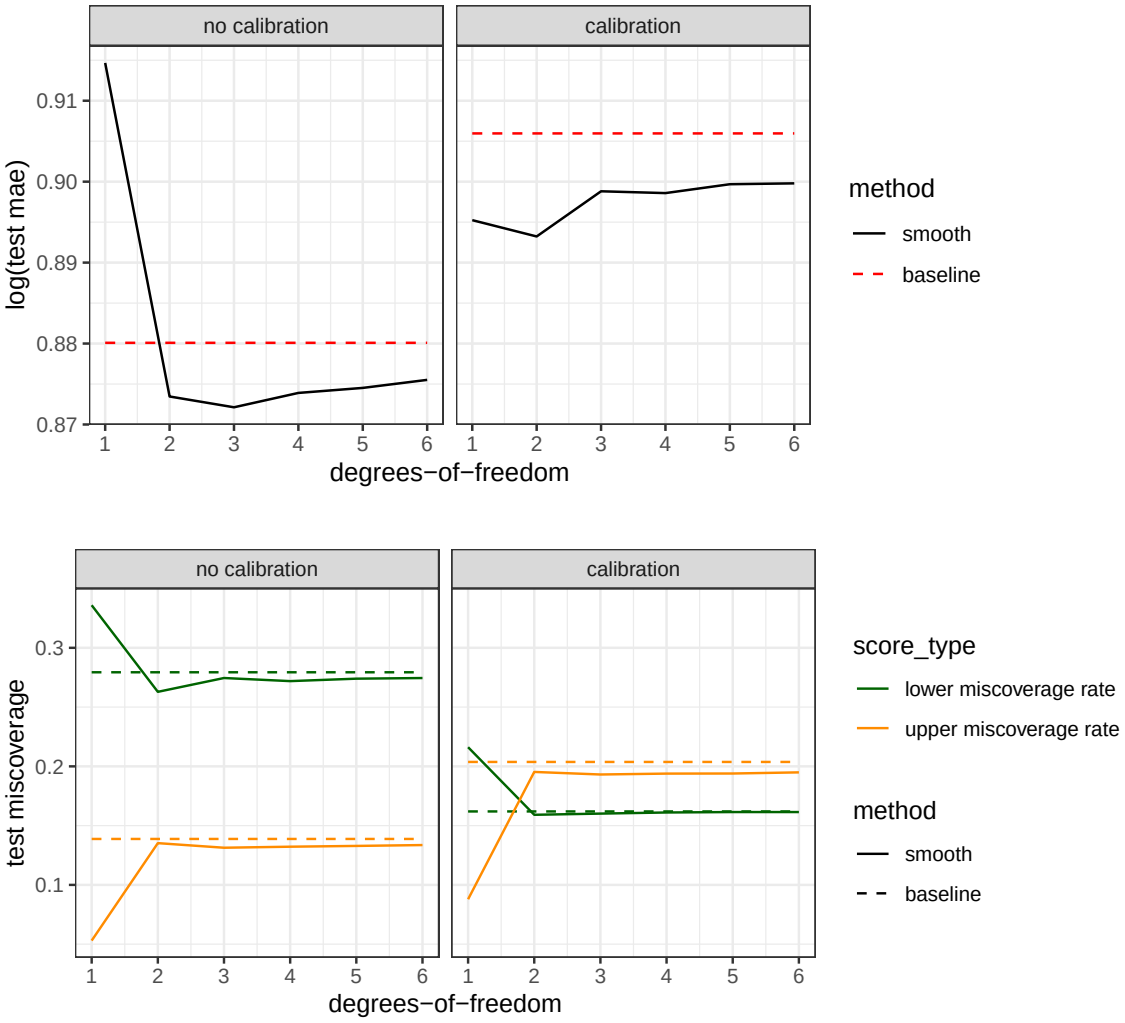


Figure 4.6: Comparison of the test performance of the *baseline* and *smooth QMPF* models for forecasting COVID-19. The plot represents the performance scores produced by the baseline model (dashed line) and the smooth models with different degrees-of-freedom (solid line). The upper plot shows the MAE score whereas the bottom plot shows the upper (orange) and lower (green) miscoverage rates. The target miscoverage rate is 20%. The left panel of each plot shows the performance of *QMPF* before conformal calibration, whereas the right panel represents the calibrated test scores. The plot demonstrates improved performance of the smooth model relative to the baseline.

