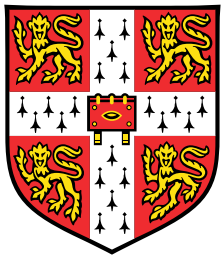
High-dimensional covariance estimation with applications to functional genomics



Harry Gray

MRC Biostatistics Unit
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

Darwin College February 2019

Declaration

This thesis is the result of my own work and includes nothing which is the outcome of work done in collaboration except as declared in the Acknowledgements and specified in the text. It is not substantially the same as any that I have submitted, or, is being concurrently submitted for a degree or diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Acknowledgements and specified in the text. I further state that no substantial part of my thesis has already been submitted, or, is being concurrently submitted for any such degree, diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Acknowledgements and specified in the text. It does not exceed the prescribed word limit for the relevant Degree Committee.

Harry Gray
February 2019

Abstract

High-dimensional covariance estimation with applications to functional genomics

Harry Gray

Covariance matrix estimation plays a central role in statistical analyses. In molecular biology, for instance, covariance estimation facilitates the identification of dependence structures between molecular variables that shed light on the underlying biological processes. However, covariance estimation is generally difficult because high-throughput molecular experiments often generate high-dimensional and noisy data, possibly with missing values. In such context, there is a need to develop scalable and robust estimation methods that can improve inference by, for example, taking advantage of the many sources of external information available in public repositories.

This thesis introduces novel methods and software for estimating covariance matrices from high-dimensional data. Chapter 2 introduces a flexible and scalable Bayesian linear shrinkage covariance estimator. This accommodates multiple shrinkage target matrices, allowing the incorporation of external information from an arbitrary number of sources. It is also less sensitive to target misspecification and can outperform state-of-the-art single-target linear shrinkage estimators.

Chapter 3 explores a dimensionality reduction approach — probabilistic principal component analysis — as a model-based covariance estimation method that can handle missing values. By assuming a low-dimensional latent structure, this is particularly useful when the inverse covariance is required (e.g. network inference). All of our methods are implemented as well-documented open-source `R` libraries.

Finally, Chapter 4 presents a case study using a dataset of cytokine expression in patients with traumatic brain injury. Studies of this type are crucial to researching the inflammatory response in the brain and potential patient recovery. However, due to the difficulties in patient recruitment, they result in high-dimensional datasets with relatively low sample sizes. We show how our methods can facilitate the multivariate analysis of cytokines across time and different treatment regimes.

Acknowledgements

This work was funded by a very generous grant from The Wellcome Trust. I would like to express my sincere thanks to them for having been selected for this opportunity four years ago and for all of the life-changing events that have arisen because of it.

This thesis contains an amount of collaborative work that is to be declared here. Firstly, all work presented in this thesis was conducted under the guidance of my supervisors Sylvia Richardson, Gwenaël Leday, and Catalina Vallejos. The contents of Chapter 2 consists of a draft manuscript that is available as a preprint at https://arxiv.org/abs/1809.08024, in which I am listed as first author. Gwenaël Leday and Catalina Vallejos contributed towards code to generate the figures in Sections 2.6 and 2.7. Otherwise, all work presented in this chapter is my own.

The work presented in Chapter 3 was done so under the guidance of Paul Kirk of the MRC-Biostatistics Unit. The efforts to obtain the derivations presented in Appendices B.1, B.2, B.5, and in developing the software package presented in Section 3.9 were shared approximately equally. Apart from this, all other work in this chapter is my own. This includes all results generated using the software package.

The work of Chapter 4 was done in collaboration with Adel Helmy and Eric Thelin of the Department of Clinical Neuroscience at Addenbrooke's Hospital. Adel and Eric provided the dataset that is presented for analysis and consulted on biologically motivated questions to pursue. All other work from this chapter is my own.

I am very grateful to all those listed above for their contributions towards this academic work. Without their help and guidance, neither this thesis nor any magnitude of my academic development would have been possible. I am particularly grateful for the insurmountable patience and understanding shown by my supervisors throughout the past few years.

I am also humbled by the support, love, and enjoyment that has been shared with me by my friends and family throughout this time. I have been incredibly fortunate to be surrounded by so many unforgettable people and lack the words to describe what you all mean to me. I would like to express my deepest gratitude towards you all, knowing that the most valuable discovery I have stumbled upon during my Ph.D has been learning just how much good people can create.

Table of contents

Chapter 1

Introduction

This thesis is concerned with the estimation of high-dimensional covariance matrices in the context of functional genomics, whose aim is to study gene regulation using high-throughput molecular data. This Chapter introduces biological and statistical concepts as a background for the following Chapters. It also provides some context about the field of molecular biology that has become a data-rich discipline (both in the amount and complexity) that poses challenges.

1.1 Background and context

Genetic information within a cell is responsible for the various functions that it can carry out. Molecular biology aims at understanding the processes by which this information is processed to determine cellular functions. Understanding these processes is crucial for understanding and treating diseases.

The rapid development and decreasing cost of high-throughput technologies in the past decades have remarkably changed the face of molecular biology. These technologies allow the simultaneous measurement of thousands of molecular variables (e.g. genes, proteins, metabolites, etc.) and generate large amounts of molecular data that prove difficult to analyse.

The data generated by high-throughput experiments are diverse. For example, sequencing experiments generate discrete count data, whilst microarrays provide continuous data. Statistical methods need to be developed for each type of data in order to model them correctly. Statistical methods are therefore challenged by the nature of this data.

The data generated by high-throughput experiments are high-dimensional. This means that the number of measured molecular variables is much larger than the number of samples. Such data are statistically more challenging to analyse because classic

approaches, such as maximum likelihood that are based on large sample assumptions, fail (Section 1.2).

Besides the data generated by biotechnologies, there exists a wealth of auxiliary information, such as platform annotations, public data repositories (such as The Cancer Genome Atlas; http://cancergenome.nih.gov/) and databases (such as gene ontology; http://geneontology.org) that are very useful to improve model interpretation as well as statistical power. Such auxiliary information therefore constitutes an important source of external information that is desirable to take into account.

In this rich environment where big, high-dimensional data as well as auxiliary information are increasingly available, there is a strong need to develop methods and software that are computationally efficient as well as flexible to make the most out of the available information.

1.2 Covariance estimation

Molecular entities engage in complex interactions in order to produce biological functionality. Examples of this are found in signalling pathways, such as p53, in which many proteins collaborate to regulate the cell cycle and prevent cancer. Analyses that focus on individual variables (e.g. genes) are incapable of capturing the intricacies of these higher-order interactions, providing the need for more complicated multivariate statistical analysis. The covariance is a statistic that captures pairwise linear associations and univariate variances. It can be useful for exploring simple and interpretable relationships between pairs of variables as well as characterising groups of variables that behave similarly. The covariance matrix may also be used to perform more complicated analyses. For example, in the case of principal component analysis, the observed variables are mapped onto unobserved latent variables by decomposing information contained within the covariance matrix.

However, estimating the covariance matrix from high-dimensional data is challenging. This is a consequence of the number of parameters that are required to estimate it and the relatively small sample sizes with which to estimate them. In fact, the number of parameters required to estimate the covariance matrix grows quadratically with the number of variables under study. This means that there are not enough degrees of freedom, which results in a sample covariance matrix whose entries are estimated with a high amount of statistical error.

Another way of viewing this problem is to consider the eigen-decomposition of the sample covariance matrix. Whenever the sample size is lower than the number of measurements, the covariance matrix is rank deficient. This means that many eigenvalues of the matrix are equal to zero. Consequently the condition number, which

is the ratio between the largest and smallest eigenvalue and can be interpreted as a measure of error when performing arithmetic operations, is extremely large or infinite. Large condition numbers also arise in situations in which the sample size is only marginally larger than the number of measurements. A large condition number severely hampers the reliability and usefulness of the estimator. For example, it is not possible to obtain an estimate of the inverse covariance, which is the basis for the reconstruction of networks (conditional independence graphs).

As concrete evidence of this problem, we provide a small numerical example. For different combinations of the number of variables $p \in \{200,400,600,800,1000\}$ and number of observations $n \in \{10p, 2p, p, p/2, p/10\}$, we generate 100 data sets from a multivariate normal distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, with identity covariance matrix $\boldsymbol{\Sigma} = \boldsymbol{I}_{p\times p}$. For each generated data set $\boldsymbol{X}$, we compute: (i) the sample covariance matrix $\boldsymbol{S} = \boldsymbol{X}\boldsymbol{X}^\top/n$, (ii) the associated (squared) Frobenius distance between $\boldsymbol{S}$ and $\boldsymbol{\Sigma}$ as a measure of error, hereafter referred to as the Frobenius loss and defined as $\|\boldsymbol{\Sigma} - \boldsymbol{S}\|_F^2 = \sum_i^p \sum_j^p (\Sigma_{ij} - S_{ij})^2$, and (iii) the condition number of $\boldsymbol{S}$. These results are summarised in Figure 1.1. As described, we observe a higher estimation error (reflected in larger Frobenius distances) when the ratio p/n increases. We also observe that $\boldsymbol{S}$ is singular whenever $n \leq p$.

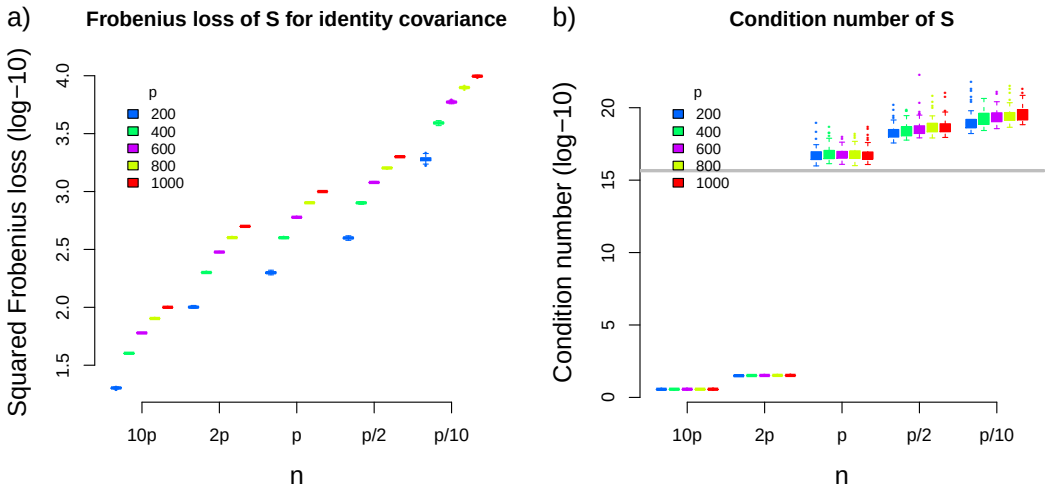


Fig. 1.1 Properties of the sample estimator for $\boldsymbol{\Sigma}$. Sub-figure (a) shows the squared Frobenius distance between the sample covariance matrix $\boldsymbol{S} = \boldsymbol{X}\boldsymbol{X}^\top/n$ and $\boldsymbol{\Sigma}$ whereas (b) shows the condition number of $\boldsymbol{S}$. The grey line represents the condition number for which matrices are declared numerically singular by the `solve()` function in `R`.

The solution to this problem is to perform regularisation, i.e. to modify the estimator to ensure that it is well-conditioned. This is typically done by introducing a penalty term in the likelihood and restricting the parameter space (e.g. via hard inequality constraints). Next, we discuss such methods.

1.3 Regularised covariance estimation

The regularisation of the sample covariance matrix is a well-studied problem [88], for which many types of solution have been proposed, such as thresholding [12, 21], inducing sparsity [14], and imposing a condition number constraint [111], with no one-size-fits-all approach emerging. In this thesis, we focus on linear shrinkage estimators [70] as well as probabilistic principal component analysis (PPCA) [102], which are the subjects of Chapters 2 and 3, respectively. Here, we introduce these two approaches.

1.3.1 Linear shrinkage estimators

Linear shrinkage estimators represent a simple, computationally efficient, and interpretable solution to the problem of covariance estimation. Originating with Stein [99] in order to improve the estimator of the mean of a multivariate Normal distribution, the idea is to linearly weight the sample estimator with a biased 'target' estimator. In the context of covariance matrix estimation, a linear shrinkage estimator is often defined as

$$\hat{\boldsymbol{\Sigma}} = \alpha\boldsymbol{\Delta} + (1-\alpha)\boldsymbol{S}, \tag{1.1}$$

where $\boldsymbol{\Delta}$ is positive definite and known as the target matrix and $\alpha \in (0,1)$ is the shrinkage intensity. The shrinkage intensity α can, for example, be determined to minimise the mean squared-error (MSE) $\mathbb{E}\left[\|\boldsymbol{\Sigma} - \hat{\boldsymbol{\Sigma}}\|_F^2\right]$, which can be decomposed [95, 70] as

$$\begin{aligned}\mathrm{MSE}(\hat{\boldsymbol{\Sigma}}) =& \mathrm{Var}(\hat{\boldsymbol{\Sigma}}) + \mathrm{Bias}^2(\hat{\boldsymbol{\Sigma}}) && (1.2)\\ =& \sum_{i=1}^{p}\sum_{j=1}^{p} \alpha^2 \mathrm{Var}\left(\Delta_{ij}\right) + (1-\alpha)^2 \mathrm{Var}(S_{ij}) + 2\alpha(1-\alpha)\mathrm{Cov}\left(S_{ij}, \Delta_{ij}\right) \\ &+ \left(\alpha\mathbb{E}\left[\Delta_{ij} - S_{ij}\right] + \mathrm{Bias}(S_{ij})\right)^2. && (1.3)\end{aligned}$$

Comparing Equation (1.1) with Equation (1.2) and recalling that $\boldsymbol{\Delta}$ induces more bias and $\boldsymbol{S}$ induces more variance, linear shrinkage represents a bias-variance trade-off in this case. The weight α controls this trade-off in order to lower the overall MSE when compared to that of $\boldsymbol{S}$. The expansion in Equation (1.3) shows exactly how each element of the shrinkage estimator contributes to the MSE. If $\boldsymbol{\Delta}$ is similar to $\boldsymbol{S}$ then that reduces the burden of the latter terms but increases the burden of the first term and covariance term. A $\boldsymbol{\Delta}$ with high bias might well reduce the contribution of the first term, but it will certainly increase the contribution of the second from last term. The ideal scenario would be a $\boldsymbol{\Delta}$ which has low variance but also retains key similarities of $\boldsymbol{\Sigma}$, which in small samples may or may not be contained within $\boldsymbol{S}$.

Previous work in this area has mostly focussed on optimally estimating α for a specified target matrix $\boldsymbol{\Delta}$. In their seminal paper, Ledoit and Wolf [70] introduced a method for estimating the optimal α that results in a closed-form computationally trivial solution for a single-parameter diagonal target matrix under a Gaussian assumption for the data. Since then, their methodology has been used to generate solutions for more target matrices [95], improved α estimation [24], and relaxing the distributional assumptions of the data [103]. Conversely, very few works have focussed onto extending this model to use multiple shrinkage targets [64, 6, 59], despite it representing a promising avenue for more flexible shrinkage.

1.3.2 Probabilistic principal component analysis

Principal component analysis (PCA) is a popular dimension reduction technique that aims to find the linear projection with minimum MSE with the data [86]. It can also be formulated as a lower-dimensional linear projection of the data onto orthogonal axes, such that the variance of the projection is maximised [58]. For our purposes, we focus on the latter interpretation. This is because the basis of the orthogonal projection intuitively coincides with the eigenvectors of the sample covariance matrix $\boldsymbol{S}$, denoted $(\boldsymbol{u}_1,\ldots,\boldsymbol{u}_p)$, and the magnitude of variance along those vectors corresponds to the magnitude of their associated eigenvalues $(\lambda_1,\ldots,\lambda_p)$. We assume from now that the eigenvalues are arranged in descending order $\lambda_1 > \ldots > \lambda_p$.

For n samples of p observed variables $\boldsymbol{x}_j$, $j = 1,\ldots,n$, with empirical mean $\overline{\boldsymbol{x}} = \frac{1}{n}\sum_{j=1}^{n}\boldsymbol{x}_j$, the PCA projection is defined using

$$\boldsymbol{x}_j = \boldsymbol{W}\boldsymbol{z}_j + \overline{\boldsymbol{x}}_j, \tag{1.4}$$

with the columns of $\boldsymbol{W} = (\boldsymbol{u}_1,\ldots,\boldsymbol{u}_p)$. The matrix $\boldsymbol{W}$ in Equation (1.4) represents a transformation of the latent variables into the observed p-dimensional space. Dimension reduction is achieved through retaining only the eigenvectors associated with the q largest eigenvalues $(\boldsymbol{u}_1,\ldots,\boldsymbol{u}_q)$, $q < p$, e.g. reducing the number of columns of $\boldsymbol{W}$ from p to q. The vector $\boldsymbol{z}_j$ is then estimated as $\boldsymbol{W}^\top(\boldsymbol{x}_j - \overline{\boldsymbol{x}}_j)$. By doing this, the equality in Equation (1.4) is broken and the procedure becomes an approximation that retains the q orthogonal directions with maximum variance, i.e. discarding minimal variance. Such an approximation is clearly most beneficial in situations where the large majority of variance is dominated by just a few eigenvectors, or equivalently when the data truly lie in a lower-dimensional space.

PCA is a descriptive technique which is clearly useful for data compression and visualisation. However, it offers limited inferential value and, due to its empirical nature,

suffers from the poor sample covariance estimation that is apparent in high-dimensional situations (Section 1.2).

PPCA extends PCA to a probabilistic model in which the observed variables may be expressed as linear transformations of lower-dimensional independent Gaussian latent variables plus some additional random noise. For a vector of p observed variables $\boldsymbol{x}_j$ the PPCA model has the form

$$\boldsymbol{x}_j = \boldsymbol{W}\boldsymbol{z}_j + \boldsymbol{\mu} + \boldsymbol{\epsilon} \tag{1.5}$$

where $\boldsymbol{x}_j$ is a p-dimensional observation (column) in the data matrix $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$, and $\boldsymbol{z}_j$ is a column of $\boldsymbol{Z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n)$ containing q-dimensional latent factors $\boldsymbol{z}_j$. Assuming Gaussian distributions for $\boldsymbol{x}_j|\boldsymbol{z}_j$ and $\boldsymbol{\epsilon}$ induces a Gaussian distribution for $\boldsymbol{x}_j$ with covariance matrix

$$\boldsymbol{\Sigma} = \boldsymbol{W}\boldsymbol{W}^\top + \sigma^2 \boldsymbol{I}_{p\times p}. \tag{1.6}$$

Equation (1.6) decomposes $\boldsymbol{\Sigma}$ into a rank-deficient matrix of covariances $\boldsymbol{W}\boldsymbol{W}^\top$ plus a diagonal matrix of independent variance terms for each variable. This decomposition means that the lowest eigenvalue of the estimated covariance matrix is non-zero and equal to σ^2. That implies that the resulting estimator is never singular. The estimator also contains fewer parameters to estimate, since the interactions of the observed variables are determined by the projection from lower-dimensional space.

Although being noted for the form of its covariance estimator, PPCA has received little attention in the covariance estimation literature. This is despite the attractive property that it can model latent dependencies, is always invertible, and in fact has a very convenient closed-form equation for the computation of its inverse that is greatly beneficial when there are a large number of variables (Chapter 3).

Ma [73], Cai et al. [22] consider sparse estimation of the loadings matrix through imposing constraints on the size of its entries. Fan et al. [42, 43, 41] propose a computationally simple estimator by applying adaptive thresholding to a more general model of PPCA, known as an approximate factor model. An approximate factor model (AFM) is obtained if the independence assumption on the columns of $\boldsymbol{Z}$ is relaxed and the isotropic covariance matrix of $\boldsymbol{\epsilon}$ is relaxed to instead have distinct diagonal elements and very small non-diagonal elements. The exact covariance structure that AFM attempts to approximate is one with zero elements in the off-diagonal entries, and this model is known as a factor model.

Factor models have a more complicated overall covariance structure than PPCA:

$$\boldsymbol{\Sigma} = \boldsymbol{W}\boldsymbol{F}\boldsymbol{W}^\top + \boldsymbol{E}, \tag{1.7}$$

where $z_j \sim \mathcal{N}(\mathbf{0}, \boldsymbol{F})$ and $\boldsymbol{E}$ is a diagonal matrix whose entries are not necessarily equal. This covariance structure has received more attention than PPCA in the literature (Section 1.4.2), likely due to its more general nature. However, with more generality comes more computational difficulty. Whereas the model parameters for PPCA have closed-form maximum likelihood solutions [102], factor models do not. In order to overcome this difficulty, various algorithms have been proposed, e.g [96, 5]. The basis of these algorithms is a methodology for performing maximum likelihood inference whilst handling latent variables, known as Expectation-Maximisation (EM) [33]. This technique can also be used to handle missing values in the PPCA model [60]. In the next Section, we introduce the EM algorithm in its general form.

1.3.3 EM algorithm in general

We present the theory of EM as introduced in Bishop [16], but adapted to the case of continuous latent variables. Denote the observed and latent variables as $\boldsymbol{X}$ and $\boldsymbol{Z}$ in a probabilistic model containing a set of parameters $\boldsymbol{\theta}$. We wish to perform inference on $\boldsymbol{\theta}$ by maximising the likelihood

$$\mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta}) = \int \mathrm{p}(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta})\mathrm{d}\boldsymbol{Z}. \tag{1.8}$$

Suppose in this situation that $\mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta})$ is difficult to optimise and that the optimisation of $\mathrm{p}(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta})$ is significantly easier. The aim of an EM algorithm is to maximise the difficult likelihood $\mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta})$ by creating an identity involving $\mathrm{p}(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta})$ and using the property that it is easier to maximise. To derive this identity, we define a distribution over the latent variables $\mathrm{q}(\boldsymbol{Z})$ and, further, a functional $\mathcal{L}$ that depends on both q and $\boldsymbol{\theta}$ as

$$\mathcal{L}(\mathrm{q},\boldsymbol{\theta}) = \int \mathrm{q}(\boldsymbol{Z}) \ln \frac{\mathrm{p}(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta})}{\mathrm{q}(\boldsymbol{Z})}\mathrm{d}\boldsymbol{Z}. \tag{1.9}$$

$\mathcal{L}$ is termed a functional due to the input argument q, which is itself a function (and more specifically a probability distribution). Using the product rule of probability to obtain $\mathrm{p}(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta}) = \mathrm{p}(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta})\mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta})$ and substituting this into Equation (1.9) gives

$$\begin{aligned}\mathcal{L}(\mathrm{q},\boldsymbol{\theta}) &= \int_{\boldsymbol{Z}} \mathrm{q}(\boldsymbol{Z}) \ln \frac{\mathrm{p}(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta})}{\mathrm{q}(\boldsymbol{Z})}\mathrm{d}\boldsymbol{Z} + \int_{\boldsymbol{Z}} \mathrm{q}(\boldsymbol{Z}) \ln \mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta})\mathrm{d}\boldsymbol{Z} &(1.10)\\ &= -\mathrm{KL}(\mathrm{q}||\mathrm{p}_{\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}}) + \ln \mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta}), &(1.11)\end{aligned}$$

where $\mathrm{KL}(\mathrm{q}||\mathrm{p}_{\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}})$ denotes the Kullback-Leibler divergence between q and $\mathrm{p}_{\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}}$ with the subscript $\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}$ explicitly indicating the conditioning. A trivial rearrangement shows us that

$$\ln \mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta}) = \mathcal{L}(\mathrm{q},\boldsymbol{\theta}) + \mathrm{KL}(\mathrm{q}||\mathrm{p}_{\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}}), \tag{1.12}$$

at which point we recall that $\mathrm{KL}(\mathrm{q}||\mathrm{p}_{\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}}) \geq 0$ with equality if and only if $\mathrm{q}(\boldsymbol{Z}) = \mathrm{p}(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta})$. This deduces that $\mathcal{L}(\mathrm{q},\boldsymbol{\theta}) \leq \ln \mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta})$ and so is a lower bound. Using this framework, the EM algorithm iteratively increases this lower bound in such a way that $\ln \mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta})$ also always increases.

In the first step (the E step), we keep the current parameter values $\boldsymbol{\theta}^{\mathrm{old}}$ fixed and maximise $\mathcal{L}(\mathrm{q},\boldsymbol{\theta}^{\mathrm{old}})$ with respect to $\mathrm{q}(\boldsymbol{Z})$. Observing Equation (1.11) and noting that $\ln \mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta}^{\mathrm{old}})$ does not depend on $\mathrm{q}(\boldsymbol{Z})$ shows us that this maximum occurs when $\mathrm{KL}(\mathrm{q}||\mathrm{p}_{\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}^{\mathrm{old}}}) = 0$, which can only be when $\mathrm{q}(\boldsymbol{Z}) = \mathrm{p}(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}^{\mathrm{old}})$. Thus, the point at which $\mathcal{L}(\mathrm{q},\boldsymbol{\theta}^{\mathrm{old}})$ is maximised with respect to $\mathrm{q}(\boldsymbol{Z})$ is exactly when it equals $\ln \mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta}^{\mathrm{old}})$.

The next step (the M step) consists of fixing $\mathrm{q}(\boldsymbol{Z})$ and then maximising $\mathcal{L}(\mathrm{q},\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ to give new parameter estimates $\boldsymbol{\theta}^{\mathrm{new}}$. By definition $\mathcal{L}(\mathrm{q},\boldsymbol{\theta}^{\mathrm{old}}) \leq \mathcal{L}(\mathrm{q},\boldsymbol{\theta}^{\mathrm{new}})$ and so $\ln \mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta}^{\mathrm{old}}) \leq \ln \mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta}^{\mathrm{new}})$ (unless it is already at the maximum). But, in addition to the increase of $\mathcal{L}(\mathrm{q},\boldsymbol{\theta})$ we also have that $\mathrm{KL}(\mathrm{q}||\mathrm{p}_{\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{\theta}^{\mathrm{new}}})$ is no longer zero. The result of this is that not only does $\ln \mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta})$ increase, but it increases at least as much as the lower bound $\mathcal{L}(\mathrm{q},\boldsymbol{\theta})$.

These E and M steps are then iterated until convergence, at which point $\boldsymbol{\theta}^{\mathrm{old}} = \boldsymbol{\theta}^{\mathrm{new}}$, $\mathcal{L}(\mathrm{q},\boldsymbol{\theta}^{\mathrm{new}}) = \ln \mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta}^{\mathrm{new}})$ and therefore the value of $\boldsymbol{\theta}$ that maximises $\mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta})$ has been found. Due to the assumption that $\mathrm{p}(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\theta})$ is significantly easier to optimise than $\mathrm{p}(\boldsymbol{X}|\boldsymbol{\theta})$, these iterative steps are much simpler to perform than the original optimisation. One caveat however, is that the EM solution is not guaranteed to converge to the global maximum.

Note that in the above formulation, missing values may be treated as unobserved latent variables, being absorbed into the latent variable term $\boldsymbol{Z}$ and updated accordingly. We present multiple implementations of this method for PPCA as a tool for covariance matrix estimation in the presence of missing values in Chapter 3.

1.4 Bayesian approaches

Here, we introduce the Bayesian approach to covariance estimation.

1.4.1 Bayesian inference for covariance matrices

In Bayesian statistics, the model parameters of interest are treated as unobservable random quantities with an underlying probability distribution that reflects our uncertainty associated with them. Having observed $\boldsymbol{X}$ with covariance matrix $\boldsymbol{\Sigma}$, Bayes' Theorem can be stated as

$$\mathrm{p}(\boldsymbol{\Sigma}|\boldsymbol{X}) = \frac{\mathrm{p}(\boldsymbol{X}|\boldsymbol{\Sigma})\mathrm{p}(\boldsymbol{\Sigma})}{\mathrm{p}(\boldsymbol{X})}. \tag{1.13}$$

The term $\mathrm{p}(\boldsymbol{\Sigma}|\boldsymbol{X})$ in Equation (1.13) is known as the posterior distribution of $\boldsymbol{\Sigma}$. The posterior provides us with a distribution on $\boldsymbol{\Sigma}$ after having observed $\boldsymbol{X}$, which contains all of the information about $\boldsymbol{\Sigma}$. Summaries of the posterior distribution are then used as point estimates for $\boldsymbol{\Sigma}$. The posterior expectation $\mathbb{E}[\boldsymbol{\Sigma}|\boldsymbol{X}]$, for instance, minimises the MSE.

1.4.2 Covariance matrix prior distribution

The term $\mathrm{p}(\boldsymbol{\Sigma})$ in Equation (1.13) is known as the prior distribution and it expresses a belief about $\boldsymbol{\Sigma}$ prior to observing the data. The prior can be used to impart additional information about $\boldsymbol{\Sigma}$ into the model, which may come from a variety of sources such as expert opinion or knowledge of previous experiments. The prior introduces a term into the posterior that is often used as a restriction for certain values of $\boldsymbol{\Sigma}$. This can take the form of a constraining parametric distributional assumption, or even element-wise penalties on large values of $\boldsymbol{\Sigma}$. This restriction plays an important role in high-dimensional settings, in which sample sizes are low and more information is leveraged from the prior to impact the posterior.

However, the choice of prior distribution, and therefore the regularisation that is performed, is subjective. Selecting a prior for the covariance matrix is difficult but it must ensure that the matrices that it generates are positive definite. A common choice is the inverse-Wishart distribution, which can be defined as

$$\mathrm{p}(\boldsymbol{\Sigma}|\nu,\boldsymbol{\Psi}) = 2^{-\frac{\nu p}{2}}\Gamma_P^{-1}\left(\frac{\nu}{2}\right) |\, \boldsymbol{\Psi} \,|^{\frac{\nu}{2}} |\, \boldsymbol{\Sigma} \,|^{-\frac{\nu+p+1}{2}} \exp\left\{-\frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Psi}\boldsymbol{\Sigma}^{-1}\right)\right\} \tag{1.14}$$

for degrees of freedom parameter $\nu > p-1$ and $(p\times p)$ positive-definite scale matrix $\boldsymbol{\Psi}$. The mean of this distribution is given by $\mathbb{E}(\boldsymbol{\Sigma}|\nu,\boldsymbol{\Psi}) = \frac{\boldsymbol{\Psi}}{\nu-p-1}$ whenever $\nu > p+1$, the existence of which we assume hereafter (i.e. $\nu > p+1$). The element-wise variances are given by $\mathrm{Var}(\Sigma_{ij}|\nu,\boldsymbol{\Psi}) = \frac{(\nu-p+1)\psi_{ij}^2+(\nu-p-1)\psi_{ii}\psi_{ij}}{(\nu-p)(\nu-p-1)^2(\nu-p-3)}$.To aid interpretation of the hyper-parameters of this density, we instead adopt the mean-centred parametrisation of the Inverse-Wishart distribution used by Hannart and Naveau [52]. The new parametrisation is obtained through the following bijective transformation:

$$(\alpha,\boldsymbol{\Delta}) = \left(\frac{\nu-p-1}{n+\nu-p-1}, \frac{\boldsymbol{\Psi}}{\nu-p-1}\right) \quad \Leftrightarrow \quad (\nu,\boldsymbol{\Psi}) = \left(\frac{\alpha n}{1-\alpha}+p+1, \frac{\alpha n}{1-\alpha}\boldsymbol{\Delta}\right), \tag{1.15}$$

where $\alpha \in (0,1)$ and $\boldsymbol{\Delta}$ is positive definite. By definition, we now have $\mathbb{E}(\boldsymbol{\Sigma}|\alpha,\boldsymbol{\Delta}) = \boldsymbol{\Delta}$ and so the interpretation of $\boldsymbol{\Delta}$ as the prior mean of $\boldsymbol{\Sigma}$ is clear. The element-wise prior variances are now given by $\mathrm{Var}(\Sigma_{ij}|\alpha,\boldsymbol{\Delta}) \approx \frac{1-\alpha}{\alpha n}\left(\Delta_{ij}^2+\Delta_{ii}\Delta_{jj}\right)$ and so α can be seen to control the element-wise prior precision.

This choice of prior is practical because it guarantees positive definiteness and it is conjugate to the multivariate Normal distribution, the latter being a frequent assumption for the likelihood $\mathrm{p}(\boldsymbol{X}|\boldsymbol{\Sigma})$. The term conjugate means that this pair of prior and likelihood evoke a tractable posterior distribution of the same form as the prior. A major benefit of the inverse-Wishart prior is therefore computational. Additional interpretation of the hyperparameters can be extracted from the posterior distribution, $\mathrm{p}(\boldsymbol{\Sigma}|\boldsymbol{X},\alpha,\boldsymbol{\Lambda}) = \text{inv-Wishart}\left((2-\alpha)^{-1},\alpha\boldsymbol{\Lambda}+(1-\alpha)\boldsymbol{S}\right)$, so that it can be seen that the posterior expectation is a linear shrinkage estimator $\mathbb{E}(\boldsymbol{\Sigma}|\boldsymbol{X},\alpha,\boldsymbol{\Lambda}) = \alpha\boldsymbol{\Lambda}+(1-\alpha)\boldsymbol{S}$. From this perspective, it can now be seen that α controls the degree to which we favour our prior expectation $\boldsymbol{\Lambda}$ over the sample covariance matrix $\boldsymbol{S}$. This provides guidance on setting these parameters — $\boldsymbol{\Lambda}$ should reflect an expectation about $\boldsymbol{\Sigma}$ and α our confidence in the expectation.

It is worth noting that hyperparameters which define the distributions of other parameters (in the conjugate model above, these were $\alpha,\boldsymbol{\Lambda}$) may themselves be assigned prior distributions, referred to as hyper-priors. The extent to which this affects the complexity of the model and the resulting difficulty of inference depends upon the particular scenario and hyper-priors that have been assumed. In the model above, setting prior distributions on the hyperparameters breaks the conjugacy property and the posterior distribution is no longer available in closed-form, allowing greater flexibility at the cost of complicating the inference.

Bayesian frameworks for factor models have also been a popular feature in the literature for some time, e.g. [2, 51]. Priors that have been used for the parameters in Equation (1.7) include; treating $\boldsymbol{W}$ as a vector and assuming it has a multivariate normal distribution [4], inverse Wishart for $\boldsymbol{W}\boldsymbol{F}\boldsymbol{W}^\top$ [84], inverse gamma for the non-zero elements of $\boldsymbol{E}$ [84, 11, 90], and spike and slab priors on the elements of $\boldsymbol{W}$ [8, 76] with Indian buffet process on an infinite-column W [90].

The flexibility that comes with the factor model and these prior distributions is marred by the increased computational intensity that comes with performing inference. These approaches obtain an intractable posterior distribution that requires computationally intensive sampling or approximate inference in order to obtain parameter estimates. Sampling has the drawback that it is slow and may require large amounts of computational memory as well as storage, whilst approximate inference has no guaranteed global convergence.

Other possible choices for the covariance prior include a reference prior [112] or a hierarchical model for the covariance matrix [30]. These approaches also require computationally intensive sampling or approximate inference. In Section 1.4.4, we recap the general form of an approximate inference tool that can be used to evaluate some of these more complex models.

1.4.3 Competing models

Suppose now that we have more than one model for $\boldsymbol{\Sigma}$ that leads to different posterior inferences. In the simple conjugate case outlined above, this could be models $\mathcal{M}_1 = \{\alpha_1, \boldsymbol{\Lambda}_1\}$ and $\mathcal{M}_2 = \{\alpha_2, \boldsymbol{\Lambda}_2\}$. The uncertainty surrounding these models is contained within $\mathrm{p}(\mathcal{M}_1)$ and $\mathrm{p}(\mathcal{M}_2)$, their prior probability distributions. Upon observing $\boldsymbol{X}$, Bayesian inference for these models can be performed in accordance with Bayes theorem, using $\mathrm{p}(\mathcal{M}_1|\boldsymbol{X}) \propto \mathrm{p}(\mathcal{M}_1)\mathrm{p}(\boldsymbol{X}|\mathcal{M}_1)$ and $\mathrm{p}(\mathcal{M}_2|\boldsymbol{X}) \propto \mathrm{p}(\mathcal{M}_2)\mathrm{p}(\boldsymbol{X}|\mathcal{M}_2)$. The likelihoods $\mathrm{p}(\boldsymbol{X}|\mathcal{M}_1)$ and $\mathrm{p}(\boldsymbol{X}|\mathcal{M}_2)$ represent the probabilities of observing the data assuming each of the competing models, termed the marginal likelihood. The ratio between these quantities is known as the Bayes factor [62], which provides the relative likelihood of the observed data if it truly were generated by each model. Selecting the model whose Bayes factor is greatest then represents one method for model selection.

The term marginal likelihood comes from the formula to evaluate it, for $\mathcal{M}_1$ we have $\mathrm{p}(\boldsymbol{X}|\mathcal{M}_1) = \int \mathrm{p}(\boldsymbol{X}|\boldsymbol{\Sigma}, \mathcal{M}_1)\mathrm{p}(\boldsymbol{\Sigma}|\mathcal{M}_1)\mathrm{d}\boldsymbol{\Sigma}$ and similarly for $\mathcal{M}_2$. Thus, the marginal likelihood given $\mathcal{M}_1$ is obtained by integrating out the parameter of interest $\boldsymbol{\Sigma}$ from the joint density $\mathrm{p}(\boldsymbol{X}, \boldsymbol{\Sigma}|\mathcal{M}_1)$. If we consider the full set of competing models which exhaust the model space, then $\mathrm{p}(\boldsymbol{X}|\mathcal{M}_d)$, $d = 1, \ldots, D$ can be seen as a function to be maximised over the $\mathcal{M}_d$, yielding the model with highest marginal likelihood and equivalently the model whose Bayes factor exceeds 1 when compared with all other models. This type of marginal likelihood maximisation is known as empirical Bayes [9], or type-II maximum likelihood [7], and is another method for model selection. It is typically performed on the log scale due to simplicity and numerical stability. This empirical Bayes approach is applied to the conjugate model by Hannart and Naveau [52] for $\boldsymbol{\Sigma}$ having first fixed $\boldsymbol{\Lambda}$, so that the model space to maximise over is reduced to $\mathcal{M}_\alpha = \{\alpha \in (0,1)\}$.

Instead of selecting a particular model, we may wish to incorporate the model uncertainty into our posterior distribution for $\boldsymbol{\Sigma}$, which is obtained as

$$\mathrm{p}(\boldsymbol{\Sigma}|\boldsymbol{X}) = \sum_{d=1}^{D} \mathrm{p}(\boldsymbol{\Sigma}|\boldsymbol{X}, \mathcal{M}_d)\mathrm{p}(\mathcal{M}_d|\boldsymbol{X}). \tag{1.16}$$

This marginal posterior is then a mixture distribution of each posterior for a specified model weighted by its corresponding model posterior probability. This technique is known as Bayesian Model Averaging [57]. We use this idea to average over hyperparameters in the conjugate distribution on $\boldsymbol{\Sigma}$ in Chapter 2.

1.4.4 Variational inference

Recall that when computing the marginal likelihood $\mathrm{p}(\boldsymbol{X})$, we had to marginalise the parameter of interest from its joint density with $\boldsymbol{X}$ conditional on any assumed model

parameters. In simple models, this integration is available in closed-form, but for many more complex models, it is not. In addition, the numerical integration techniques that lead to exact solutions of the integral may be too cumbersome in practice, e.g. by requiring infeasible computational resources. In this situation, one might wish to be pragmatic and obtain an approximate solution to evaluating the marginal likelihood. Here, we introduce an approximate Bayesian inference technique akin to the EM algorithm in Section 1.3.3 known as variational inference [17].

In contrast to the EM algorithm, the Bayesian framework allows the parameters $\boldsymbol{\theta}$ to now be viewed as a collection of random variables, each with their own associated prior distributions, and therefore absorbed into the set of unobserved variables $\boldsymbol{Z}$. If we now wish to perform inference on $\boldsymbol{Z}$, then it is first necessary to obtain their respective posterior distributions. For models of sufficient complexity, the posterior $\mathrm{p}(\boldsymbol{Z}|\boldsymbol{X})$ is intractable and so this is not possible. We therefore seek to find an appropriate approximation to it that has a tractable form to perform inference.