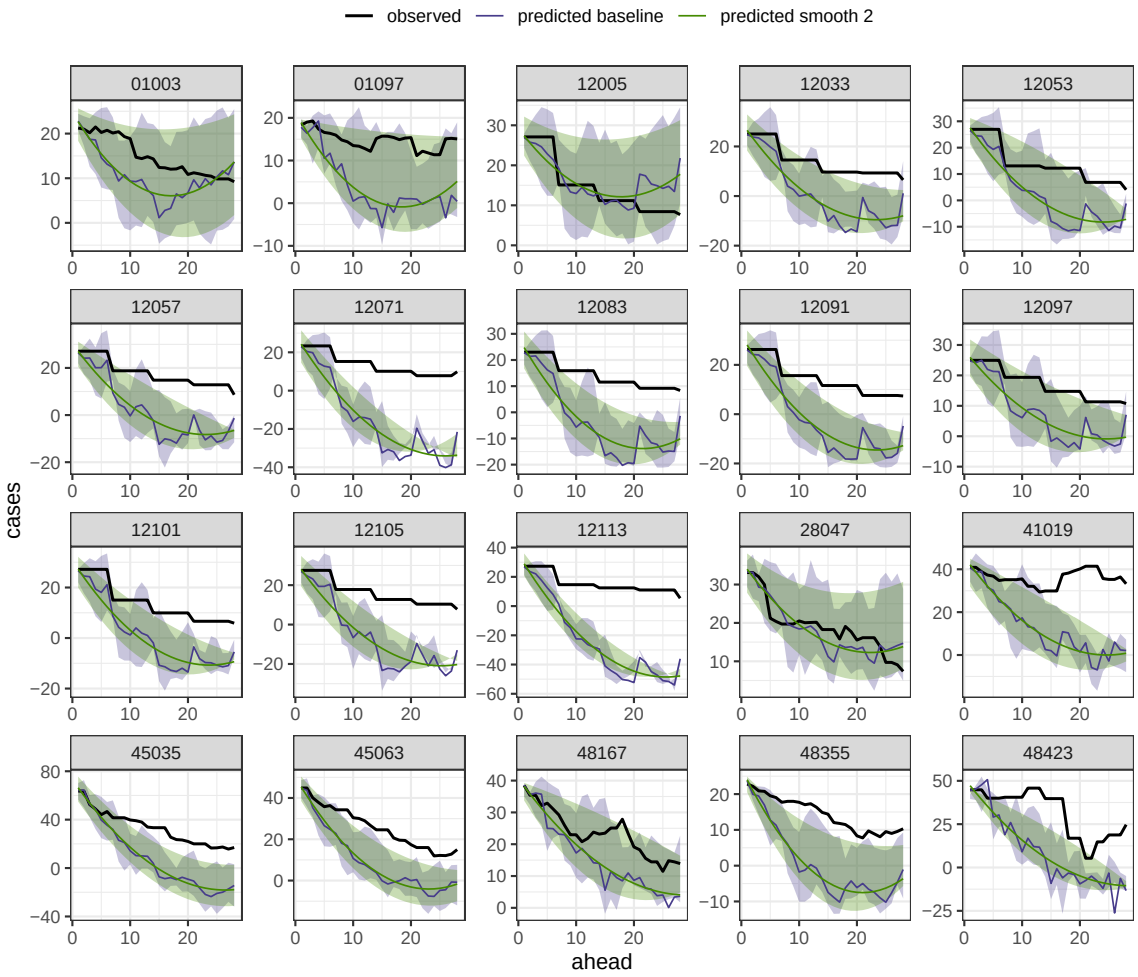


Figure 4.7: Comparison of the test predictions of the *baseline* and *smooth QMPF* models for forecasting COVID-19. The plot displays the out-of-sample fits produced by the baseline (purple) and the best smooth model with $d = 3$ (green). The fits are presented only for October 2. The bold black line shows the true observed newly reported cases, whereas predicted values are represented by thin colored lines. The prediction interval obtained by 0.2 and 0.8 quantiles is also displayed (shaded region).

4.10 Discussion

In this paper, we proposed a time-series forecasting approach intended to predict multiple "ahead" values of the signal simultaneously. The baseline method, commonly used in the literature, suggests treating each ahead value independently, thereby fitting several separate models. On the contrary, the *smooth MPF* technique takes into account that the same signal measured at different time points in the forecasting model. It assumes that the model coefficients depend smoothly on time, thereby forecasting multiple ahead values with a single smooth curve. We develop the proposed approach in a least-squares framework, which can be handled easily by multiple linear regression. Subsequently, we extend the methodology to forecasting the prediction intervals via quantile regression. We illustrate the benefits of smoothing in the context of multi-period forecasting through a small simulation as well as on an example using county-level COVID-19 incident cases.

There remains additional opportunity for future work. In the current study, we consider a limited set of predictors: cases, estimated percentage of people experiencing COVID-like illness, and the proportion of people reporting illness in their local community. One interesting direction would be to extend this set and include additional indicators from the COVIDCast database such as social behavior or mobility data. From the methodological point of view, this would require us to develop an efficient way to combine smooth multi-period forecasting with regularization. For instance, smooth structure in the coefficients can be handled by group-type penalties such as group-lasso.

Bibliography

An, N. H. and D. T. Anh (2015). Comparison of Strategies for Multi-step-Ahead Prediction of Time Series Using Neural Network. In *2015 International Conference on Advanced Computing and Applications (ACOMP)*, pp. 142–149.

Anderson, D. G. (1965). Iterative Procedures for Nonlinear Integral Equations. *J. ACM 12*(4), 547–560.

Ay, F., E. M. Bunnik, N. Varoquaux, S. M. Bol, J. Prudhomme, J. P. Vert, W. S. Noble, and K. G. Le Roch (2014). Three-dimensional modeling of the P. falciparum genome during the erythrocytic cycle reveals a strong connection between genome architecture and gene expression. *Genome Research 24*, 974–88.

Ban, F., D. Woodruff, and R. Zhang (2019). Regularized weighted low rank approximation. In *Advances in Neural Information Processing Systems*, Volume 32. Curran Associates, Inc.

Barch, D. M., G. C. Burgess, M. P. Harms, S. E. Petersen, B. L. Schlaggar, M. Corbetta, M. F. Glasser, S. Curtiss, S. Dixit, C. Feldt, D. Nolan, E. Bryant, T. Hartley, O. Footer, J. M. Bjork, R. Poldrack, S. Smith, H. Johansen-Berg, A. Z. Snyder, and D. C. Van Essen (2013, October). Function in the human connectome: Task-fMRI and individual differences in behavior. *NeuroImage 80*, 169–189.

Ben Taieb, S., A. Sorjamaa, and G. Bontempi (2010). Multiple-output modeling for multi-step-ahead time series forecasting. *Neurocomputing 73*(10), 1950–1957. Subspace Learning / Selected papers from the European Symposium on Time Series Prediction.

Bontempi, G. (2008, 01). Long term time series prediction with multi-input multi-output local learning. *Proceedings of the 2nd European Symposium on Time Series Prediction (TSP), ESTSP08*.

Box, G. E. P. and G. M. Jenkins (1976). *Time Series Analysis: Forecasting and Control*. Holden-Day.

Breiman, L., J. H. Friedman, R. A. Olshen, and C. J. Stone (1984). *Classification and Regression Trees*. New York: Chapman and Hall.

Buja, A., D. F. Swayne, M. L. Littman, N. Dean, H. Hofman, and L. Chen (2008). Data Visualization With Multidimensional Scaling. *Journal of Computational and Graphical Statistics 17*, 444–472.

Candès, E. and B. Recht (2008, 11). Exact Matrix Completion via Convex Optimization. *9*, 717–772.

Candès, E. and T. Tao (2010). The Power of Convex Relaxation: Near-Optimal Matrix Completion. *IEEE Transactions on Information Theory 56*(5), 2053–2080.

Cao, K. A. L., P. Martin, C. Robert-Granie, and P. Besse (2009). Sparse canonical methods for biological data integration: application to a cross-platform study. *BMC Bioinformatics 10(34)*, 1–17.

Capurso, D., H. Bengtsson, and M. R. Segal (2016). Discovering hotspots in functional genomic data superposed on 3D chromatin configuration reconstructions. *Nucleic Acids Research 44*, 2028–2035.

Capurso, D. and M. R. Segal (2014). Distance-based assessment of the localization of functional annotations in 3D genome reconstructions. *BMC Genomics 15*, 992.

Carver, C. S. and T. L. White (1994). Behavioral Inhibition, Behavioral Activation, and Affective Responses to Impending Reward and Punishment: The BIS/BAS Scales. *Journal of Personality and Social Psychology 67*(2), 319–333.