In a similar fashion to the EM formulation, the log-marginal likelihood $\ln \mathrm{p}(\boldsymbol{X})$ can be given as

$$\ln \mathrm{p}(\boldsymbol{X}) = \mathcal{L}(\mathrm{q}) + \mathrm{KL}(\mathrm{q}||\mathrm{p}), \tag{1.17}$$

where

$$\mathcal{L}(\mathrm{q}) = \int \mathrm{q}(\boldsymbol{Z}) \ln \left(\frac{\mathrm{p}(\boldsymbol{X}, \boldsymbol{Z})}{\mathrm{q}(\boldsymbol{Z})} \right) \mathrm{d}\boldsymbol{Z} \tag{1.18}$$

$$\mathrm{KL}(\mathrm{q}||\mathrm{p}) = -\int \mathrm{q}(\boldsymbol{Z}) \ln \left(\frac{\mathrm{p}(\boldsymbol{Z}|\boldsymbol{X})}{\mathrm{q}(\boldsymbol{Z})} \right) \mathrm{d}\boldsymbol{Z}. \tag{1.19}$$

Maximisation of $\mathcal{L}(\mathrm{q})$ with respect to $\mathrm{q}(\boldsymbol{Z})$ is again equivalent to minimising the KL divergence between $\mathrm{q}(\boldsymbol{Z})$ and the posterior of the latent variables $\mathrm{p}(\boldsymbol{Z}|\boldsymbol{X})$, which occurs when the two distributions are equivalent. However, we have assumed that $\mathrm{p}(\boldsymbol{Z}|\boldsymbol{X})$ is intractable — rendering a standard EM approach infeasible.

One way to overcome this challenge is to restrict $\mathrm{q}(\boldsymbol{Z})$ to be a more convenient family of distributions. The primary constraint is that $\mathrm{q}(\boldsymbol{Z})$ must be tractable. After that, it is beneficial to choose a flexible family of distributions so as to achieve the best approximation. We focus on a factorised distribution for $\mathrm{q}(\boldsymbol{Z})$ known as a mean field approximation [85]. This takes the form

$$\mathrm{q}(\boldsymbol{Z}) = \prod_{l=1}^{L} \mathrm{q}_l(\boldsymbol{Z}_l), \tag{1.20}$$

in which it is assumed that the latent variables $\boldsymbol{Z}$ may be partitioned into disjoint groups $\boldsymbol{Z}_l$ where $l = 1, \ldots, L$. Maximisation of Equation (1.17) is now done with respect to each $\mathrm{q}_l(\boldsymbol{Z}_l)$. The optimal solutions are found by substituting Equation (1.20) into (1.18)

[16] and take the form

$$\ln \mathrm{q}_l^*(\boldsymbol{Z}_l) = \mathbb{E}_{m \neq l}[\ln \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z})] + \text{const}, \tag{1.21}$$

where $\mathbb{E}_{m \neq l}[\cdot]$ denotes an expectation taken with respect to all indices of $\boldsymbol{Z}_m$ for $m = 1, \ldots, L$ with $m \neq l$. The optimal $\mathrm{q}_l(\boldsymbol{Z}_l)$ is thus found by marginalising all other partitions of $\boldsymbol{Z}$ from the joint distribution $\mathrm{p}(\boldsymbol{X}, \boldsymbol{Z})$. The exact form of the solutions are context-specific and clearly depend on the other factors $\boldsymbol{Z}_{m \neq l}$. The VB algorithm is completed by cycling through Equation (1.21) for each factor until convergence, which is guaranteed but may not be global. The lower bound can be a useful quantity to check the convergence of different estimates.

1.5 Inverse covariance matrices

A parameter related to the covariance matrix that is also of significant interest is its inverse $\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$, known as the precision matrix. For the purpose of this thesis, we introduce the importance of the precision matrix through a framework known as Gaussian Graphical Models (GGMs) [110]. GGMs are a class of graphical model whose underlying assumption about the data generating process is a Gaussian distribution. More formally, denote $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to be the undirected graph with vertices $\mathcal{V} = \{1, \ldots, p\}$ and edges $\mathcal{E} = (e_{ij})$, with e_{ij} equal to 1 or 0 depending upon if vertices i and j are adjacent in $\mathcal{G}$, or not, respectively. The defining property of a GGM is that if $\boldsymbol{x} = (x_1, \ldots, x_p) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ then $\omega_{ij} = 0$ if and only if x_i and x_j are conditionally independent given $x_{\mathcal{V} \setminus \{i,j\}}$, where $\mathcal{V} \setminus \{i, j\}$ denotes the set $\mathcal{V}$ excluding elements i and j. Therefore, the edge set $\mathcal{E}$ is defined by variables which are not pairwise conditionally independent given all other variables, and moreover that this dependence structure is fully described by the elements of the inverse covariance matrix $\boldsymbol{\Omega}$.

A more interpretable quantity that can be defined using $\boldsymbol{\Omega}$ is the partial correlation coefficient

$$r_{ij} = -\frac{\omega_{ij}}{\sqrt{\omega_{ii}\omega_{jj}}}. \tag{1.22}$$

This represents the correlation between x_i and x_j conditional upon all other $x_{\mathcal{V} \setminus \{i,j\}}$. From Equation (1.22) it can be seen that r_{ij} contains the dependence component of ω_{ij}, with the other terms ω_{ii} and ω_{jj} being precision components. This is completely analogous to the marginal correlation and variance decomposition of the covariance statistic. Using this intuition, it can also be seen that $\omega_{ij} = 0$ can only occur whenever $r_{ij} = 0$. It is thus sufficient to check conditional independence in a GGM by inspecting each r_{ij}.

Thus, when estimating precision matrices in GGMs, it is of great importance to infer those values which are truly 0 or not in order to correctly capture the graphical model. It is this condition that creates the biggest discriminant between covariance and precision matrix estimation, and which significantly changes the nature of inference. For this reason, precision matrix estimation is scarcely mentioned in this thesis. When it is, the problem is constrained to first obtaining a dense estimate of $\mathbf{\Sigma}$ and then applying a testing procedure to its inverse in order to find those $r_{ij} = 0$, rather than estimating $\mathbf{\Omega}$ directly. Precision matrix estimation is a large field of literature, the reader is referred to Dempster [32] for an introduction to the problem, including the sparse [45] and dense [106] estimation frameworks.

1.6 Contributions and outline

This thesis provides contributions to: (i) shrinkage covariance estimation using multiple shrinkage targets, (ii) algorithms for probabilistic principal component analysis, (iii) computationally efficient software packages for high-dimensional covariance matrix estimation and, (iv) the application of linear shrinkage covariance estimators with multiple targets to time series data.

This thesis is organised as follows. In Chapter 2, we revisit linear shrinkage estimation and provide a novel Bayesian extension that allows for multiple targets to be included in a computationally efficient framework, named Target-Averaged linear Shrinkage (TAS) [50]. We demonstrate the performance of TAS in comparison to existing single-target shrinkage methods using both model-based and predictive validation simulation protocols. Using a publicly available pan-cancer protein expression dataset, we show how TAS can easily incorporate multiple datasets as prior information using 31 shrinkage targets. Finally, we show how the shrinkage weights for each cancer type reflects putative similarities reported in the literature.

In Chapter 3 we turn our attention towards PPCA as a method for covariance matrix estimation. We present some attractive properties of its induced covariance structure that appear to not have received much attention in the literature, particularly its ability to handle missing values. We then introduce and fully derive four different algorithms for performing PPCA in the presence of missing values. The mathematical derivation of these methods is not present in the current literature and thus itself is a novel contribution. We present a numerical comparison of their performance in terms of estimation accuracy and timing for data simulated from the PPCA model. We select the best-performing algorithm to compare against TAS in the predictive validation simulation protocol from Chapter 2. Finally, we demonstrate some functionality of the

software package `pcaNet` and in particular how to reconstruct a network from a real Arabidopsis thaliana dataset using PPCA.

In Chapter 4 we conduct a real data analysis on a clinical trial dataset for patients with traumatic brain injury (TBI), in which inflammatory proteins called cytokines have had their expression measured over time. We describe the methodology of the trial and present an exploratory analysis. We then proceed with a multivariate analysis by applying TAS using multiple targets from previous time points. We show that this facilitates downstream analysis by characterising stable groups and interactions of cytokines over time using cluster analysis and network reconstruction.

We finish with a concluding chapter that summarises the work that is presented in this thesis and provides an outlook for future work.

Chapter 2

Target-Averaged linear Shrinkage Estimation

2.1 Introduction

Covariance matrix estimation plays a central role in statistical analyses. In molecular biology, for instance, covariance estimation facilitates the identification of dependence structures between molecular variables that shed light on the underlying molecular or cellular processes [46, 95]. Because high-throughput omics experiments typically measure a large number of molecular variables (e.g. gene expression) on relatively few samples, the sample covariance is generally singular or ill-conditioned. This means that the sample covariance matrix suffers from high estimation error that can affect subsequent numerical tasks, such as computing its useful matrix inverse (precision matrix). This problem has been well studied [31, 88, 41, 39] and many solutions have been proposed over the last decades. These usually modify the sample covariance so as to stabilise estimation. Some solutions adopt sparse, lasso-type, regularisation that enforces most entries of the estimated covariance matrix to be equal to zero [12, 21, 14], whereas other solutions adopt non-sparse, ridge-type, regularization that does not yield zero entries [70, 108, 111, 106]. The choice of a particular form of regularization typically depends on the statistical goals and computational constraints [13].

Single-target linear shrinkage (STS) estimators are ridge-type estimators, which are defined as a convex combination between the sample covariance matrix and a pre-specified positive definite *target* matrix. These estimators are very popular in practice due to their simplicity, ease of interpretation and computational efficiency [95]. For these reasons, they have also been theoretically well studied [70, 103, 44, 59, 24]. The performance of STS estimators, however, is highly dependent on the choice of an appropriate target matrix (see Section 2.6). Different target matrices have been proposed

in the literature, but the choice is ultimately guided by the application and the presumed structure of the unknown covariance matrix [39].

Despite a large literature, surprisingly little has been done to extend STS estimators to allow shrinkage towards multiple shrinkage targets. Bartz et al. [6] and Lancewicki and Aladjem [64] have proposed multi-target linear shrinkage estimators that represent optimal convex combinations, in the mean square sense, between the sample covariance matrix and multiple shrinkage targets. Ikeda et al. [59] propose a linear shrinkage estimator with two target matrices and derive its shrinkage intensities using a decision theoretic framework for two simple common targets under Bayesian assumptions. Unfortunately, none of these methods are implemented in available software.

In this chapter, we introduce a linear shrinkage estimator that can accommodate multiple general shrinkage target matrices, and thereby incorporate uncertainty about the target choice. The proposed estimator is obtained within a conjugate Bayesian framework which is computationally efficient, even when the number of samples, variables or shrinkage targets is relatively large. Using both simulated and real data, we show that the multi-target estimator is less sensitive to the misspecification of some of its targets and can outperform state-of-the-art (non-parametric) STS estimators. Moreover, we show that the target-specific weights can be usefully interpreted. We apply our approach to a pan-cancer proteomic data set where we illustrate how multiple sources of external information, obtained from different cancer types, can be incorporated within the target set. In particular, it is shown that target-specific shrinkage weights can provide insights into the differences and similarities between cancer types. The method proposed in this paper is implemented as an `R` package and freely available at http://github.com/HGray384/TAS.

This chapter is organised as follows. In Section 2.2, we describe STS estimators and in Section 2.3 we introduce its Bayesian counterpart. We present an extension to this to allow for multiple shrinkage targets in Section 2.4. Section 2.5 discusses the potential target matrices to consider when using multiple targets. Section 2.6 and 2.7 compare the performance of the proposed estimator to state-of-the-art STS estimators using simulated and real data, respectively. We apply our approach in Section 2.8 to a pan-cancer proteomic data set from The Cancer Proteome Atlas. In Section 2.9 we mention some details about the computational implementation of the model. Last, Section 2.10 discusses linear shrinkage estimation by means of multiple targets and concludes on future directions. All code used to produce the results shown in this chapter is available at http://github.com/HGray384/TAS-paper-code.

2.2 Single-target linear shrinkage covariance estimation

Let $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ be a matrix containing n independent observations drawn from a p-variate Normal distribution with zero mean vector and positive definite covariance matrix $\boldsymbol{\Sigma}$ (hereby denoted $\boldsymbol{\Sigma} \succ 0$). The log-likelihood of $\boldsymbol{X}$ is then

$$\log \mathrm{p}(\boldsymbol{X}|\boldsymbol{\Sigma}) \propto \log|\boldsymbol{\Sigma}^{-1}| - \mathrm{Tr}[\boldsymbol{S}\boldsymbol{\Sigma}^{-1}], \tag{2.1}$$

where $\boldsymbol{S} = \boldsymbol{X}\boldsymbol{X}^\top/n$. The maximum likelihood estimator (MLE) induced by Equation (2.1) is $\hat{\boldsymbol{\Sigma}} = \boldsymbol{S}$, which is ill-conditioned or singular whenever n is small relative to p.

This section describes the class of single-target linear shrinkage estimators as a solution to this problem, as well as a Bayesian counterpart which we generalise to accommodate multiple shrinkage target matrices. The latter provides a more flexible framework while retaining computational efficiency.

Recall from Section 1.3.1, that an STS estimator is defined as a weighted average between the MLE and a single pre-specified matrix $\boldsymbol{\Delta}$, often referred to as the *shrinkage target*, i.e.:

$$\hat{\boldsymbol{\Sigma}} = \alpha\boldsymbol{\Delta} + (1-\alpha)\boldsymbol{S}, \quad \text{with } \alpha \in (0,1) \text{ and } \boldsymbol{\Delta} \succ 0. \tag{2.2}$$

This estimator can be thought of in terms of a bias-variance trade-off [70], which is calibrated through the shrinkage intensity or weight α. Values of α close to one define a low-variance but high-bias estimator ($\hat{\boldsymbol{\Sigma}} \approx \boldsymbol{\Delta}$), whilst values of α closer to zero define a low-bias but high-variance estimator ($\hat{\boldsymbol{\Sigma}} \approx \boldsymbol{S}$). Following this line of thought, alternative definitions of Equation (2.2) use the unbiased estimator $\frac{n}{n-1}\boldsymbol{S}$ instead of the MLE. In any case, the optimal balance for this trade-off often lies away from these limiting cases and analytical solutions have been proposed under different assumptions [95, 24, 44, 103].

The estimator in Equation (2.2) can also be viewed as a penalised MLE under a specific ridge-type penalty [106]. To derive this, Equation (2.1) is modified by substituting $\boldsymbol{S}$ for $(1-\alpha)\boldsymbol{S}$ and adding the penalty term $\mathrm{Tr}[\boldsymbol{\Delta}\boldsymbol{\Sigma}^{-1}]$ with penalty parameter α so that it becomes proportional to

$$\log|\boldsymbol{\Sigma}^{-1}| - (1-\alpha)\mathrm{Tr}[\boldsymbol{S}\boldsymbol{\Sigma}^{-1}] - \alpha\mathrm{Tr}[\boldsymbol{\Delta}\boldsymbol{\Sigma}^{-1}]. \tag{2.3}$$

The maximum of this penalised likelihood is achieved by the shrinkage estimator in Equation (2.2) [106]. With high regularisation ($\alpha \approx 1$) the resulting log-likelihood becomes that of jointly independent Gaussian variables with covariance matrix $\boldsymbol{\Delta}$, and vice versa with covariance matrix $\boldsymbol{S}$ for $\alpha \approx 0$, matching the intuition of Equation (2.2).

2.3 Conjugate Bayesian framework

In a Bayesian framework, an STS estimator of the covariance matrix can be obtained in closed-form by placing an inverse-Wishart prior on $\boldsymbol{\Sigma}$ [23, 52]. Adopting the parametrisation of Hannart and Naveau [52] (Section 1.15), we denote $\boldsymbol{\Sigma}|\alpha,\boldsymbol{\Delta} \sim \text{Inv-Wishart}(\alpha,\boldsymbol{\Delta})$ with $\alpha \in (0,1)$ and $\boldsymbol{\Delta} \succ 0$. Under this parametrisation it follows that $\mathbb{E}(\boldsymbol{\Sigma}|\alpha,\boldsymbol{\Delta}) = \boldsymbol{\Delta}$ and

$$\mathbb{E}(\boldsymbol{\Sigma}|\boldsymbol{X},\alpha,\boldsymbol{\Delta}) = \alpha\boldsymbol{\Delta} + (1-\alpha)\boldsymbol{S}, \tag{2.4}$$

thereby making explicit that the marginal posterior expectation $\mathbb{E}(\boldsymbol{\Sigma}|\boldsymbol{X},\alpha,\boldsymbol{\Delta})$ of $\boldsymbol{\Sigma}$ is an STS estimator with shrinkage target equal to the prior expectation of $\boldsymbol{\Sigma}$.

In recent work, Hannart and Naveau [52] introduced a general framework for empirical Bayes estimation (through marginal likelihood maximisation) of α and $\boldsymbol{\Delta}(\boldsymbol{\theta})$ when the shrinkage target is parametrised in terms of a low-dimensional vector $\boldsymbol{\theta}$. In the particular case where the shrinkage target is fully specified a priori, the problem of estimating α reduces to the optimisation of a univariate concave objective function. Hannart and Naveau [52] observed that the empirical Bayes estimate of α is often close to the value that minimises the mean square error. However, the uncertainty regarding this estimate can be large in some cases (see Appendix A.1).

2.4 Incorporating uncertainty about α and $\boldsymbol{\Delta}$

In this section, we hierarchically extend the conjugate model introduced in Section 2.3 by placing independent hyper-prior distributions on α and $\boldsymbol{\Delta}$, such that the posterior expectation of $\boldsymbol{\Sigma}$ remains available in closed-form. We place a uniform discrete prior on α over the support $\mathcal{A} = \{a_1,\ldots,a_K\}$, where $0 < a_1 < \cdots < a_K < 1$ and $\mathrm{p}(\alpha = a_k) = 1/K$ for $k \in \{1,\ldots,K\}$. Similarly, we place a uniform discrete prior on $\boldsymbol{\Delta}$ over the support $\mathcal{D} = \{\boldsymbol{D}_1,\ldots,\boldsymbol{D}_L\}$, hereafter referred to as the *target set*. We assume that $\boldsymbol{D}_l \succ 0$ and $\mathrm{p}(\boldsymbol{\Delta} = \boldsymbol{D}_l) = 1/L$ for $l \in \{1,\ldots,L\}$. Under these priors, the marginal posterior expectation of $\boldsymbol{\Sigma}$ is given by

$$\mathbb{E}[\boldsymbol{\Sigma}|\boldsymbol{X}] = \sum_{l=1}^{L}\sum_{k=1}^{K} \mathbb{E}[\boldsymbol{\Sigma}|\boldsymbol{X},\alpha = a_k,\boldsymbol{\Delta} = \boldsymbol{D}_l]\mathrm{p}(\alpha = a_k,\boldsymbol{\Delta} = \boldsymbol{D}_l|\boldsymbol{X}), \tag{2.5}$$

where

$$\mathrm{p}(\alpha = a_k,\boldsymbol{\Delta} = \boldsymbol{D}_l|\boldsymbol{X}) = \frac{\mathrm{p}(\boldsymbol{X}|\alpha = a_k,\boldsymbol{\Delta} = \boldsymbol{D}_l)\mathrm{p}(\alpha = a_k)\mathrm{p}(\boldsymbol{\Delta} = \boldsymbol{D}_l)}{\sum_{q=1}^{L}\sum_{k=1}^{K}\mathrm{p}(\boldsymbol{X}|\alpha = a_k,\boldsymbol{\Delta} = \boldsymbol{D}_q)\mathrm{p}(\alpha = a_k)\mathrm{p}(\boldsymbol{\Delta} = \boldsymbol{D}_q)}. \tag{2.6}$$

and

$$\mathrm{p}(\boldsymbol{X}|\alpha,\boldsymbol{\Delta}) = \frac{\Gamma_p\left\{\frac{1}{2}\left(\frac{n}{1-\alpha}+p+1\right)\right\}\left|\frac{\alpha}{1-\alpha}\boldsymbol{\Delta}\right|^{\frac{\alpha n}{1-\alpha}+p+1}}{(n\pi)^{\frac{np}{2}}\Gamma_p\left\{\frac{1}{2}\left(\frac{\alpha n}{1-\alpha}+p+1\right)\right\}\left|\boldsymbol{S}+\frac{\alpha}{1-\alpha}\boldsymbol{\Delta}\right|^{\frac{n}{1-\alpha}+p+1}}. \tag{2.7}$$

In fact, under the discrete uniform prior assumptions for α and $\boldsymbol{\Delta}$, Equation (2.6) simply becomes

$$\mathrm{p}(\alpha = a_k, \boldsymbol{\Delta} = \boldsymbol{D}_l|\boldsymbol{X}) = \frac{\mathrm{p}(\boldsymbol{X}|\alpha = a_k, \boldsymbol{\Delta} = \boldsymbol{D}_l)}{\sum_{q=1}^{L}\sum_{k=1}^{K}\mathrm{p}(\boldsymbol{X}|\alpha = a_k, \boldsymbol{\Delta} = \boldsymbol{D}_q)}. \tag{2.8}$$

Note that Equation (2.5) is akin to a model average estimator [57] (Section 1.4.3), combining individual STS estimators obtained from the statistical models indexed by the support of $(\alpha,\boldsymbol{\Delta})$. The estimator in Equation (2.5) can also be re-formulated as

$$\mathbb{E}[\boldsymbol{\Sigma}|\boldsymbol{X}] = \sum_{l=1}^{L}\sum_{k=1}^{K}[a_k\boldsymbol{D}_l + (1-a_k)\boldsymbol{S}]\,\mathrm{p}(\alpha = a_k, \boldsymbol{\Delta} = \boldsymbol{D}_l|\boldsymbol{X}) \tag{2.9}$$

$$= \sum_{l=1}^{L} w_l\boldsymbol{D}_l + \left(1 - \sum_{l=1}^{L} w_l\right)\boldsymbol{S}, \tag{2.10}$$

where

$$w_l = \sum_{k=1}^{K} a_k\mathrm{p}(\alpha = a_k, \boldsymbol{\Delta} = \boldsymbol{D}_l|\boldsymbol{X}) \tag{2.11}$$

is a target-specific posterior weight synthesising the contribution of the target $\boldsymbol{D}_l$ relative to the target set $\mathcal{D}$. This reformulation shows that $\mathbb{E}[\boldsymbol{\Sigma}|\boldsymbol{X}]$ lies within the family of multi-target linear shrinkage estimators: it is a convex combination between the MLE and the target matrices $\boldsymbol{D}_1,\ldots,\boldsymbol{D}_L$. We refer to the estimator in Equation (2.10) as the Target-Averaged linear Shrinkage (TAS) estimator, hereafter denoted by $\hat{\boldsymbol{\Sigma}}_{\mathrm{TAS}}$.

The proposed estimator has several desirable properties. First, it provides a generic framework where any positive definite target matrix can be incorporated in the target set $\mathcal{D}$. Second, it is computationally attractive since the computation of Equation (2.10) only requires $K \times L$ evaluations of the marginal likelihood of a Gaussian conjugate model, which is available in closed-form via Equation (2.7) (with derivation in Appendix A.2). Also, when an additional target matrix $\boldsymbol{D}_{L+1}$ is added to the set $\mathcal{D}$, updating (2.10) only requires K new marginal likelihood evaluations and subsequently re-distributing the weights. Third, the target-specific weights w_l may provide valuable insights (see Sections 2.6, 2.7, 2.8).

2.5 Choice of shrinkage target matrices

The performance of the TAS estimator depends on the choice of the set of target matrices $\mathcal{D}$, much alike the performance of STS estimators depends on the choice of the target matrix $\mathbf{\Lambda}$. Here, we discuss the choice of $\mathcal{D}$.

In the absence of prior information, the set $\mathcal{D}$ may include, for example, the nine target matrices described in Table 2.1. Such choice may be seen as a sensible starting point due to the popularity of these nine targets in the literature. Note, however, that some of the targets can be nearly identical in some cases (e.g. $\boldsymbol{T}_2$ and $\boldsymbol{T}_5$ when $\bar{r} \approx 0$), so the posterior weights in (2.11) must be interpreted with care. It is also possible to further enrich this set with any covariance structures not listed in Table 2.1. Examples include Toeplitz, higher-order autoregressive, or latent factor structures [e.g. 23, 69].

The set $\mathcal{D}$ may also be used to incorporate external information about $\mathbf{\Sigma}$, provided this can be translated into a positive definite covariance matrix. The availability of such information may arise in situations where the same set of molecular variables has been measured on an independent sample that is thought to be biologically related (e.g. similar disease). In this case, a target matrix may be constructed using the sample covariance matrix of the auxiliary data, or regularised versions thereof. This is illustrated in Section 2.8 using data from The Cancer Proteome Atlas.

	zero correlation ($r_{ij} = 0$)	constant correlation ($r_{ij} = \bar{r}$)	decaying correlations ($r_{ij} = \bar{r}^{\lvert i-j \rvert}$)
unit variance ($v_i = 1$)	$\boldsymbol{T}_1$	$\boldsymbol{T}_4$	$\boldsymbol{T}_7$
common variance ($v_i = \bar{s}$)	$\boldsymbol{T}_2$	$\boldsymbol{T}_5$	$\boldsymbol{T}_8$
unequal variances ($v_i = s_{ii}$)	$\boldsymbol{T}_3$	$\boldsymbol{T}_6$	$\boldsymbol{T}_9$

Table 2.1 Popular choices of shrinkage target matrices for STS estimators. A shrinkage target $\boldsymbol{T} = \boldsymbol{V}^{1/2}\boldsymbol{R}\boldsymbol{V}^{1/2}$, with $\boldsymbol{V} = \text{diag}\{v_1, \ldots, v_p\}$ a diagonal variance matrix and $\boldsymbol{R} = (r_{ij})_{1 \leq i < j \leq p}$ a correlation matrix. Here, s_{ij} denotes the $(i,j)^{\text{th}}$ element of the sample covariance matrix $\boldsymbol{S}$; $\bar{s}$ and $\bar{r}$ are the averages of the empirical variances and correlations, respectively.

2.6 Model-based simulation study

In this section, we study the performance of the proposed estimator using simulated data. We generate $M = 100$ data sets of size $n \in \{25, 50, 75\}$ from a p-variate Gaussian distribution with zero mean vector and covariance matrix Σ, where $p = 100$. Four

distinct covariance structures are considered, yielding the following four simulation scenarios:

- **Scenario 1: common variance, zero correlation.** $\boldsymbol{\Sigma}_1 = 5 \times \boldsymbol{I}_{p\times p}$,
- **Scenario 2: unit variance, constant correlation.** $\boldsymbol{\Sigma}_2 = \boldsymbol{I}_{p\times p} + 0.3 \times (\mathbf{1}_{p\times p} - \boldsymbol{I}_{p\times p})$, where $\mathbf{1}_{q\times r}$ is the $q \times r$ unit matrix with elements all equal to one.
- **Scenario 3: unequal variances, decaying correlations.** $\boldsymbol{\Sigma}_3 = \boldsymbol{D}^{1/2}\boldsymbol{C}\boldsymbol{D}^{1/2}$, where the $(i,j)^{\text{th}}$ entry of $\boldsymbol{C}$ equals $(-0.7)^{|i-j|}$ and $\boldsymbol{D} = \text{diag}(d_1,\ldots,d_p)$ with $d_i \sim \mathcal{U}(1,5)$.
- **Scenario 4: unit variance, block-diagonal correlation.** $\boldsymbol{\Sigma}_4 \sim$ Inv-Wishart, such that $\mathbb{E}[\boldsymbol{\Sigma}_4] \propto \boldsymbol{B}$, where $\boldsymbol{B}$ is a block-diagonal matrix with two identical $p/2 \times p/2$ blocks, each with the same constant correlation structure that was used in scenario 2.

These scenarios have been chosen to capture distinct covariance structures that are represented in the default target set $\mathcal{D} = \{\boldsymbol{T}_1,\ldots,\boldsymbol{T}_9\}$ (i.e. $\boldsymbol{T}_2$, $\boldsymbol{T}_4$ and $\boldsymbol{T}_9$ for scenarios 1, 2 and 3 respectively), as well as to include a case (scenario 4) that is not captured by the target set $\mathcal{D}$. Using data simulated under these scenarios, we compare the performance of the multi-target shrinkage estimator $\hat{\boldsymbol{\Sigma}}_{\text{TAS}}$, with target set $\mathcal{D}$ and the nine STS estimators obtained when using each of the shrinkage targets in $\mathcal{D}$ separately, e.g. Equation (2.10) using just a single target. These are denoted by $\hat{\boldsymbol{\Sigma}}_{\text{ST1}}$, ..., $\hat{\boldsymbol{\Sigma}}_{\text{ST9}}$. We also consider the estimators of Schäfer et al. [95] and Touloumis [103], respectively implemented in the `R` packages `corpcor` and `ShrinkCovMat`. The estimator of Schäfer et al. [95] is an STS estimator obtained via a two-step approach in which the sample variances are shrunk towards their median and the sample correlations shrunk towards zero. We denote this estimator by $\hat{\boldsymbol{\Sigma}}_{\text{cpc}}$. The estimators proposed by Touloumis [103] are three non-parametric STS estimators (i.e. they do not rely on distributional assumptions) with shrinkage targets $\boldsymbol{T}_1$, $\boldsymbol{T}_2$, and $\boldsymbol{T}_3$. We denote these by $\hat{\boldsymbol{\Sigma}}_{\text{AT1}}$, $\hat{\boldsymbol{\Sigma}}_{\text{AT2}}$, and $\hat{\boldsymbol{\Sigma}}_{\text{AT3}}$, respectively. Additional linear shrinkage estimators that were considered for the comparison include those of Chen et al. [24], Fisher and Sun [44], and the single target estimator from Ikeda et al. [59]. Within a multivariate normal framework, Chen et al. [24] employed the Rao-Blackwell theorem to improve upon the estimator in Ledoit and Wolf [70]. Instead of using less precise data-derived estimates of the optimal shrinkage intensity, Fisher and Sun [44] improves estimation under the multivariate normal framework by exploiting the assumed Gaussian properties in order to directly compute it. Ikeda et al. [59] constructs non-parametric estimators of the optimal shrinkage intensity that differ only by a small term from those of Touloumis [103]. These three methods have been shown to perform worse than or equal to those in Touloumis [103] and so they are not included in the comparison.

To assess the performance of these 14 estimators, we report the Percentage Relative Improvement in Average Loss (PRIAL) [103, 59]:

$$\frac{\sum_{m=1}^{M} \|\boldsymbol{\Sigma} - \boldsymbol{S}^{(m)}\|_F^2 - \sum_{m=1}^{M} \|\boldsymbol{\Sigma} - \hat{\boldsymbol{\Sigma}}^{(m)}\|_F^2}{\sum_{m=1}^{M} \|\boldsymbol{\Sigma} - \boldsymbol{S}^{(m)}\|_F^2} * 100, \tag{2.12}$$

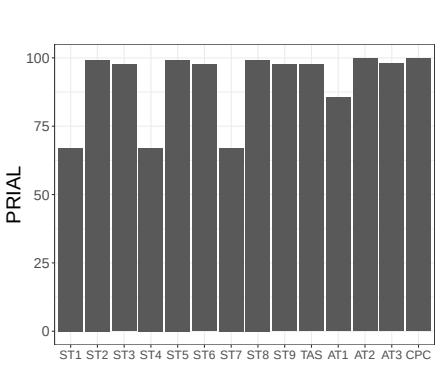
where $\|\cdot\|_F$ denotes the Frobenius norm. The PRIAL measures the relative improvement of an estimator $\hat{\boldsymbol{\Sigma}}$ over the sample covariance matrix $\boldsymbol{S}$, across the M simulated data sets. A negative value indicates that the estimator $\hat{\boldsymbol{\Sigma}}$ does not improve upon $\boldsymbol{S}$, whereas a positive value indicates an improvement. The improvement is relatively small when the PRIAL value is close to 0% (in which case $\hat{\boldsymbol{\Sigma}}$ is relatively closer to $\boldsymbol{S}$) and relatively large when the PRIAL value is close to 100% (in which case $\hat{\boldsymbol{\Sigma}}$ is relatively closer to $\boldsymbol{\Sigma}$). The PRIAL can also be interpreted as the improvement of performing shrinkage versus no shrinkage. When considering changes in PRIAL as n gets closer to p, it is important to keep in mind that reductions in PRIAL are not necessarily an indicator of diminishing performance of the shrinkage estimators; it is just that the sample estimator could be benefitting more from the increase in sample size than the shrinkage estimators are, which is entirely reasonable e.g. when the shrinkage targets do not capture the underlying covariance structure. This is an artefact of the PRIAL metric.

Figures 2.1 and 2.2 summarise the results obtained for $n = 25$ (results for $n \in \{50, 75\}$, which are similar to that of $n = 25$, are provided in Appendix A.4). Overall, we observe that the performance of STS estimators clearly varies across the different simulation scenarios, and that it may strongly depend on the choice of shrinkage target. Large PRIAL values are observed for STS estimators when the shrinkage target resembles the true covariance matrix (e.g. $\boldsymbol{T}_4$ in scenario 2), whereas negative PRIAL values (indicating that the estimator performs worse than the sample covariance matrix) are observed in cases where the shrinkage target is *misspecified* (see scenario 2). In contrast, the TAS estimator achieves a similar performance with respect to the best STS estimator without having to choose the correct shrinkage target, and this even when the target set does not contain the true underlying covariance structure (see scenario 4). This highlights a key strength of the proposed multi-target estimator, namely that it is less sensitive to misspecification of its targets.

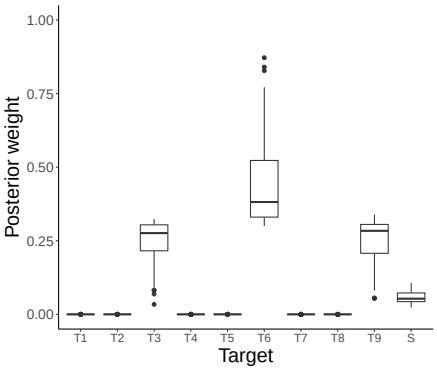
As illustrated in the right panels of Figures 2.1 and 2.2, target-specific posterior weights (see Equation (2.11)) can also provide insights about the structure of the true covariance matrix $\boldsymbol{\Sigma}$. For example, in scenario 3, TAS allocates the highest posterior weight to shrinkage targets that match the underlying covariance structure of the data (i.e. $\boldsymbol{T}_9$). A similar behaviour is observed in scenario 1 and 2, although this is less clear. Indeed, the shrinkage target $\boldsymbol{T}_6$ is assigned the largest weight in scenario 2, while it would be expected that $\boldsymbol{T}_4$ has the highest weight. Similarly, the shrinkage targets $\boldsymbol{T}_3$,

$\boldsymbol{T}_6$ and $\boldsymbol{T}_9$ have high posterior weights in scenario 1 whereas it would be expected that $\boldsymbol{T}_2$ has the highest weight. However, closer inspection of the shrinkage targets (see Appendix Figure A.7) shows that $\boldsymbol{T}_6$ is almost equal to $\boldsymbol{T}_4$ in scenario 2, and that $\boldsymbol{T}_3$, $\boldsymbol{T}_6$ and $\boldsymbol{T}_9$ are almost equal to $\boldsymbol{T}_2$ in scenario 1. It is also observed that the distances (as measured by the Frobenius norm) between each of these targets to the true covariance matrix are almost equal (see Appendix Figure A.7). Additionally, in scenario 4, the highest posterior weight is assigned to the shrinkage target $\boldsymbol{T}_6$ that is the closest to the true covariance matrix, along with targets $\boldsymbol{T}_4$ and $\boldsymbol{T}_5$. Overall, these simulations suggest that shrinkage weights are capable to exclude (i.e. the posterior weight is equal to zero) shrinkage targets whose shape is quite distinct to the true underlying covariance structure. These results also show that having very similar shrinkage targets in the target set $\mathcal{D}$ does not harm the performance of the TAS estimator, but that it may complicate the interpretation of the (posterior) shrinkage weights. Thus we would recommend that Frobenius distance between targets are systematically evaluated and considered together with the shrinkage weights.

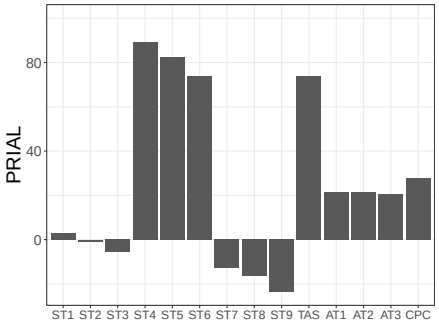
The non-parametric estimators $\hat{\boldsymbol{\Sigma}}_{\text{AT1}}, \hat{\boldsymbol{\Sigma}}_{\text{AT2}}$ and $\hat{\boldsymbol{\Sigma}}_{\text{AT3}}$ perform in general better than their parametric counterparts $\hat{\boldsymbol{\Sigma}}_{\text{ST1}}, \hat{\boldsymbol{\Sigma}}_{\text{ST2}}$ and $\hat{\boldsymbol{\Sigma}}_{\text{ST3}}$. This suggests that, when using the same shrinkage target, improved performance can be obtained by relaxing distributional assumptions. However, alike the behaviour observed for $\hat{\boldsymbol{\Sigma}}_{\text{T1}}, \ldots, \hat{\boldsymbol{\Sigma}}_{\text{T9}}$, the performance of $\hat{\boldsymbol{\Sigma}}_{\text{AT1}}, \ldots, \hat{\boldsymbol{\Sigma}}_{\text{AT3}}$ can also be affected by the choice of shrinkage target (see scenarios 1 and 3). Finally, on average, we observe that the proposed multi-target TAS estimator performs similarly to $\hat{\boldsymbol{\Sigma}}_{\text{cpc}}$ (scenarios 1 and 3) or better (scenario 2 and 4, where the true covariance matrix has a more dense structure). However, this is not true for scenario 4 $n = 75$ (Figure A.6).



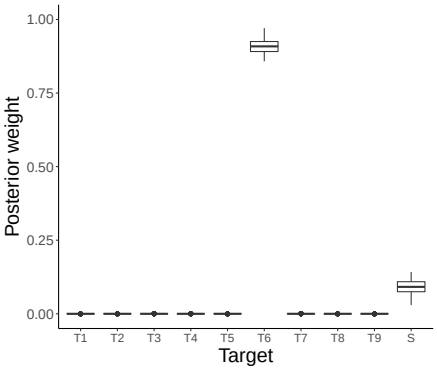
(a) Scenario 1: PRIAL



(b) Scenario 1: target-specific posterior weights



(c) Scenario 2: PRIAL



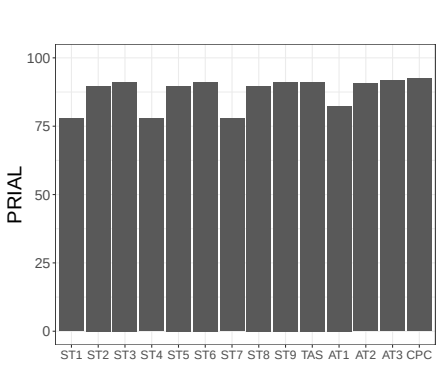
(d) Scenario 2: target-specific posterior weights

Fig. 2.1 Simulation results for scenarios 1 and 2 when $n = 25$. Barplots display the PRIAL for each estimator and boxplots display target-specific posterior weights (see Equation (2.11)) of the TAS estimator. ST1, ..., ST9 refer to the nine STS estimators, TAS to estimator (2.10), AT1, ..., AT3 to the three estimators of Touloumis [103] and CPC to the estimator of Schäfer et al. [95].