Caudai, C., E. Salerno, M. Zopp, and A. Tonazzini (2015). Inferring 3d chromatin structure using a multiscale approach based on quaternions. *BMC Bioinformatics 16*, 234.

Chen, R., L. Yang, and C. Hafner (2004). Nonparametric multistep-ahead prediction in time series analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 66*(3), 669–686.

Chen, X., L. Han, and J. Carbonell (2012). Structured Sparse Canonical Correlation Analysis. *Proceedings of Machine Learning Research 22*, 199–207.

Cheng, H., P. N. Tan, J. Gao, and J. Scripps (2006). Multistep-Ahead Time Series Prediction. In *Advances in Knowledge Discovery and Data Mining*, Berlin, Heidelberg, pp. 765–774. Springer Berlin Heidelberg.

Combettes, P. L. and J. C. Pesquet (2011). *Proximal Splitting Methods in Signal Processing*, pp. 185–212. Springer New York.

Combettes, P. L. and V. R. Wajs (2005). Signal Recovery by Proximal Forward-Backward Splitting. *Multiscale Modeling & Simulation 4*(4), 1168–1200.

Cox, R. W. and J. S. Hyde (1997). Software tools for analysis and visualization of fMRI data. *NMR in Biomedicine 10*(4-5), 171–178.

Dale, A. M., B. Fischl, and M. I. Sereno (1999). Cortical Surface-Based Analysis: I. Segmentation and Surface Reconstruction. *NeuroImage 9*(2), 179–194.

Dekker, J., K. Rippe, M. Dekker, and N. Kleckner (2002). Capturing chromosome conformation. *Science 295*, 1306–1311.

Desikan, R. S., F. Ségonne, B. Fischl, B. T. Quinn, B. C. Dickerson, D. Blacker, R. L. Buckner, A. M. Dale, R. P. Maguire, B. T. Hyman, M. S. Albert, and R. J. Killiany (2006). An automated labeling system for subdividing the human cerebral cortex on MRI scans into gyral based regions of interest. *NeuroImage 31*, 968–980.

Dixon, J. R., S. Selvaraj, F. Yue, A. Kim, Y. Li, Y. Shen, M. Hu, J. S. Liu, and B. Ren (2012). Topological domains in mammalian genomes identified by analysis of chromatin contacts. *Nature 485*, 376–380.

Duan, Z., M. Andronescu, K. Schutz, S. McIlwain, Y. J. Kim, C. Lee, J. Shendure, S. Fields, C. A. Blau, and W. S. Noble (2010). A three-dimensional model of the yeast genome. *Nature 465*, 363–367.

Dutta, A., X. Li, and P. Richtárik (2018). Weighted Low-Rank Approximation of Matrices and Background Modeling. *CoRR*.

Dutta, A., J. Liang, and X. Li (2021). A Fast and Adaptive SVD-free Algorithm for General Weighted Low-rank Recovery.

Eckart, C. and G. Young (1936). The approximation of one matrix by another of lower rank. *Psychometrika 1*(3), 211–218.

Esteban, O., C. J. Markiewicz, R. W. Blair, C. A. Moodie, A. I. Isik, A. Erramuzpe, J. D. Kent, M. Goncalves, E. DuPre, M. Snyder, H. Oya, S. S. Ghosh, J. Wright, J. Durnez, R. A. Poldrack, and K. J. Gorgolewski (2019). fMRIPrep: a robust preprocessing pipeline for functional MRI. *Nature Methods 16*(1), 111–116.

Fan, L., H. Li, J. Zhuo, Y. Zhang, J. Wang, L. Chen, Z. Yang, C. Chu, S. Xie, A. R. Laird, P. T. Fox, S. B. Eickhoff, C. Yu, and T. Jiang (2016). The Human Brainnetome Atlas: A New Brain Atlas Based on Connectional Architecture. *Cerebral Cortex (New York, N.Y.: 1991) 26*(8), 3508–3526.

Feuerverger, A., Y. He, and K. S. (2012). Statistical Significance of the Netflix Challenge. *Statistical Science 27*(2), 202 – 231.

Fudenberg, G. and L. A. Mirny (2012). Higher-order chromatin structure: bridging physics and biology. *Current Opinions in Genetics & Development 22*, 115–124.

Glasser, M. F., S. N. Sotiropoulos, J. A. Wilson, T. S. Coalson, B. Fischl, J. L. Andersson, J. Xu, S. Jbabdi, M. Webster, J. R. Polimeni, D. C. Van Essen, and M. Jenkinson (2013, October). The minimal preprocessing pipelines for the Human Connectome Project. *NeuroImage 80*, 105–124.

Gonzalez, I., D. S, P. Martin, and A. Baccini (2008). CCA: An R Package to Extend Canonical Correlation Analysis. *Journal of Statistical Software 23(12)*, 1–14.

Haber, S. N. (2017, January). Chapter 1: Anatomy and Connectivity of the Reward Circuit. In J. C. Dreher and L. Tremblay (Eds.), *Decision Neuroscience*, pp. 3–19. San Diego: Academic Press.

Hardoon, D., S. Szedmak, and J. Shawe-Taylor (2005). Canonical Correlation Analysis: An Overview with Application to Learning Methods. *Neural computation 16*, 2639–64.

Hardoon, D. R. and J. Shawe-Taylor (2011). Sparse Canonical Correlation Analysis. *Machine Learning 83*, 331—-353.

Hastie, T. J., R. Mazumder, J. D. Lee, and R. Zadeh (2015). Matrix Completion and Low-Rank SVD via Fast Alternating Least Squares. *Journal of Machine Learning Research 16*(104), 3367–3402.

Hastie, T. J. and W. Stuetzle (1989). Principal curves. *Journal of the American Statistical Association 406*, 502–516.

Hastie, T. J., R. J. Tibshirani, and J. H. Friedman (2009). *The Elements of Statistical Learning*. New York: Springer.

Hastie, T. J., R. J. Tibshirani, and M. J. Wainwright (2015). *Statistical Learning with Sparsity: The Lasso and Generalizations*. New York: Chapman and Hall.

Hotelling, H. (1936). Relations between two sets of variables. *Biometrika 28*, 321—-377.

Hutchins, L. N., S. M. Murphy, P. Singh, and J. H. Graber (2008). Position-dependent motif characterization using non-negative matrix factorization. *Bioinformatics 24*, 2684–2690.

Härdle, W. and L. Simar (2007). *Applied Multivariate Statistical Analysis: Second edition*. Springer.

Jenkinson, M., C. F. Beckmann, T. E. J. Behrens, M. W. Woolrich, and S. M. Smith (2012). FSL. *NeuroImage 62*(2), 782–790.

Jolliffe, I. (2002). *Principal Component Analysis*. New York: Springer.

Kidziński, L., F. K. C. Hui, D. I. Warton, and T. J. Hastie (2020). Generalized Matrix Factorization.

Koenker, R. (2004). *quantreg: An R package for quantile regression and related methods*.

Koenker, R. (2005). *Quantile Regression*. Econometric Society Monographs. Cambridge University Press.

Kruskal, J. B. and M. Wish (1978). *Multidimensional Scaling*. Newbury Park: Sage.

Kuss, M. (2003). The geometry of kernel canonical correlation analysis.

Lando, D., T. J. Stevens, S. Basu, and E. D. Laue (2018). Calculation of 3D genome structures for comparison of chromosome conformation capture experiments with microscopy: An evaluation of single-cell Hi-C protocols. *Nucleus 9*, 190–201.

Lee, C. S., R. W. Wang, H. H. Chang, D. Capurso, M. R. Segal, and J. E. Haber (2016). Chromosome position determines the success of double-strand break repair. *Proceedings of the National Academy of Science 113*, 146–154.

Lesne, A., J. Riposo, P. Roger, A. Cournac, and J. Mozziconacci (2014). 3D genome reconstruction from chromosomal contacts. *Nature Methods 11*, 1141–1143.

Leurgans, S. E., R. A. Moyeed, and B. W. Silverman (1993). Canonical Correlation Analysis when the Data are Curves. *Journal of the Royal Statistical Society 55(3)*, 725—-740.

Lieberman-Aiden, E., N. L. van Berkum, L. Williams, M. Imakaev, T. Ragoczy, A. Telling, I. Amit, B. R. Lajoie, P. J. Sabo, M. O. Dorschner, R. Sandstrom, B. Bernstein, M. A. Bender, M. Groudine, A. Gnirke, J. Stamatoyannopoulos, L. A. Mirny, E. S. Lander, and J. Dekker (2009). Comprehensive mapping of long-range contacts reveals folding principles of the human genome. *Science 326*, 289–293.