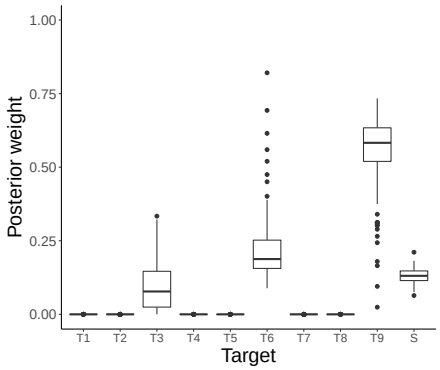
2.7 Predictive validation simulation

Here, we employ gene expression data from The Cancer Genome Atlas (TCGA) and a data partitioning strategy to assess the performance of the estimator in Equation (2.10) and evaluate the benefits of incorporating external information into the target set $\mathcal{D}$. We retrieved, using the R package *cgdsr* [61], all TCGA level 3 normalised gene expression data that were measured using the Agilent 244K Custom gene Expression G4502A_07 array. The data span 10 cancer types. However, we consider the following two low-dimensional extracts:

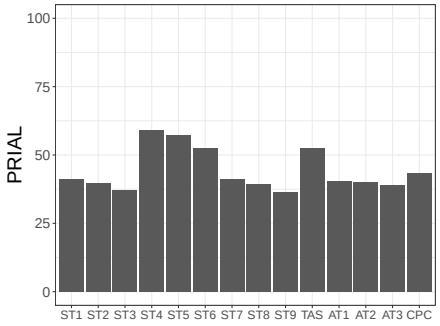
- **Data set 1:** p53 pathway in breast cancer ($p = 68$ genes in $N = 529$ samples)



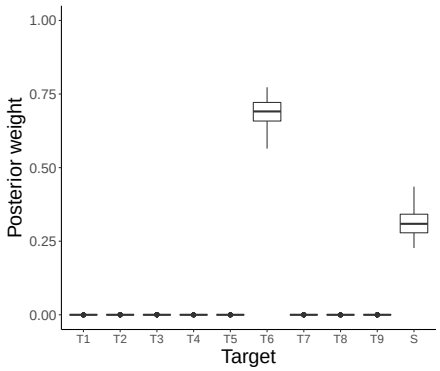
(a) Scenario 3: PRIAL



(b) Scenario 3: target-specific posterior weights



(c) Scenario 4: PRIAL



(d) Scenario 4: target-specific posterior weights

Fig. 2.2 Simulation results for scenarios 3 and 4 when $n = 25$. Barplots display the PRIAL for each estimator and boxplots display target-specific posterior weights (see Equation (2.11)) of the TAS estimator. ST1, ..., ST9 refer to the nine STS estimators, TAS to estimator (2.10), AT1, ..., AT3 to the three estimators of Touloumis [103] and CPC to the estimator of Schäfer et al. [95].

- **Data set 2:** apoptosis pathway in ovarian cancer ($p = 86$ genes in $N = 558$ samples)

As the true covariance structures between genes in these two data sets are unknown, we use a data partitioning strategy [105, 68] to assess the performance of estimators $\hat{\mathbf{\Sigma}}_{\text{TAS}}$, $\hat{\mathbf{\Sigma}}_{\text{AT1}}$, $\hat{\mathbf{\Sigma}}_{\text{AT2}}$, $\hat{\mathbf{\Sigma}}_{\text{AT3}}$ and $\hat{\mathbf{\Sigma}}_{\text{cpc}}$ (in light of the results shown in Section 2.6, $\hat{\mathbf{\Sigma}}_{\text{ST1}},\ldots,\hat{\mathbf{\Sigma}}_{\text{ST9}}$ are excluded from this comparison). The strategy is illustrated in Figure 2.3. For a given data set, the strategy consists of randomly splitting the full data matrix ($p \times N$) into a small sample size ($p \times n$) and a large sample size ($p \times (N-n)$) data matrix, for $n \in \{p/4, p/2, 3p/4\}$. Given this partition, all estimators are computed using the small sample size data matrix, whereas the sample covariance matrix obtained from the large sample size data matrix is used as a proxy for the true covariance when calculating the PRIAL (see Equation (2.12)). This procedure is repeated 1,000 times for data sets 1 and 2, and for the three different values of n investigated.

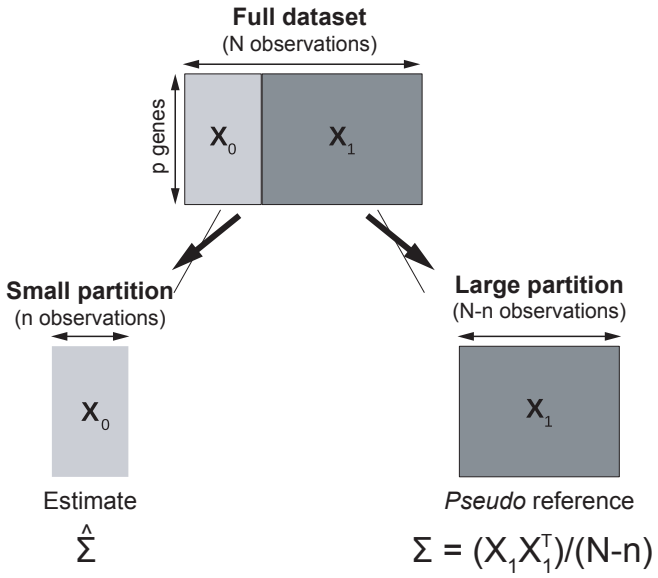
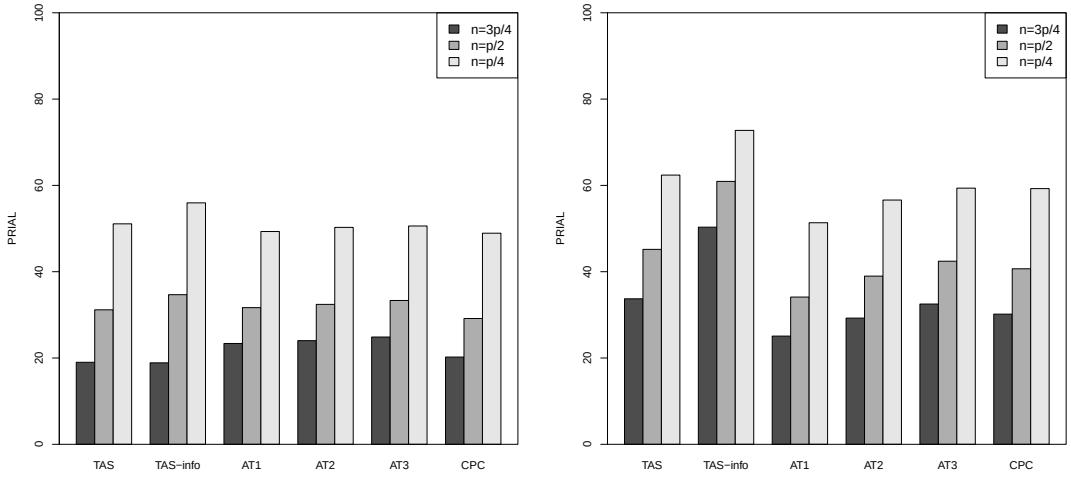


Fig. 2.3 Illustration of the data-partition strategy.

To illustrate the benefits of incorporating external information into the target set, we also consider the multi-target shrinkage estimator $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$ with target set $\mathcal{D}_{\text{info}} = \mathcal{D} \cup \hat{\mathbf{\Sigma}}_{\text{ext}}$, where $\hat{\mathbf{\Sigma}}_{\text{ext}}$ is an estimate of the covariance between genes that is obtained from independent data. For data sets 1 and 2, we obtain such estimates by pooling the TCGA gene expression data from the nine other cancer types for which expression levels were measured using the Agilent platform. To ensure that $\hat{\mathbf{\Sigma}}_{\text{ext}}$ is positive definite and well-conditioned, we use a regularised estimate (obtained using Equation (2.10)) instead of the pooled sample covariance.

Figure 2.4 summarises the results for the experiment described above. Overall, for data set 1, all estimators achieve a similar PRIAL regardless of n/p ratios (Figure 2.4(a)). For data set 2, however, we observe that $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$ (and to a lesser extent $\hat{\mathbf{\Sigma}}_{\text{TAS}}$) outperforms all other estimators. This highlights another key strength of the TAS estimator: its ability to incorporate external information within the target set can substantially improve performance.

Figure 2.5 shows the distribution of target-specific posterior weights (see Equation (2.11)) in estimators $\hat{\mathbf{\Sigma}}_{\text{TAS}}$ and $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$ across the 1,000 random data partitions performed for data set 2 . We observe in Figure 2.5(a) that the shrinkage target $\boldsymbol{T}_6$ (constant correlation and unequal variances) is assigned the largest weight in estimator $\hat{\mathbf{\Sigma}}_{\text{TAS}}$, among all targets. This may be due to the fact that genes within the apoptosis pathway are expected to have high correlations between each other. Therefore, the shrinkage estimation of the covariance matrix may benefit from a shrinkage target



(a) P53 pathway - breast cancer (b) Apoptosis pathway - ovarian cancer

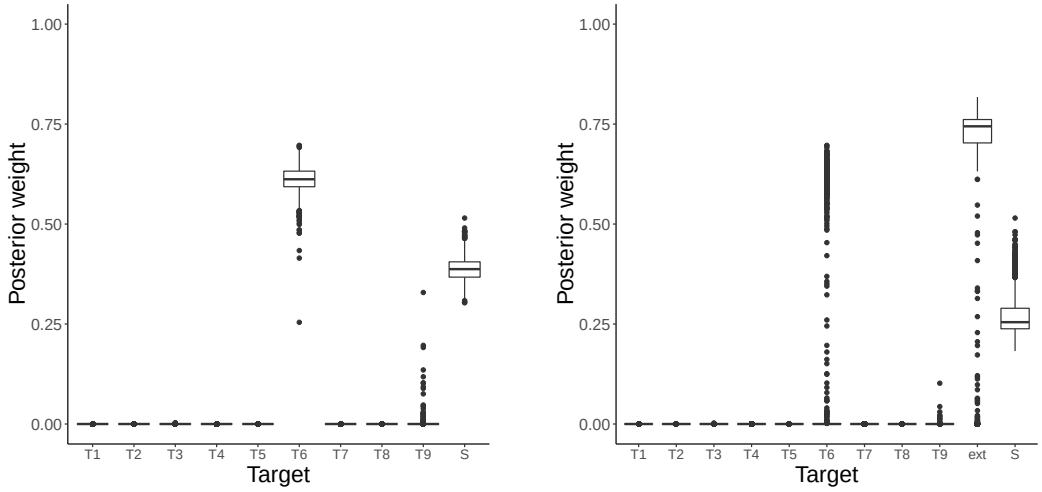
Fig. 2.4 Results of the TCGA gene expression predictive validation simulation. Barplots display the PRIAL calculated for each estimator for (a) data set 1 (p53 pathway, breast cancer samples) and, (b) data set 2 (apoptosis pathway, ovarian cancer samples).

whose off-diagonal elements are not equal to zero. On the other hand, we observe in Figure 2.5(b) that the shrinkage target $\hat{\mathbf{\Sigma}}_{\text{ext}}$, derived from external data, is assigned the largest weight in estimator $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$. This is in line with the results shown in Figure 2.4, where the incorporation of external information substantially improved performance in data set 2. Appendix Figures A.8 and A.9 show that $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$ also puts more weight on the shrinkage target $\hat{\mathbf{\Sigma}}_{\text{ext}}$ for data set 1, which results only in a small improvement in PRIAL (see Figure 2.4(a)). Overall, complementary results in Appendix A.5 for other n/p ratios show, as expected, that when n increases, both $\hat{\mathbf{\Sigma}}_{\text{TAS}}$ and $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$ put more weight on the sample covariance matrix in both data sets. Also, in Appendix Figure A.11 are heatmaps to show the similarity between targets.

Finally, while the multivariate normal assumption does not seem to be supported by these two gene expression data sets (see Appendix A.6), it is found that the non-parametric estimators of Touloumis [103] do not generally outperform the TAS estimator, which assumes multivariate normality. In fact, the opposite can occur for specific choices of target matrices (e.g. when external information is included). This may suggest that accounting for multiple shrinkage target matrices may be more critical than flexible distributional assumptions.

2.8 Application to protein expression data

In this section, we apply our method to protein expression data from The Cancer Proteome Atlas (tcpaportal.org/tcpa). In particular, we consider the PANCAN32 data set, focusing on level 4 normalised expression levels of 209 proteins that were measured

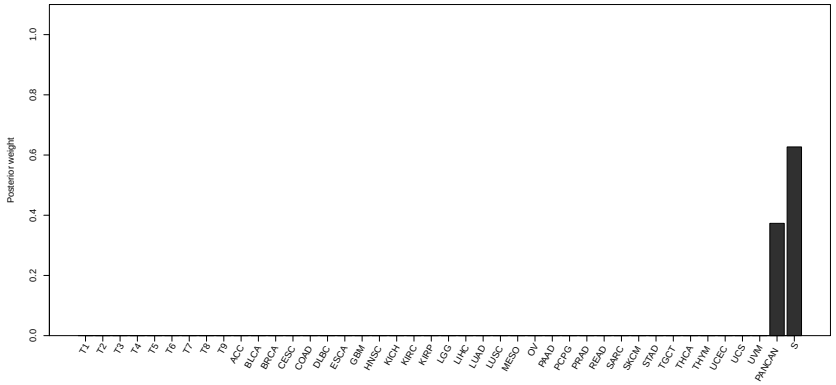


(a) $\hat{\boldsymbol{\Sigma}}_{\text{TAS}}$ (b) $\hat{\boldsymbol{\Sigma}}_{\text{TAS-info}}$

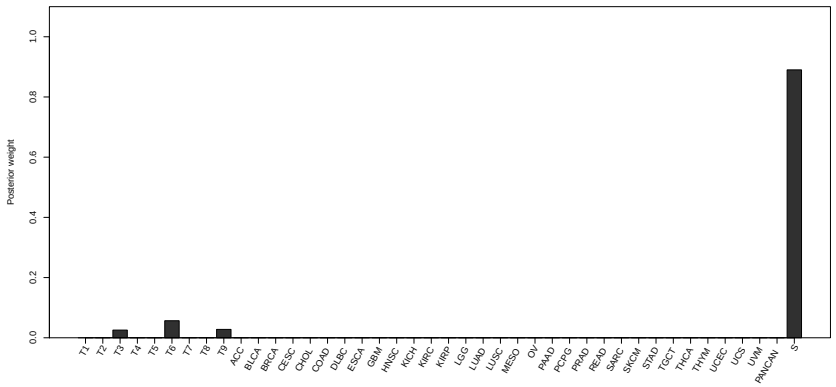
Fig. 2.5 Target-specific posterior weights (see Equation (2.11)) obtained for estimators $\hat{\boldsymbol{\Sigma}}_{\text{TAS}}$ and $\hat{\boldsymbol{\Sigma}}_{\text{TAS-info}}$ across the 1,000 random data partitions of the ovarian cancer data set when $n = p/2$. The target "ext" in $\hat{\boldsymbol{\Sigma}}_{\text{TAS-info}}$ stands for the shrinkage target $\hat{\boldsymbol{\Sigma}}_{\text{ext}}$ estimated from external data.

on 7,694 samples across 32 cancer types. Appendix Table A.1 provides for each cancer type its acronym and the number of samples.

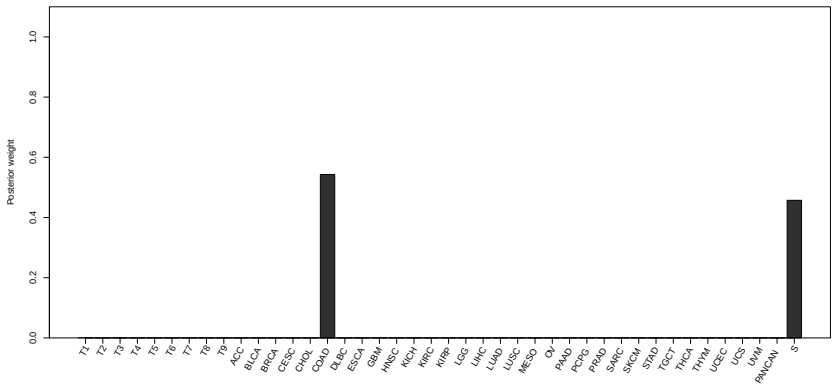
We first use the TAS estimator to estimate the covariance between the 209 proteins separately for three histologically different cancers, namely cholangiocarcinoma (CHOL), liver hepatocellular carcinoma (LIHC) and rectum adenocarcinoma (READ). For each of these three data sets, the target set of the TAS estimator includes the nine targets of Table 2.1 (denoted $\boldsymbol{T}_1, \ldots, \boldsymbol{T}_9$), 31 targets derived from each of the other cancer types (which we will refer to by their acronyms in Appendix Table A.1) and one target obtained by pooling the data from the 31 cancer types (referred to as PANCAN). To ensure that shrinkage targets derived from independent data sets are positive definite and well-conditioned, we use the TAS estimate using the nine targets of Table 2.1 instead of the sample covariance matrix (however any other regularisation technique may be used instead).



(a) Cholangiocarcinoma (CHOL)



(b) Liver hepatocellular carcinoma (LIHC)



(c) Rectum adenocarcinoma (READ)

Fig. 2.6 Target-specific posterior weights in estimators $\hat{\mathbf{\Sigma}}_{\text{TAS}}$ when analysing a) the cholangiocarcinoma, b) liver hepatocellular carcinoma and, c) rectum adenocarcinoma proteomic data sets.

Figure 2.6 reports target-specific posterior weights (see Equation (2.11)) of the TAS estimator obtained for each of these three data sets. This shows that the TAS estimator assigns large weights to different types of shrinkage targets across these datasets. For example, the PANCAN shrinkage target (that pools data from the 31 remaining cancers) is assigned a large weight in the Cholangiocarcinoma (CHOL) data set but not in the

other two data sets. Virtually no weight is attributed to any of the targets derived from external data in the Liver hepatocellular carcinoma (LIHC) data set, whereas in the Rectum adenocarcinoma (READ) data set a large weight is assigned to the shrinkage target derived from the colon adenocarcinoma (COAD) cancer data. For the latter, it is biologically plausible that the dependence structure between proteins in rectum and colon adenocarcinoma samples are similar because both tumours are histologically related. Overall, these observations support the conclusions that covariance estimation may or may not benefit from the incorporation of external information and that, when it does, estimation can benefit both from generic (e.g. the PANCAN shrinkage target) and specific (e.g. the COAD shrinkage target) prior information.

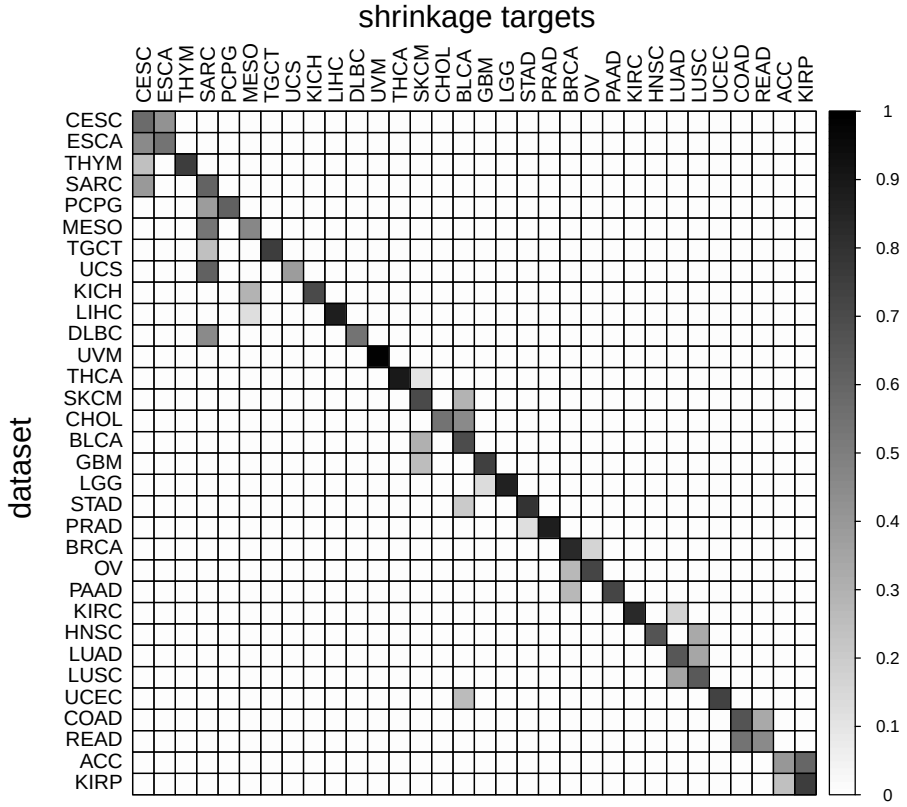


Fig. 2.7 Posterior shrinkage weights obtained by the TAS estimator. Columns represent the shrinkage targets comprised in the target set of the TAS estimator. Elements on the diagonal represent shrinkage weights associated with the sample covariance of the data set. The per-row sum is equal to one.