Lin, D., J. Zhang, J. Li, V. Calhoun, H.-W. Deng, and Y.-P. Wang (2013). Group Sparse Canonical Correlation Analysis for genomic data integration. *BMC bioinformatics 14*, Article 245.

Lykou, A. and J. Whittaker (2010). Sparse CCA using a Lasso with positivity constraints. *Computational Statistics and Data Analysis 54(12)*, 3144—-3157.

Mai, V. V. and M. Johansson (2020). Anderson Acceleration of Proximal Gradient Methods.

Makridakis, S., E. Spiliotis, and V. Assimakopoulos (2018, 03). Statistical and machine learning forecasting methods: Concerns and ways forward. *PLOS ONE 13*(3), 1–26.

Mardia, K. V., J. T. Kent, and J. M. Bibby (1979). *Multivariate Analysis*. New York, NY: Academic Press.

Markovsky, I. (2011). *Low Rank Approximation: Algorithms, Implementation, Applications*. Springer Publishing Company, Incorporated.

Mazumder, R., T. J. Hastie, and R. J. Tibshirani (2010). Spectral Regularization Algorithms for Learning Large Incomplete Matrices. *Journal of Machine Learning Research 11*, 2287–2322.

Mitelman, F., B. Johansson, and F. Mertens (2007). The impact of translocations and gene fusions on cancer causation. *Nature Reviews Cancer 7*, 233–245.

Muggeo, V. M. (2008). segmented: an R package to fit regression models with broken-line relationships. *Rnews 8*, 20–25.

Nesreen, K. A., F. A. Amir, E. G. Neamat, and E. S. Hisham (2010). An empirical comparison of machine learning models for time series forecasting. *Econometric Reviews 29*(5-6), 594–621.

Nesterov, Y. (1983). A method for solving the convex programming problem with convergence rate $o(1/k^2)$. *Proceedings of the USSR Academy of Sciences 269*, 543–547.

Nguyen, L. T., J. Kim, and B. Shim (2019). Low-Rank Matrix Completion: A Contemporary Survey. *IEEE Access 7*, 94215–94237.

Oksanen, J., F. G. Blanchet, M. Friendly, R. Kindt, P. Legendre, D. McGlinn, P. R. Minchin, R. B. O'Hara, G. L. Simpson, P. Solymos, H. Stevens, E. Szoecs, and H. Wagner (2016). vegan: Community Ecology Package. *R package version 2*, 4–1.

Park, J. and S. Lin (2017). A random effect model for reconstruction of spatial chromatin structure. *Biometrics 73*, 52–62.

Parkhomenko, E., D. Tritchlera, and J. Beyene (2009). Sparse canonical correlation analysis with application to genomic data integration. *Statistical Applications in Genetics and Molecular Biology 8*, 1–34.

Pruim, R. H. R., M. Mennes, D. van Rooij, A. Llera, J. K. Buitelaar, and C. F. Beckmann (2015). ICA-AROMA: A robust ICA-based strategy for removing motion artifacts from fMRI data. *NeuroImage 112*, 267–277.

Ramani, V., X. Deng, K. L. Gunderson, F. J. Steemers, C. M. Disteche, W. S. Noble, Z. Duan, and J. Shendure (2017). Massively multiplex single-cell Hi-C. *Nature Methods 14*, 263–266.

Rao, S. S., M. H. Huntley, N. C. Durand, E. K. Stamenova, I. D. Bochkov, J. T. Robinson, A. L. Sanborn, I. Machol, A. D. Omer, E. S. Lander, and E. L. Aiden (2014). A 3D map of the human genome at kilobase resolution reveals principles of chromatin looping. *Cell 159*, 1665–1680.

Razenshteyn, I., Z. Song, and D. P. Woodruff (2016). Weighted Low Rank Approximations with Provable Guarantees. STOC '16, New York, NY, USA, pp. 250–263. Association for Computing Machinery.

Reinhart, A., L. Brooks, M. Jahja, A. Rumack, J. Tang, S. Agrawal, W. Al Saeed, T. Arnold, A. Basu, J. Bien, Á. A. Cabrera, A. Chin, E. J. Chua, B. Clark, S. Colquhoun, N. DeFries, D. C. Farrow, J. Forlizzi, J. Grabman, S. Gratzl, A. Green, G. Haff, R. Han, K. Harwood, A. J. Hu,

R. Hyde, S. Hyun, A. Joshi, J. Kim, A. Kuznetsov, W. La Motte-Kerr, Y. J. Lee, K. Lee, Z. C. Lipton, M. X. Liu, L. Mackey, K. Mazaitis, D. J. McDonald, P. McGuinness, B. Narasimhan, M. P. O'Brien, N. L. Oliveira, P. Patil, A. Perer, C. A. Politsch, S. Rajanala, D. Rucker, C. Scott, N. H. Shah, V. Shankar, J. Sharpnack, D. Shemetov, N. Simon, B. Y. Smith, V. Srivastava, S. Tan, R. J. Tibshirani, E. Tuzhilina, A. K. Van Nortwick, V. Ventura, L. Wasserman, B. Weaver, J. C. Weiss, S. Whitman, K. Williams, R. Rosenfeld, and R. J. Tibshirani (2021). An open repository of real-time covid-19 indicators. *Proceedings of the National Academy of Sciences 118*(51).

Rennie, J. and N. Srebro (2005, 01). Fast maximum margin matrix factorization for collaborative prediction. pp. 713–719.

Rieber, L. and S. Mahony (2017). miniMDS: 3D structural inference from high-resolution hi-c data. *Bioinformatics 33*, 261–266.

Robin, G., J. Josse, E. Moulines, and S. Sardy (2019). Low-rank model with covariates for count data with missing values. *Journal of Multivariate Analysis 173*, 416–434.

Romano, Y., E. Patterson, and E. Candès (2019). Conformalized quantile regression. In *NeurIPS*.

Rosenthal, M., D. Bryner, F. Huffer, S. Evans, A. Srivastava, and N. Neretti (2019). Bayesian Estimation of 3D Chromosomal Structure from Single Cell Hi-C Data. *Journal of Computational Biology 26*, 1191–1202.

Salomon, J. A., A. Reinhart, A. Bilinski, E. J. Chua, W. La Motte-Kerr, M. M. Rönn, M. B. Reitsma, K. A. Morris, S. LaRocca, T. H. Farag, F. Kreuter, R. Rosenfeld, and R. J. Tibshirani (2021). The us covid-19 trends and impact survey: Continuous real-time measurement of covid-19 symptoms, risks, protective behaviors, testing, and vaccination. *Proceedings of the National Academy of Sciences 118*(51).

Segal, M. R. and H. L. Bengtsson (2015). Reconstruction of 3D genome architecture via a two-stage algorithm. *BMC Bioinformatics 16*, 373.

Segal, M. R. and H. L. Bengtsson (2018). Improved accuracy assessment for 3D genome reconstructions. *BMC Bioinformatics 19*, 196.

Setsompop, K., B. A. Gagoski, J. R. Polimeni, T. Witzel, V. J. Wedeen, and L. L. Wald (2012). Blipped-controlled aliasing in parallel imaging for simultaneous multislice echo planar imaging with reduced g-factor penalty. *Magnetic Resonance in Medicine 67*(5), 1210–1224.

Shavit, Y., F. K. Hamey, and P. Lio (2014). FisHiCal: an R package for iterative FISH-based calibration of Hi-C data. *Bioinformatics 30*, 3120–3122.

Somerville, L. H., S. Y. Bookheimer, R. L. Buckner, G. C. Burgess, S. W. Curtiss, M. Dapretto, J. S. Elam, M. S. Gaffrey, M. P. Harms, C. Hodge, S. Kandala, E. K. Kastman, T. E. Nichols, B. L. Schlaggar, S. M. Smith, K. M. Thomas, E. Yacoub, D. C. Van Essen, and D. M. Barch (2018). The Lifespan Human Connectome Project in Development: A large-scale study of brain connectivity development in 5-21 year olds. *NeuroImage 183*, 456–468.

Srebro, N. and T. Jaakkola (2003). Weighted Low-Rank Approximations. *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003) 20*, 720–727.

Stevens, T. J., D. Lando, S. Basu, L. P. Atkinson, Y. Cao, S. F. Lee, M. Leeb, K. J. Wohlfahrt, W. Boucher, A. O'Shaughnessy-Kirwan, J. Cramard, A. J. Faure, M. Ralser, E. Blanco, L. Morey, M. Sanso, M. G. S. Palayret, B. Lehner, L. Di Croce, A. Wutz, B. Hendrich, D. Klenerman, and E. D. Laue (2017). 3D structures of individual mammalian genomes studied by single-cell Hi-C. *Nature 544*, 59–64.