We now illustrate that the TAS estimator can provide insights regarding the relationship between the 32 cancer types shown in Table 2.1. For each of the 32 cancer data sets, we consider the TAS estimator with target set comprising of 31 shrinkage targets derived from the other 31 cancer types. We use the same strategy as above to make sure the shrinkage targets are positive definite and well-conditioned. Figure 2.7 displays the

posterior shrinkage weights obtained by the TAS estimator for each of the 32 cancer data sets. Our results suggest that high posterior weights might indicate similarity between cancers in terms of covariance structures. In particular, the target-specific posterior weights suggest a relatively high similarity between cancers with known putative biological similarity: (a) lung adenocarcinoma (LUAD) and lung squamous cell carcinoma (LUSC), both subtypes of non-small cell lung cancer [40]; (b) COAD and READ, both colorectal cancers [79] and (c) breast invasive carcinoma (BRCA) and ovarian serous cystadenocarcinoma (OV), with known common susceptibility genes [63]. These pairs of cancers have been also shown to be similar by pancancer analyses of previous releases of the TCPA dataset [e.g. 97]. Additionally, our results suggest a high similarity between esophageal carcinoma (ESCA) and cervical squamous cell carcinoma and endocervical adenocarcinoma (CESC). Both of these cancers have been found to be linked to human papillomavirus [107, 72].

Figure 2.7 also suggests that covariance estimation for cancers with small sample size can benefit from shrinkage towards cancer types with a large number of samples. Examples include adrenocortical carcinoma (ACC; $n = 46$) with kidney renal papillary cell carcinoma (KIRP; $n = 208$), uterine carcinosarcoma (UCS; $n = 48$) with sarcoma (SARC; $n = 221$), as well as cholangiocarcinoma (CHOL; $n = 30$) with bladder urothelial carcinoma (BLCA; $n = 344$). Despite this, no posterior weight was allocated to other cancer types in the case of Uveal melanoma (UVM; $n = 12$). This could be a consequence of its very small sample size, or it may suggest that protein interactions in UVM are unrelated to that of the other cancers. Future releases of TCPA, in which more samples are available, could enable us to confirm this.

2.9 Software

The TAS method is available as an `R` package `TAS` at http://github.com/HGray384/TAS. The main function in `TAS` is of course the implementation of the TAS estimator in Equation (2.10), whose corresponding function is `taShrink`. The function allows the user full flexibility over the model parameters. The input parameters `targets` and `alpha` allow the user to specify the sets $\mathcal{D}$ and $\mathcal{A}$. As default, $\mathcal{D}$ comprises the nine shrinkage targets defined in Table 2.1 and the argument `without` allows the user to conveniently exclude some of the targets from this default set without having to manually input them.

The default support $\mathcal{A}$ is set as $\{a_1 = 0.01, a_2 = 0.02, \ldots, a_{99} = 0.99\}$ (note that increasing the granularity of this grid does not affect results; see Appendix A.3). However, these choices can easily be modified when using the software. We remark that the $K \times L$ marginal likelihood evaluations that are required to compute Equation (2.10)

can easily be parallelised to further reduce computational time. We observe, however, that this is not critical in practice (see Table 2.2).

	$p = 100$	$p = 500$	$p = 1000$
$n = 100$	0.08	4.46	33.61
$n = 250$	0.09	4.68	33.21
$n = 500$	0.10	5.14	34.14

Table 2.2 Average time in seconds (over 100 repetitions) to compute the TAS estimate (using the nine targets in Table 2.1) as a function of the number n of samples and p of variables. Timings were measured on a Dell OptiPlex7040 with Intel Core i7-6700CPU.

The option `plots` allows the user to choose whether to display the posterior weights of Equation (2.11) as a bar chart as part of the function call to `taShrink`, generating graphics similar to Figures 2.1, 2.2, and 2.6. To demonstrate the exact output, we simulate $n = 5$ data vectors from $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_{10\times 10})$ so that $p = 10$ and $\boldsymbol{\Sigma} = \boldsymbol{I}_{10\times 10}$ and apply `taShrink` with `plots=TRUE`, shown in Figure 2.8.

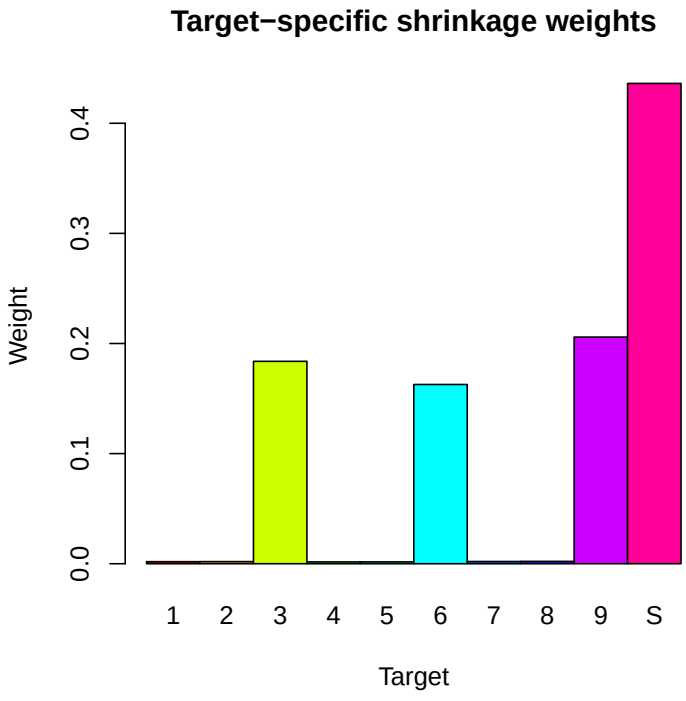


Fig. 2.8 Posterior shrinkage weights obtained by the TAS estimator for $n = 5$ data vectors with $p = 10$ generated from $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_{10\times 10})$ so that $\boldsymbol{\Sigma} = \boldsymbol{I}_{10\times 10}$.

The option `shrink.var` in `taShrink` is a feature currently under development that will allow the user to only shrink correlations, i.e. disabling the shrinkage of variances.

This is useful for shrinking correlations and variances separately, and therefore allowing a different shrinkage parameter for each, as in Schäfer et al. [95]. Another feature that is under development for `taShrink` is the option `grid.corr`. This input will allow the user to specify a grid of correlation values around the estimated correlation parameter $\bar{r}$ used in the target matrix (Table 2.1). This can be seen as giving the user the range of uncertainty to be considered around the sample correlation estimate.

`taShrink` uses the function `getTargetSet` to construct the default target set and `logML` to compute the log marginal-likelihood of the data given each target. Both of these functions are implemented using a back-end interface to `C++` through the `Rcpp` [34] and `RcppArmadillo` [35] `R` packages. The `Rcpp` package presents a useful tool for statistical software that is to be used in an applied context since it retains the friendly surface elements for users of `R` whilst allowing developers to implement their code in ways that are typically restrictive when using only base `R`, e.g. performing heavy computations more quickly. The `RcppArmadillo` package takes this further, by allowing the developer to access the `C++` linear algebra library `Armadillo` [93] for efficient implementations of many data structures and manipulations using syntax that is not as challenging as that of `C++`. Although there are no heavy computations used within `getTargetSet`, populating the entries of the target matrices can take some time for large p using base `R`. Finally, the weights from Equation (2.11) are recovered from the output of `logML` via the *log-sum-exp* trick [77] to avoid numerical underflow.

When only one target is provided to `taShrink`, then single target shrinkage is performed via the function `gcShrink` with the option `weighted=TRUE`. This results in single-target shrinkage performed as in Equation (2.10) with one target, i.e. averaging the uncertainty over the shrinkage weight only. The corresponding `C++` function for single target construction `getTarget` is then called and `logML` is evaluated for that target and grid of α. When `gcShrink` is called by itself with the option `weighted=FALSE`, then only the value of α that maximises the log marginal-likelihood is used as the shrinkage intensity (i.e. all of the posterior weight is assigned to this value) and the empirical Bayes estimator of Hannart and Naveau [52] is therefore employed. The `plots` option allows a graphical display of the log marginal-likelihood evaluated at each value of α, including clear visualisation of the value with the highest marginal likelihood. Figure 2.9 shows the output of `gcShrink` with `plots=TRUE` for $n = 5$ data vectors from $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_{10\times 10})$ so that $p = 10$ and $\boldsymbol{\Sigma} = \boldsymbol{I}_{10\times 10}$ using the truth $\boldsymbol{I}_{10\times 10}$ as the target.

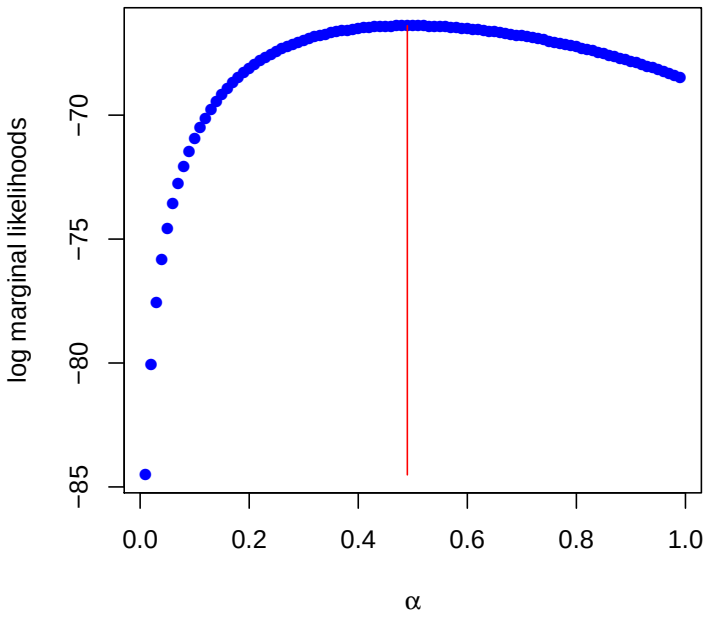


Fig. 2.9 Log marginal-likelihood values obtained by the TAS estimator for $n = 5$ data vectors with $p = 10$ generated from $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_{10\times10})$ so that $\boldsymbol{\Sigma} = \boldsymbol{I}_{10\times10}$ and $\boldsymbol{I}_{10\times10}$ is used as the only target.

The output from `taShrink` can be used by the auxiliary functions `addTarget`, `targetSimilarities`, and `targetWeights` to rerun TAS with an extra target, compare the similarity of the targets in the target set, and to inspect the posterior weights assigned to each target and the sample covariance matrix, respectively. Notably, `addTarget` is much more efficient than rerunning TAS with the whole target set again since only the log marginal-likelihood values of the new target are needed to re-weight with those of the previous targets in order to re-estimate TAS, as highlighted in Section 2.4. The output of `targetSimilarities` is shown in Appendix Figure A.7 for the model-based simulations.

2.10 Discussion

We proposed a flexible, yet computationally simple, Bayesian covariance estimator that can accommodate an arbitrary number of shrinkage target matrices. The estimator is particularly useful in high-dimensional settings ($n << p$), where shrinkage is most important, and when external information is available. For these reasons, the present work is particularly relevant in the context of high-throughput genomic experiments due to (i) the central role that covariance estimation plays in multivariate data analyses,

(ii) the high-dimensionality of the data and, (iii) the increasing availability of large open data repositories (e.g. TCGA) which can provide relevant external information for specific studies.

To the best of our knowledge, only Bartz et al. [6], Lancewicki and Aladjem [64] and Ikeda et al. [59] have proposed multi-target linear shrinkage estimators for covariance estimation. Numerical comparison with these methods has not yet been performed and, in part because no implementations of them are provided.

Both Bartz et al. [6] and Lancewicki and Aladjem [64] independently developed the same multi-target linear shrinkage estimator for covariance matrices. Bartz et al. [6] focusses on the general method of multi-target linear shrinkage (e.g. not limited to only covariance matrix estimation) and also provides the theory to establish consistency of the estimator in high-dimensional situations. Lancewicki and Aladjem [64] instead focusses solely on covariance matrix estimation using multi-target linear shrinkage, in particular deriving an optimisation function to estimate the optimal shrinkage intensities for a general form of target matrix that encompasses several of those in Table 2.1. The function has the attractive property that it is strictly convex and so can be solved using appropriate algorithms.

In practice, it is unclear how much computational time is needed to solve this optimisation function (though it is strictly convex) and if it is feasible for large problems. Lancewicki and Aladjem [64] present simulation results for $p = 50$ and $n \leq 30$, and the maximum that Bartz et al. [6] present is $p = 500$, $n = 500$ – neither present computing time for the user. Additionally, for a large number of targets or targets that are very similar, the optimisation problem is ill-posed [64].

These frequentist methods are conceptually different to the TAS estimator. They focus on estimating the weights that produce an optimal linear combination of targets (in the mean-square sense). This differs from the TAS approach in which multiple shrinkage targets are weighted according to their individual model evidence. TAS is focussed upon incorporating uncertainty rather than achieving optimality. On inspection, it would also appear that the computational simplicity of TAS means that it will be more scalable than these methods to large problems and many targets, though this will require more thorough analysis. Additionally, TAS is not limited by similarity of targets in the target set, although interpretation of the assigned weights does become less clear in this situation.

Independent of this work, Ikeda et al. [59] have proposed a two-target shrinkage estimator also motivated from a conjugate Bayesian perspective. Ikeda et al. [59] opt to take a different approach to Hannart and Naveau [52] when estimating the shrinkage weights. Instead of choosing the shrinkage intensity that maximises the marginal likelihood as in Hannart and Naveau [52], they decide to take a decision theoretic approach and choose the shrinkage parameter that minimises the expected MSE of

the multi-target estimator. It is clear that in this way the estimator of Ikeda et al. [59] also differs from TAS, which uses the marginal likelihood to weight each proposed shrinkage intensity. Again the difference can be seen as incorporating uncertainty rather than achieving optimality. Peer-reviewers have identified the comparison of all of these multi-target shrinkage estimators as a necessary piece of future work for publication of TAS.

In TAS, a uniform prior distribution is imposed on the support of the shrinkage intensity α. This expresses a belief that, a priori, there is no reason to favour shrinkage towards the target matrix over the sample covariance matrix. In addition to the independence assumption between the priors on α and $\mathbf{\Delta}$, this belief is expressed for every target in the target set $\mathcal{D}$. In principle, the uniform assumption on the prior of α may be relaxed. However, this might not be desirable without also relaxing the independence assumption between the priors of α and $\mathbf{\Delta}$, since otherwise the belief about the shrinkage intensity is propagated across all target matrices. Such a belief might be reasonable with knowledge only of the ratio n/p since the higher the dimensionality then the more shrinkage is expected and we have no other information about the data in order to favour any single target. However, paradoxically, this requires prior information about the data. Unfortunately though, relaxing the independence assumption means that the computation of Equation (2.6) is greatly increased. In this situation, the computational simplicity of linear shrinkage is violated and a more complex modelling approach would be advised. Therefore, the simple prior assumptions that TAS adopts express an appropriate level of uncertainty whilst remaining coherent with the methodology on which it is based.

TAS tends to allocate more weight to the most complex target matrices in the default target set, e.g. those with more parameters such as the unequal variance structure. This is done even when competing simpler target matrices achieve a similar, or higher, PRIAL — indicating that overfitting is present. The main source for this is that the parameters that characterise each of the default targets, namely the variance and correlation, are estimated from the data itself and the model does not penalise this. Though seemingly arbitrary, Hannart and Naveau [52] show that these values are often MLEs after more formally parametrising the target matrix and therefore represent another empirical Bayes-type procedure. Because of this, the marginal likelihood is more likely to favour the unequal variance structure that uses the sample variances as plug-in values. Clearly, a fully Bayesian treatment of these parameters would be most desirable, yet unsurprisingly it increases computational complexity. One heuristic approach to avoid this in TAS is by specifying a range of values for each parameter and then constructing target matrices that exhaust these values.

We envisage two main extensions for our work. Firstly, much like the performance of STS estimators depends on the choice of a target matrix, the performance of the

proposed TAS estimator depends on the choice of a target set. In the absence of relevant prior information, we constructed a *default* target set using shrinkage target matrices that are popular in the STS literature. However, further research is required to determine a more comprehensive and generic default target set. Such a target set would ideally cover a wide range of structures, to ensure that there is enough flexibility in the shrinkage. The latter must also take into account that, if the chosen target set contains shrinkage targets with overlapping shape, shrinkage weights need to be interpreted together with the pairwise Frobenius distance between targets, and their distance to the empirical covariance. Finally, it would be useful to extend the present work to non-Gaussian settings to allow for example the analysis of count data obtained from RNA sequencing experiments. These experiments provide greater specificity with higher throughput than array-based technologies. Potential avenues include hierarchical latent representations [3, 48] and data transformation strategies [28, 18, 113]. Nonetheless, as normal approximations can have good performance in RNA sequencing data [e.g. 65], we foresee that the current TAS estimator might have practical utility in such contexts.

Chapter 3

Covariance estimation through Probabilistic Principal Component Analysis

3.1 Introduction

In the previous chapter, we considered linear shrinkage estimators for covariance matrix estimation. In the present chapter, we consider probabilistic principal component analysis (PPCA) [92, 102], a model-based generalisation of conventional principal component analysis (PCA), which is primarily used for dimension reduction. PPCA is a specific type of factor model with isotropic Gaussian noise, where inference of model parameters is typically performed via maximum likelihood estimation using the expectation-maximisation (EM) algorithm or, under a Bayesian framework, using variational Bayesian (VB) inference.

As well as putting PCA on a principled, probabilistic footing, the PPCA model provides an estimate of the covariance matrix and its inverse. The interpretation of PPCA as a covariance model of high-dimensional data has long been known (e.g. Section 4.3 of [102]), but it has been under-used in practice. This is despite having two substantial benefits: (i) the generative data model allows missing values to be straightforwardly handled and, (ii) due to its low-dimensional representation, the inverse of a PPCA covariance matrix can be computed very efficiently.

Ma [73], Cai et al. [22] both consider the PPCA model as a means for high dimensional covariance matrix estimation, focussing on sparse estimation of the columns of the loadings matrix $\boldsymbol{W}$ via hard-threshold penalisation. This leads to estimates of W that retain the most important latent directions of variance, and estimate the least important directions by the zero vector. This can be seen as reducing estimation error from variance at the cost of increasing potential bias from false zero values. Although

this a large area of the literature for regularisation, in this thesis we do not focus on sparse estimation and so we do not revisit these works in this Chapter.

Fan et al. [42, 43, 41] considers AFM for high dimensional covariance matrix estimation. The basis of the method is also to apply a threshold. Instead, the threshold is applied to the dense covariance matrix of the sample errors $\hat{\boldsymbol{E}}$, which is obtained from $\boldsymbol{S}$ in the sample analogue of Equation (1.7). Again, the purpose is to reduce error by discarding small variance, at the expense of introducing bias through sparsity. This model has been well-studied by the authors and many theoretical properties have been established for it using classical and high-dimensional asymptotics [42, 43, 41]. However in this Chapter, we restrict our scope to PPCA models only and so do not revisit this either.

One thing to note about the papers mentioned above, and the factor model references from Section 1.3.2, is that they are complex notationally, mathematically, and most are computationally too. At the same time, it is not uncommon for the derivations of results to not be presented and software to implement the methods to not be available. This presents a limitation when attempting to numerically compare methods, especially for those new to the field or unfamiliar with the technical details contained within. This is augmented by the natural variation in notation used by different authors. This motivates some of the contributions of the present Chapter.

In this Chapter, we (i) provide a unified overview and comparison of three existing algorithms for performing PPCA, (ii) extend an existing PPCA method to the case of missing values, (iii) provide detailed mathematical derivations that are missing from the literature, (iv) provide more efficient implementations of these algorithms, (v) present these algorithms in a well-documented open source `R`-package `pcaNet` available at http://github.com/HGray384/pcaNet, (vi) illustrate its usefulness on synthetic as well as real data, and (vii) discuss the use of PPCA for high-dimensional (inverse-) covariance matrix estimation. Throughout, we focus on computational aspects, particularly on the treatment of missing values and on how to select an appropriate number of latent dimensions.