Tozzi, L., B. Staveland, B. Holt-Gosselin, M. Chesnut, S. E. Chang, D. Choi, M. Shiner, H. Wu, G. Lerma-Usabiaga, O. Sporns, D. M. Barch, I. H. Gotlib, T. J. Hastie, A. B. Kerr, R. A. Poldrack, B. A. Wandell, M. Wintermark, and L. M. Williams (2020). The human connectome project for disordered emotional states: Protocol and rationale for a research domain criteria study of brain connectivity in young adult anxiety and depression. *NeuroImage 214*, 116715.

Trieu, T., O. Oluwadare, and J. Cheng (2019). Hierarchical reconstruction of high-resolution 3D models of large chromosomes. *Scientific Reports 9*, 4971.

Tuzhilina, E., T. J. Hastie, and M. R. Segal (2020). Principal curve approaches for inferring 3d chromatin architecture. *Biostatistics*.

Varoquaux, N., F. Ay, W. S. Noble, and J. P. Vert (2014). A statistical approach for inferring the 3D structure of the genome. *Bioinformatics 30*, 26–33.

Vinod, H. D. (1976). Canonical Ridge and Econometrics of Joint Production. *Journal of Econometrics 4(2)*, 147—-166.

Waaijenborg, S., P. C. V. de Witt Hamer, and A. Zwinderman (2008). Quantifying the Association between Gene Expressions and DNA-Markers by Penalized Canonical Correlation Analysis. *Statistical Applications in Genetics and Molecular Biology 7(1)*, Article 3.

Walker, H. F. and P. Ni (2011). Anderson Acceleration for Fixed-Point Iterations. *SIAM Journal on Numerical Analysis 49*(4), 1715–1735.

Wang, H. T., J. Smallwood, and J. Mourao-Miranda (2020). Finding the needle in a high-dimensional haystack: Canonical correlation analysis for neuroscientists. *Neuroimage 216*, 116745.

Wang, S., J. H. Su, B. J. Beliveau, B. Bintu, J. R. Moffitt, C. T. Wu, and X. Zhuang (2016). Spatial organization of chromatin domains and compartments in single chromosomes. *Science 353*, 598–602.

Wardenaar, K. J., T. van Veen, E. J. Giltay, E. de Beurs, B. W. J. H. Penninx, and F. G. Zitman (2010). Development and validation of a 30-item short adaptation of the Mood and Anxiety Symptoms Questionnaire (MASQ). *Psychiatry Research 179*(1), 101–106.

Watson, D., L. A. Clark, and A. Tellegen (1988). Development and validation of brief measures of positive and negative affect: the PANAS scales. *Journal of Personality and Social Psychology 54*(6), 1063–1070.

Witten, D. M. and W. S. Noble (2012). On the assessment of statistical significance of three-dimensional colocalization of sets of genomic elements. *Nucleic Acids Research 40*, 3849–3855.

Witten, D. M. and R. J. Tibshirani (2009). Extensions of Sparse Canonical Correlation Analysis with Applications to Genomic Data. *Statistical applications in genetics and molecular biology 8(1)*, Article 28.

Yang, T., F. Zhang, G. G. Yardimci, F. Song, R. C. Hardison, W. S. Noble, F. Yue, and Q. Li (2017). HiCRep: assessing the reproducibility of Hi-C data using a stratum- adjusted correlation coefficient. *Genome Research 27*, 1939–1949.

Zhang, G., B. E. Patuwo, and M. Y. Hu (1998). Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting 14*(1), 35–62.

Zhang, J., B. O'Donoghue, and S. Boyd (2020). Globally Convergent Type-I Anderson Acceleration for Nonsmooth Fixed-Point Iterations. *SIAM Journal on Optimization 30*(4), 3170–3197.

Zhang, Z., G. Li, T. K. C., and W. K. Sung (2013). 3D chromosome modeling with semi-definite programming and Hi-C data. *Journal of Computational Biology 20*, 831–846.

Zhuang, X., Z. Yang, and D. Cordes (2020). A technical review of canonical correlation analysis for neuroscience applications. *Human Brain Mapping 41(13)*, 3807–3833.

Zou, C., Y. Zhang, and Z. Ouyang (2016). HSA: integrating multi-track Hi-C data for genome-scale reconstruction of 3D chromatin structure. *Genome Biology 17*, 40.