This chapter is organised as follows. In Section 3.2, we introduce the PPCA model and properties of the associated covariance matrix estimator. In Section 3.3, we present maximum likelihood estimation of the PPCA model with and without missing values using EM algorithms. Section 3.4 presents Bayesian estimation of the PPCA model as well as fast approximate inference using VB with and without missing values. In Section 3.5, we discuss the selection of the latent dimension size in the context of covariance matrix estimation. Section 3.6 highlights the attractive property of the PPCA model for computing the inverse covariance matrix. Section 3.7 presents numerical simulations from model-based data-generation protocols of the three PPCA algorithms to see how they compare with varying numbers of missing values and data dimensions

in terms of covariance estimation accuracy and timing. In Section 3.8 we compare the performance of three PPCA algorithms on real data. In Section 3.9 we apply our method to the Arabidopsis thaliana dataset. Finally, Section 3.10 concludes with a discussion of the results presented in this chapter.

3.2 The PPCA framework

Recall from Section 1.3.2 that the PPCA model is

$$\boldsymbol{x}_j = \boldsymbol{W}\boldsymbol{z}_j + \boldsymbol{\mu} + \boldsymbol{\epsilon}, \tag{3.1}$$

where $\boldsymbol{x}_j$ is a p-dimensional observation (column) in the data matrix $\boldsymbol{X} = (\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n)$, and $\boldsymbol{z}_j$ is a column of $\boldsymbol{Z} = (\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n)$ containing latent factors $\boldsymbol{z}_j \in \mathbb{R}^q$. The $p \times q$ projection matrix $\boldsymbol{W}$ links the two sets of variables, while $\boldsymbol{\mu}$ permits the model to have non-zero mean. Noise is incorporated into the model through $\boldsymbol{\epsilon}$. In PPCA, an isotropic Gaussian noise model is assumed, so that $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0},\sigma^2\boldsymbol{I}_{p\times p})$. This can be interpreted as each of the p dimensions of $\boldsymbol{x}_j$ having independent random variances of equal magnitude σ^2. We note that $\boldsymbol{x}_j$ may possess missing values and assume throughout that these are missing at random [71].

The latent variables are defined to be independent and identically distributed Gaussian variables with unit variance, i.e. $\boldsymbol{z}_j \sim \mathcal{N}(\boldsymbol{0},\boldsymbol{I}_{q\times q})$. We have that $\boldsymbol{x}_j|\boldsymbol{z}_j \sim \mathcal{N}(\boldsymbol{W}\boldsymbol{z}_j + \boldsymbol{\mu},\sigma^2\boldsymbol{I}_{p\times p})$, therefore marginalising over $\boldsymbol{z}_j$ gives $\boldsymbol{x}_j \sim \mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})$, where

$$\boldsymbol{\Sigma} = \boldsymbol{W}\boldsymbol{W}^\top + \sigma^2\boldsymbol{I}_{p\times p}. \tag{3.2}$$

$\boldsymbol{\Sigma}$ is therefore the covariance matrix for the data assuming the PPCA model. This formulation of $\boldsymbol{\Sigma}$ is highly structured, with covariance information contained within $\boldsymbol{W}\boldsymbol{W}^\top$. The extra variance information contained within the diagonal matrix $\sigma^2\boldsymbol{I}_{p\times p}$ can be seen as a regularising term. The rank of the matrix $\boldsymbol{W}$ is $q \ll p$ since it represents a projection of the latent variables into the observed space. This means that the matrix $\boldsymbol{W}\boldsymbol{W}^\top$ is rank deficient, also with rank $q \ll p$. This equates to the smallest $p-q$ eigenvalues of $\boldsymbol{W}\boldsymbol{W}^\top$ being equal to zero. The addition of the term $\sigma^2\boldsymbol{I}_{p\times p}$ effectively sets these smallest eigenvalues to be equal to σ^2 and so yields a matrix of full rank that can be considered an acceptable covariance matrix. The result is that $\boldsymbol{\Sigma}$ is now invertible, hence a solution has been found by constraining the set of possible covariance matrices to those of the form of Equation (3.2). Note that $\boldsymbol{\Sigma}$ of this form may not be well conditioned. This is the case whenever σ^2 is small, or the largest eigenvalue of $\boldsymbol{W}\boldsymbol{W}^\top$ is very large.

Equation (3.2) indicates that estimation of the covariance matrix, $\boldsymbol{\Sigma}$, can be addressed via inference of $\boldsymbol{W}$ and σ. This can be performed via Bayesian or non-Bayesian methods [92, 102, 15]. We first discuss the latter.

3.3 Non-Bayesian methods

In this section, we inspect non-Bayesian approaches to perform inference under the model displayed in Equation (3.1). These estimation procedures rely upon maximum likelihood estimation. Whilst a closed form solution for the maximum likelihood estimates exists, we show that their computation can be challenging for high-dimensional and incomplete datasets. In this situation, the EM algorithm [75] provides a way to perform maximum likelihood estimation whilst overcoming these challenges.

3.3.1 Closed-form maximum likelihood inference

According to the PPCA model $\boldsymbol{x}_j \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for each independent sample $\boldsymbol{x}_j$, $j = 1, \ldots, n$. The log-likelihood is then:

$$\ln \mathrm{p}(\boldsymbol{X} | \boldsymbol{\mu}, \boldsymbol{W}, \sigma^2) = -\frac{np}{2} \ln 2\pi - \frac{n}{2} \ln |\boldsymbol{\Sigma}| - \frac{1}{2} \sum_{j=1}^{n} (\boldsymbol{x}_j - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}_j - \boldsymbol{\mu}). \tag{3.3}$$

The maximum likelihood solutions are obtained analytically [102] as

$$\boldsymbol{\mu}_{\mathrm{ML}} = \bar{\boldsymbol{x}} = \frac{1}{n} \sum_{j=1}^{n} \boldsymbol{x}_j, \tag{3.4}$$

$$\boldsymbol{W}_{\mathrm{ML}} = \boldsymbol{U} (\boldsymbol{\Lambda} - \sigma^2_{\mathrm{ML}} \boldsymbol{I}_{q \times q})^{\frac{1}{2}} \boldsymbol{R}, \tag{3.5}$$

$$\sigma^2_{\mathrm{ML}} = \frac{1}{p - q} \sum_{s=q+1}^{p} \lambda_s, \tag{3.6}$$

where $\boldsymbol{U} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_q)$ is the matrix of eigenvectors of $\boldsymbol{S}$ which correspond to its q largest eigenvalues $\lambda_1, \ldots, \lambda_q$ (in decreasing order or magnitude), $\boldsymbol{\Lambda}$ is the diagonal matrix whose non-zero elements correspond $\lambda_1, \ldots, \lambda_q$, and $\boldsymbol{R}$ is an orthogonal matrix.

Despite the convenient availability of these estimators in closed-form, they come with notable disadvantages for our purposes. In particular, $\boldsymbol{W}_{\mathrm{ML}}$ and σ^2_{ML} rely upon the eigen-decomposition of $\boldsymbol{S}$. In high-dimensional situations, the eigenvalues and eigenvectors are not reliable since they are biased. In addition, they are unavailable for an incomplete data matrix, e.g. when there are missing values. Here, we focus on two alternative implementations (based on the EM algorithm) that are able to overcome these challenges [92, 60].

3.3.2 Estimation using EM

The EM algorithm is a method for finding maximum likelihood solutions for models with latent variables [75]. The general idea is to maximise a lower bound of the logarithm of the likelihood. The EM algorithm guarantees an increase in the lower bound at each iteration until it converges to the global solution. If the global maximum has been found, then the parameters that are estimated to maximise the lower bound at convergence are then equal to the maximum likelihood solutions (Section 1.3.3). We first consider the EM application to the PPCA model without missing values. Then we focus on the implementations of [98] and [60], who incorporate the presence of missing values.

EM algorithm in PPCA without missing values

In this section, we present the E and M step updates for the algorithm, with the full derivation being provided in Appendix B.1. Since the update equations are available in the literature, the contribution of this Section is to provide the full derivations in a consistent notation.

E step

In the E step, we update the moments of the latent variables as follows:

$$\mathbb{E}\left[z_j\right] = \boldsymbol{M}^{-1}\boldsymbol{W}^\top(\boldsymbol{x}_j - \boldsymbol{\mu}) \tag{3.7}$$

$$\mathbb{E}\left[z_j z_j^\top\right] = \sigma^2\boldsymbol{M}^{-1} + \mathbb{E}\left[z_j\right]\mathbb{E}[z_j]^\top. \tag{3.8}$$

M step

In the M step, we update the model parameters using:

$$\boldsymbol{\mu}_{\text{new}} = \frac{1}{n}\sum_{j=1}^{n}\left(\boldsymbol{x}_j - \boldsymbol{W}\mathbb{E}\left[z_j\right]\right). \tag{3.9}$$

$$\boldsymbol{W}_{\text{new}} = \left(\sum_{j=1}^{n}(\boldsymbol{x}_j - \boldsymbol{\mu})\mathbb{E}\left[z_j\right]^\top\right)\left(\sum_{j=1}^{n}\mathbb{E}\left[z_j z_j^\top\right]\right)^{-1} \tag{3.10}$$

$$\begin{aligned}\sigma^2_{\text{new}} = &\frac{1}{np}\sum_{j=1}^{n}\|\boldsymbol{x}_j - \boldsymbol{\mu}\|^2 + \text{Tr}\left[\mathbb{E}\left[z_j z_j^\top\right]\boldsymbol{W}_{\text{new}}^\top\boldsymbol{W}_{\text{new}}\right] \\ &- 2\mathbb{E}\left[z_j\right]^\top\boldsymbol{W}_{\text{new}}^\top(\boldsymbol{x}_j - \boldsymbol{\mu}),\end{aligned} \tag{3.11}$$

Implementation

The EM algorithm then consists of randomly initialising σ^2 and $\boldsymbol{W}$, and then iteratively evaluating Equations (3.7), (3.8), (3.9), (3.10) and (3.11).

EM algorithm in PPCA with missing values

We now consider two adaptations of the above EM algorithm in situations when there are missing values. Adopting the notation of Ilin and Raiko [60], we define O to be the set of indices i,j for which x_{ij} is observed (i.e. non-missing), O_i to be the set of indices j for which x_{ij} is observed for a fixed $i = 1,\ldots,p$, and O_j to be the set of indices i for which x_{ij} is observed for a fixed $j = 1,\ldots,n$. We also denote the sub-vector of $\boldsymbol{x}_j$ that contains only its observed (i.e. non-missing) values as $\boldsymbol{x}_j^{(O_j)}$, noting that its length will now be $|O_j|$. Similarly, we denote $\boldsymbol{\mu}^{(O_j)}$ to be the corresponding mean vector of $\boldsymbol{x}_j^{(O_j)}$, and use a similar notation for other vectors. Analogously, we denote $\boldsymbol{W}^{(O_j)}$ as the $(|O_j| \times q)$ sub-matrix of $\boldsymbol{W}$ that has only retained the rows with indices $i \in O_j$, and use a similar notation with other matrices.

EM algorithm 1

We consider the algorithm of Stacklies et al. [98] that is identical to that described in Section 3.3.2, except that (i) the empirical mean calculated using only the observed (i.e. non-missing) values is initially subtracted from the data, and then $\boldsymbol{\mu}_{\mathrm{ML}}$ is assumed to be zero and (ii) at the start of each E step, the missing values are replaced with their projection estimates. That is, if x_{ij} is a missing value, it is estimated as the i-th element of $\boldsymbol{W}\mathbb{E}[\boldsymbol{z}_j]$, where $\boldsymbol{W}$ is from the latest M step, and $\mathbb{E}[\boldsymbol{z}_j]$ is from the latest E step.

The assumption that $\boldsymbol{\mu}_{\mathrm{ML}}$ is equal to the empirical mean using only the observed values is made so that it may be omitted from the estimation steps. It is important to note that this is a heuristic step, since this empirical mean is not the maximum likelihood estimate of $\boldsymbol{\mu}$. It is unclear whether or not this provides a reasonable approximation in practice, and equally unclear as to the impact this has on the quality of the estimated values for the other parameters.

Although the full derivations are not presented by the authors for this algorithm, there is sufficient detail in Porta et al. [87] (and the accompanying technical note by Jacob Verbeek) to be able to derive the equations independently. The contribution of this Section is therefore in providing all of the derivations (Appendix B.2) in a consistent (with respect to other methods) notation. The update equations are available in the original publications.

E step

For the E step, we have

$$\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j\right] = \boldsymbol{M}^{-1}\boldsymbol{W}^\top \widetilde{\boldsymbol{x}}_j, \tag{3.12}$$

$$\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j z_j^\top\right] = \sigma^2 \boldsymbol{M}^{-1} + \mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j\right] \mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top \tag{3.13}$$

$$\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right] = \boldsymbol{W}^{(M_j)} \mathbb{E}_{z_j}\left[z_j\right]. \tag{3.14}$$

$$\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)} \boldsymbol{x}_j^{(M_j)\top}\right] = \sigma^2 \boldsymbol{I}_{(p-|O_j|)\times(p-|O_j|)} + \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}, z_j}\left[\boldsymbol{x}_j^{(M_j)}\right] \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}, z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]^\top. \tag{3.15}$$

where $\boldsymbol{M} = \boldsymbol{W}^\top \boldsymbol{W} + \sigma^2 \boldsymbol{I}_{q\times q}$, and $\widetilde{\boldsymbol{x}}_j$ is $\boldsymbol{x}_j$ for $\boldsymbol{x}_j^{(O_j)}$ and $\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right]$ for $\boldsymbol{x}_j^{(M_j)}$, i.e. with missing values replaced by their expectation.

M step

For the M step, we get

$$\boldsymbol{W}_{\text{new}} = \widetilde{\boldsymbol{X}}\overline{\boldsymbol{Z}}^\top \left(n\sigma^2 \boldsymbol{M}^{-1} + \overline{\boldsymbol{Z}\boldsymbol{Z}}^\top\right)^{-1}, \tag{3.16}$$

$$\sigma^2_{\text{new}} = \frac{1}{np}\left(n\text{Tr}\left[\boldsymbol{W}\sigma^2_{\text{old}}\boldsymbol{M}^{-1}\boldsymbol{W}^\top\right] + \sum_{j=1}^{n}\left\|\widetilde{\boldsymbol{x}}_j - \boldsymbol{W}\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right\|^2 + \sigma^2_{\text{old}}\sum_{j=1}^{n}(p - |O_j|)\right). \tag{3.17}$$

Implementation

The algorithm is executed by subtracting $\boldsymbol{\mu}_{\text{ML}}$ from the dataset and then iterating through Equations (3.12), (3.13), (3.14), (3.15),(3.16), (3.17) until convergence. Note that these updates depend upon the current estimate of the missing values.

EM algorithm 2

Here we consider the algorithm of Ilin and Raiko [60]. This approach differs to that of Stacklies et al. [98] in that $\boldsymbol{\mu}$ is updated at each iteration and only the observed values are used in the expectation and parameter updates.

Again, for this algorithm the update equations are available in Ilin and Raiko [60]. This means that the contribution of this Section is to present the full derivations (Appendix B.3) in notation that is consistent with the other algorithms within this Chapter.

E step

Analogously to Equations (3.7) and (3.8), the E step updates are

$$\mathbb{E}[\boldsymbol{z}_j] = \left(\boldsymbol{M}^{(O_j)}\right)^{-1} \sum_{i \in O_j} \boldsymbol{w}_i (x_{ij} - \mu_i) \tag{3.18}$$

$$\mathbb{E}\left[\boldsymbol{z}_j \boldsymbol{z}_j^\top\right] = \sigma^2 \left(\boldsymbol{M}^{(O_j)}\right)^{-1} + \mathbb{E}[\boldsymbol{z}_j]\mathbb{E}[\boldsymbol{z}_j]^\top. \tag{3.19}$$

M step

The M step updates are

$$(\mu_i)_{\text{new}} = \frac{1}{|O_i|} \sum_{j \in O_i} \left(x_{ij} - \boldsymbol{w}_i^\top \mathbb{E}[\boldsymbol{z}_j]\right) \tag{3.20}$$

$$(\boldsymbol{w}_i)_{\text{new}} = \left(\sum_{j=1}^{n} (x_{ij} - \mu_i)\mathbb{E}[\boldsymbol{z}_j]^\top\right)\left(\sum_{j \in O_i} \mathbb{E}\left[\boldsymbol{z}_j \boldsymbol{z}_j^\top\right]\right)^{-1} \tag{3.21}$$

$$\begin{aligned}\sigma^2_{\text{new}} = &\frac{1}{|O|} \sum_{ij \in O} (\boldsymbol{x}_{ij} - \mu_i)^2 + \operatorname{Tr}\left[\mathbb{E}\left[\boldsymbol{z}_j \boldsymbol{z}_j^\top\right] (\boldsymbol{w}_i)_{\text{new}} (\boldsymbol{w}_i)_{\text{new}}^\top\right] \\ &- 2(\boldsymbol{w}_i)_{\text{new}}^\top \mathbb{E}[\boldsymbol{z}_j]^\top (x_{ij} - \mu_i).\end{aligned} \tag{3.22}$$

We see that Equations (3.9), (3.10), and (3.11) can be derived as special cases of those above when there are no missing values.

Implementation

The algorithm is executed by initialising $\boldsymbol{W}, \boldsymbol{\mu}, \sigma^2, \boldsymbol{Z}$ and then iterating through Equations (3.18), (3.19), (3.20), (3.21), (3.22) until convergence. Note that in this algorithm the missing values are not estimated with each iteration, they are imputed after convergence.

A crucial point noted in Ilin and Raiko [60] is that, unlike in the complete data case when $\boldsymbol{\mu}_{\text{ML}} = \bar{\boldsymbol{x}}$, Equation (3.20) depends on the current estimates of $\mathbb{E}[\boldsymbol{z}_j]$ and $\boldsymbol{w}_i$ and so must be updated at each iteration. This is in contrast to Stacklies et al. [98], who opt to precompute and subtract the sample mean of the observed values. Another consideration is that the computations required to perform updates (3.18) and (3.21) are generally heavier than (3.7) and (3.10) because they must update each row (or column) separately using only the observed values indices.

3.4 Bayesian Methods

In this section we inspect Bayesian approaches to perform parameter estimation for the model introduced in Equation (3.1). This involves defining a prior distribution for the model parameters and then determining their posterior distribution using the log-likelihood in Equation (3.3). Summaries of the posterior distribution for each parameter can then be used as point estimates for the parameters (e.g. the posterior mean). With this model specification, the posterior distribution is intractable and so other methods of evaluation must be employed. Here, we employ the approximate inference procedure of VB to this problem and present two algorithms that implement this. The latter are computationally efficient and can handle missing values much alike the EM algorithm. Unlike EM however, this Bayesian approach offers the opportunity to automatically select the latent dimension q.

Denote $\boldsymbol{w}_g$ to be the g-th column of $\boldsymbol{W}$, noting that the subscript $g = 1,\ldots,q$ has a different domain to subscript i (used for rows of $\boldsymbol{W}$) and should not be confused with it. We specify independent priors on the columns of $\boldsymbol{W}$ as follows

$$\boldsymbol{w}_g \sim \mathcal{N}(\boldsymbol{0}, \nu_{w,g}\boldsymbol{I}_{p\times p}), \tag{3.23}$$

so that we have column-specific prior variances, $\nu_{w,g}$. This form of prior is known as automatic relevance determination (ARD) [74]. For small values of $\nu_{w,g}$, the corresponding column $\boldsymbol{w}_g$ is estimated as approximately the zero-vector. This results in no influence from this dimension in the projection of $\boldsymbol{W}$ to the observed space, and so that latent dimension is essentially removed. Initialising $q = p - 1$ then allows the prior to automatically consider and effectively remove each potential latent dimension.

Priors for $\nu_{w,g}$, $\boldsymbol{\mu}$, σ, and their hyperparameters may also then be set and will be considered in the next sections. However, Equation (3.23) alone makes the resulting posterior distribution for the parameters intractable. This means that inference for these parameters is not possible in closed-form and also that the EM algorithm cannot be used. In order to still perform this inference exactly, numerical procedures such as Markov chain Monte Carlo (MCMC) must be used. MCMC involves repeated sampling from the target distribution in order to evaluate the desired integral. This type of approach induces a large computational cost, which becomes infeasible for large dimensional problems. An alternative method is to keep computational costs relatively low by resorting to approximate inference. This can be done through the VB framework, which we describe in the next section.

3.4.1 Estimation using VB

In order to bypass the difficulty of an intractable posterior distribution, we may employ VB. Recall from Section 1.4.4, that VB is an EM-like algorithm in which the desired posterior distribution is often approximated by a specific simpler structure, e.g. fully factored. The result is that the approximation itself is tractable and may be used to estimate the lower bound of the true posterior, thereby proceeding in similar fashion to EM. The challenge is thus to find an approximation that is as close to the desired distribution, whilst retaining sufficient flexibility to represent it well. Similarly to the EM section, we first present the application of VB to the PPCA model without missing values, then we describe two algorithms; an extension to Agarwal and Bishop [1], and Ilin and Raiko [60], both in the presence of missing values.

VB algorithm in PPCA without missing values

Using the VB method (Section 1.4.4) to overcome an intractable posterior distribution, we complete the model specification for Bayesian PPCA as in Bishop [15].

Likelihood

Defining $\tau \equiv \sigma^{-2}$, the likelihood for the data is

$$\mathrm{p}(\boldsymbol{X}|\boldsymbol{Z},\boldsymbol{W},\boldsymbol{\mu},\tau) = \prod_{j=1}^{n} \mathrm{p}\left(\boldsymbol{x}_j|\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right) = \prod_{j=1}^{n} \mathcal{N}\left(\boldsymbol{W}\boldsymbol{z}_j+\boldsymbol{\mu},\tau\boldsymbol{I}_{p\times p}\right). \tag{3.24}$$

Joint distribution

The joint distribution of the latent and observed variables has the factored form:

$$\mathrm{p}(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{W},\boldsymbol{\mu},\tau) = \mathrm{p}(\boldsymbol{X}|\boldsymbol{Z},\boldsymbol{W},\boldsymbol{\mu},\tau)\mathrm{p}(\boldsymbol{Z})\prod_{g=1}^{q}\mathrm{p}(\boldsymbol{w}_g|\nu_{w,g})\mathrm{p}(\nu_{w,g})\mathrm{p}(\boldsymbol{\mu})\mathrm{p}(\tau). \tag{3.25}$$

Priors

The prior specification is

$$\mathrm{p}(\boldsymbol{Z}) = \prod_{j=1}^{n} \mathrm{p}\left(\boldsymbol{z}_j\right) \tag{3.26}$$

$$\boldsymbol{z}_j \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{q\times q}) \tag{3.27}$$

$$\boldsymbol{\mu} \sim \mathcal{N}(\boldsymbol{0}, \nu_\mu \boldsymbol{I}_{p\times p}) \tag{3.28}$$

$$\mathrm{p}(\boldsymbol{W}) = \prod_{g=1}^{q} \mathrm{p}\left(\boldsymbol{w}_g\right) \tag{3.29}$$

$$\boldsymbol{w}_g \sim \mathcal{N}(\boldsymbol{0}, \nu_{w,g} \boldsymbol{I}_{p\times p}) \tag{3.30}$$

$$\nu_{w,g} \sim \Gamma(a_\nu, b_\nu) \tag{3.31}$$

$$\tau \sim \Gamma(c_\tau, d_\tau), \tag{3.32}$$

where $\Gamma(\cdot)$ denotes the Gamma distribution. Broad priors are advised by setting $a_\nu = b_\nu = c_\tau = d_\tau = 0.001$ and $\nu_\mu = 1000$.

Variational approximation

From Equation (1.20), the variational mean field approximation takes the form

$$\mathrm{q}(\boldsymbol{Z}, \boldsymbol{W}, \nu_{w,g}, \boldsymbol{\mu}, \tau) = \mathrm{q}(\boldsymbol{Z})\mathrm{q}(\boldsymbol{W}) \prod_{g=1}^{q} \mathrm{q}(\nu_{w,g})\mathrm{q}(\boldsymbol{\mu})\mathrm{q}(\tau). \tag{3.33}$$

Optimal q* distributions

The optimal approximate distributions are given by

$$\mathrm{q}^*(\boldsymbol{Z}) = \prod_{j=1}^{n} \mathcal{N}(\overline{\boldsymbol{z}}_j, \boldsymbol{\Sigma}_{\boldsymbol{Z}}), \tag{3.34}$$

$$\mathrm{q}^*(\mu_i) = \mathcal{N}(\overline{\mu}_i, \tilde{\mu}_i) \tag{3.35}$$

$$\mathrm{q}^*(\boldsymbol{W}) = \prod_{i=1}^{p} \mathcal{N}(\overline{\boldsymbol{w}}_i, \boldsymbol{\Sigma}_{\boldsymbol{W}}) \tag{3.36}$$

$$\mathrm{q}^*(\nu_{w,g}) = \Gamma(\tilde{a}_\nu, \tilde{b}_{\nu g}) \tag{3.37}$$

$$\mathrm{q}^*(\tau) = \Gamma(\tilde{c}_\tau, \tilde{d}_\tau), \tag{3.38}$$

where the parameters are defined in the variational updates.

Variational updates

The variational updates are given by [15] as

$$\overline{\boldsymbol{z}}_j = \frac{\tilde{c}_\tau}{\tilde{d}_\tau}\boldsymbol{\Sigma}_{\boldsymbol{Z}} \sum_{i=1}^{p} \overline{\boldsymbol{w}}_i (x_{ij} - \overline{\mu}_i) \tag{3.39}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{Z}} = \frac{\tilde{d}_\tau}{\tilde{c}_\tau}\left(\frac{\tilde{d}_\tau}{\tilde{c}_\tau}\boldsymbol{I}_{q\times q} + \sum_{i=1}^{p}\left(\boldsymbol{\Sigma}_{\boldsymbol{w}_i} + \overline{\boldsymbol{w}}_i\overline{\boldsymbol{w}}_i^\top\right)\right)^{-1} \tag{3.40}$$

$$\overline{\mu}_i = \frac{\tilde{c}_\tau}{\tilde{d}_\tau}\tilde{\mu}_i \sum_{j=1}^{n}\left(x_{ij} - \overline{\boldsymbol{w}}_i^\top \overline{\boldsymbol{z}}_j\right) \tag{3.41}$$

$$\tilde{\mu}_i = \frac{\nu_\mu \tilde{d}_\tau}{\tilde{d}_\tau + n\nu_\mu \tilde{c}_\tau} \tag{3.42}$$

$$\overline{\boldsymbol{w}}_i = \frac{\tilde{c}_\tau}{\tilde{d}_\tau}\boldsymbol{\Sigma}_{\boldsymbol{W}} \sum_{j=1}^{n} \overline{\boldsymbol{z}}_j (x_{ij} - \overline{\mu}_i) \tag{3.43}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{W}} = \frac{\tilde{d}_\tau}{\tilde{c}_\tau}\left(\frac{\tilde{d}_\tau}{\tilde{c}_\tau}\mathrm{diag}\left(\frac{\tilde{a}_\nu}{\tilde{b}_{\nu g}}\right) + \sum_{j=1}^{n}\left(\boldsymbol{\Sigma}_{\boldsymbol{Z}} + \overline{\boldsymbol{z}}_j\overline{\boldsymbol{z}}_j^\top\right)\right)^{-1} \tag{3.44}$$

$$\tilde{a}_\nu = a_\nu + \frac{p}{2} \tag{3.45}$$

$$\tilde{b}_{\nu g} = b_\nu + \frac{\|\boldsymbol{w}_g\|^2}{2} \tag{3.46}$$

$$\tilde{c}_\tau = c_\tau + \frac{np}{2} \tag{3.47}$$

$$\tilde{d}_\tau = d_\tau + \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{p} x_{ij}^2 + \overline{\mu}_i^2 + 2\overline{\mu}_i\overline{\boldsymbol{w}}_i^\top\overline{\boldsymbol{z}}_j - 2x_{ij}\overline{\boldsymbol{w}}_i^\top\overline{\boldsymbol{z}}_j - 2x_{ij}\overline{\mu}_i + \mathrm{Tr}[\boldsymbol{\Sigma}_W\boldsymbol{\Sigma}_Z], \tag{3.48}$$

where $\mathrm{diag}\left(\tilde{a}_\nu/\tilde{b}_{\nu g}\right)$ denotes a diagonal matrix whose q non-zero elements correspond to $\tilde{a}_\nu/\tilde{b}_{\nu g}$. The VB algorithm is completed by cycling through these update equations until convergence is achieved, which may be monitored by calculating $\mathcal{L}(\mathrm{q})$.

VB algorithm in PPCA with missing values

The algorithm of Bishop [15] clearly cannot handle missing values without some adjustment. Here we consider the adjustments to the VB algorithm made by Oba et al. [82], Agarwal and Bishop [1], and Ilin and Raiko [60].

Variational algorithm 1

This contribution of this Section is a novel algorithm that extends the work of Agarwal and Bishop [1] to the case of missing values. The original intention was to derive the

algorithm of Oba et al. [82], since this (including update equations) is missing in the literature despite its popularity. Whilst attempting this derivation, it arose that the actual model is a special case of that which is presented in Agarwal and Bishop [1], except the resulting update equations seem to differ from those in the software provided by Oba et al. [82]. This Section therefore extends the algorithm of Agarwal and Bishop [1] to the case of missing values and it is ongoing work to investigate the discrepancy between the updates and code from Oba et al. [82]. Full derivations are provided in Appendix B.4 in consistent notation.

Likelihood

The conditional distribution for the observed values can be written as

$$\mathrm{p}\left(\boldsymbol{x}_j^{(O_j)}|\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right) = \mathcal{N}\left(\boldsymbol{W}^{(O_j)}\boldsymbol{z}_j+\boldsymbol{\mu}^{(O_j)},\tau^{-1}\boldsymbol{I}_{|O_j|\times|O_j|}\right). \tag{3.49}$$

Priors

The prior specification is as follows:

$$\mathrm{p}\left(\boldsymbol{x}_j^{(M_j)}|\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right) = \mathcal{N}\left(\boldsymbol{W}^{(M_j)}\boldsymbol{z}_j+\boldsymbol{\mu}^{(M_j)},\tau^{-1}\boldsymbol{I}_{(p-|O_j|)\times(p-|O_j|)}\right). \tag{3.50}$$

$$\mathrm{p}(\boldsymbol{z}_j) = \mathcal{N}(\boldsymbol{0},\boldsymbol{I}_{q\times q}) \tag{3.51}$$

$$\mathrm{p}(\boldsymbol{\mu},\boldsymbol{W},\tau|\boldsymbol{\alpha}) = \mathrm{p}(\boldsymbol{\mu}|\tau)\mathrm{p}(\boldsymbol{W}|\tau,\boldsymbol{\alpha})\mathrm{p}(\tau), \tag{3.52}$$

$$\mathrm{p}(\boldsymbol{\mu}|\tau) = \mathcal{N}(\overline{\boldsymbol{\mu}}_0,(\gamma_{\boldsymbol{\mu}_0}\tau)^{-1}\boldsymbol{I}_{p\times p}), \tag{3.53}$$

$$\mathrm{p}(\boldsymbol{W}|\tau,\boldsymbol{\alpha}) = \prod_{g=1}^{q}\mathrm{p}(\boldsymbol{w}_g|\tau,\alpha_g) = \prod_{g=1}^{q}\mathcal{N}(\boldsymbol{0},(\alpha_g\tau)^{-1}\boldsymbol{I}_{p\times p}), \tag{3.54}$$

$$\mathrm{p}(\tau) = \Gamma(\overline{\tau}_0,\gamma_{\tau_0}), \tag{3.55}$$

$$\mathrm{p}(\boldsymbol{\alpha}) = \prod_{g=1}^{q}\mathrm{p}(\alpha_g) = \prod_{g=1}^{q}\Gamma(\overline{\alpha}_0,\gamma_{\alpha_0}), \tag{3.56}$$

where $\boldsymbol{w}_g$ denotes the g-th column of matrix $\boldsymbol{W}$. In Oba et al. [82], the hyperparameters are fixed as $\overline{\boldsymbol{\mu}}_0 = 0$, $\overline{\tau}_0 = 1$, $\gamma_{\tau_0} = \gamma_{\boldsymbol{\mu}_0} = 10^{-10}$ (although $\gamma_{\boldsymbol{\mu}_0} = 10^{-3}$ in the code), $\overline{\alpha}_0 = 1$, and $\gamma_{\alpha_0} = 10^{-10}$.

Joint distribution

Using this prior and likelihood, the joint distribution for all variables factors as so:

$$\begin{aligned}\mathrm{p}(\boldsymbol{x}_j,\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau,\boldsymbol{\alpha}) =& \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)}|\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right)\mathrm{p}\left(\boldsymbol{x}_j^{(M_j)}|\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right)\mathrm{p}(\boldsymbol{z}_j)\mathrm{p}(\boldsymbol{\mu}|\tau)\\ &\times\mathrm{p}(\boldsymbol{W}|\tau,\boldsymbol{\alpha})\mathrm{p}(\tau)\mathrm{p}(\boldsymbol{\alpha})\end{aligned} \tag{3.57}$$

Variational approximation

From email correspondence with Shigeyuki Oba, author of Oba et al. [82], the variational approximation that is used takes the form

$$\mathrm{q}\left(\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)},\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{W},\tau,\boldsymbol{\alpha}\right)=\prod_{j=1}^{n}\mathrm{q}\left(\boldsymbol{x}_j^{(M_j)}\right)\mathrm{q}(\boldsymbol{z}_j)\mathrm{q}(\boldsymbol{\mu},\boldsymbol{W},\tau)\mathrm{q}(\boldsymbol{\alpha}). \tag{3.58}$$

Optimal q^* distributions

$$\mathrm{q}^*\left(\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)}\right)=\prod_{j=1}^{n}\mathrm{q}^*\left(\boldsymbol{x}_j^{(M_j)}\right)=\prod_{j=1}^{n}\mathcal{N}\left(\overline{\boldsymbol{x}}_j^{(M_j)},\boldsymbol{\Sigma}_{\boldsymbol{x}_j^{(M_j)}}\right) \tag{3.59}$$

$$\mathrm{q}^*(\boldsymbol{Z})=\prod_{j=1}^{n}\mathrm{q}^*(\boldsymbol{z}_j)=\prod_{j=1}^{n}\mathcal{N}(\overline{\boldsymbol{z}}_j,\boldsymbol{\Sigma}_z) \tag{3.60}$$

$$\mathrm{q}^*(\boldsymbol{\mu},\boldsymbol{W},\tau)=\mathrm{q}^*(\boldsymbol{\mu}|\boldsymbol{W},\tau)\mathrm{q}^*(\boldsymbol{W}|\tau)\mathrm{q}^*(\tau) \tag{3.61}$$

$$\mathrm{q}^*(\boldsymbol{\mu}|\boldsymbol{W},\tau)=\mathcal{N}(\boldsymbol{W}\boldsymbol{s}_{\boldsymbol{\mu}}+\boldsymbol{m}_{\boldsymbol{\mu}},(\gamma_{\boldsymbol{\mu}}\tau)^{-1}\boldsymbol{I}_{p\times p}) \tag{3.62}$$

$$\mathrm{q}^*(\boldsymbol{W}|\tau)=\prod_{i=1}^{p}\mathcal{N}(\boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i},(\tau\boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}})^{-1}) \tag{3.63}$$

$$\mathrm{q}^*(\tau)=\Gamma(\overline{\tau},\gamma_\tau) \tag{3.64}$$

$$\mathrm{q}^*(\boldsymbol{\alpha})=\prod_{j=1}^{q}\Gamma(\overline{\alpha}_j,\gamma_\alpha) \tag{3.65}$$

Variational updates

$$\boldsymbol{\Sigma}_{\boldsymbol{x}_j^{(M_j)}} = \mathbb{E}_{\tau}\left[\tau\right]^{-1} \boldsymbol{I}_{(p-|O_j|)\times(p-|O_j|)} \tag{3.66}$$

$$\overline{\boldsymbol{x}}_j^{(M_j)} = \boldsymbol{\Sigma}_{\boldsymbol{x}_j^{(M_j)}} \left(\mathbb{E}_{\boldsymbol{W},\tau,\boldsymbol{z}_j}\left[\tau \boldsymbol{W}^{(M_j)} \boldsymbol{z}_j\right] + \mathbb{E}_{\boldsymbol{\mu},\tau}\left[\tau \boldsymbol{\mu}^{(M_j)}\right]\right) \tag{3.67}$$

$$\boldsymbol{\Sigma}_z = \left(\boldsymbol{I}_{q\times q} + \mathbb{E}_{\boldsymbol{W},\tau}\left[\tau \boldsymbol{W}^\top \boldsymbol{W}\right]\right)^{-1}, \tag{3.68}$$

$$\overline{\boldsymbol{z}}_j = \boldsymbol{\Sigma}_z \left(\mathbb{E}_{\boldsymbol{W},\tau}\left[\tau \boldsymbol{W}\right]^\top - \mathbb{E}_{\boldsymbol{W},\boldsymbol{\mu},\tau}\left[\tau \boldsymbol{W}^\top \boldsymbol{\mu}\right]\right) \tag{3.69}$$

$$\gamma_{\boldsymbol{\mu}} = \gamma_{\boldsymbol{\mu}_0} + n, \tag{3.70}$$

$$\boldsymbol{s}_{\boldsymbol{\mu}} = -\frac{1}{\gamma_{\boldsymbol{\mu}}} \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right], \tag{3.71}$$

$$\boldsymbol{m}_{\boldsymbol{\mu}} = \frac{1}{\gamma_{\boldsymbol{\mu}}} \left(\gamma_{\boldsymbol{\mu}_0} \overline{\boldsymbol{\mu}}_0 + \sum_{j=1}^{n} \widetilde{\boldsymbol{x}}_j\right) \tag{3.72}$$

$$\boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}} = \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j \boldsymbol{z}_j^\top\right] - \gamma_{\boldsymbol{\mu}} \boldsymbol{s}_{\boldsymbol{\mu}} \boldsymbol{s}_{\boldsymbol{\mu}}^\top + \operatorname{diag}\left(\mathbb{E}_{\boldsymbol{\alpha}}\left[\boldsymbol{\alpha}\right]\right) \tag{3.73}$$

$$\boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i} = \boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}}^{-1} \left(\sum_{j=1}^{n} \widetilde{x}_{ij} \mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right] + \gamma_{\boldsymbol{\mu}} m_{\boldsymbol{\mu} i} \boldsymbol{s}_{\boldsymbol{\mu}}\right) \tag{3.74}$$

$$\gamma_\tau = \frac{np}{2} + \gamma_{\tau_0} \tag{3.75}$$

$$\begin{aligned}\overline{\tau} = \gamma_\tau \Bigg(&\gamma_{\tau_0} \overline{\tau}_0^{-1} + \frac{1}{2} \sum_{j=1}^{n} \boldsymbol{x}_j^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} + \frac{1}{2} \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{x}_j^{(M_j)}\right] \\ &+ \frac{\gamma_{\boldsymbol{\mu}_0}}{2} \overline{\boldsymbol{\mu}}_0^\top \overline{\boldsymbol{\mu}}_0 - \frac{\gamma_{\boldsymbol{\mu}}}{2} \boldsymbol{m}_{\boldsymbol{\mu}}^\top \boldsymbol{m}_{\boldsymbol{\mu}} - \frac{1}{2} \sum_{i=1}^{p} \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i}^\top \boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}} \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i}\Bigg)^{-1}\end{aligned} \tag{3.76}$$

$$\gamma_\alpha = \gamma_{\alpha_0} + \frac{p}{2} \tag{3.77}$$

$$\overline{\alpha}_j = \gamma_\alpha \left(\gamma_{\alpha_0} \overline{\alpha}_0^{-1} + \frac{1}{2} \mathbb{E}_{\boldsymbol{w}_j,\tau}\left[\tau \boldsymbol{w}_j^\top \boldsymbol{w}_j\right]\right)^{-1}. \tag{3.78}$$

Implementation

The algorithm is not yet implemented in software, but it is executed in similar fashion to the others. The algorithm is executed by iterating through Equations (3.66), (3.67), (3.68), (3.69), (3.70), (3.71), (3.73), (3.74), (3.75), (3.76), (3.77), (3.78). Note that here, the missing values are treated as latent variables and so are re-estimated with each iteration of the algorithm. In this algorithm, the ARD prior now considers the ratio of variance parameters $\nu_{w,g}$ and τ. In order for a latent dimension to be suppressed the

column variance $\nu_{w,g}$ must now be outweighed by the variance of the model noise τ. Stated alternatively, the product of $\nu_{w,g}$ and σ^2 must be close to zero.

Variational algorithm 2

We now focus on the algorithm from Ilin and Raiko [60], full derivations of all equations are given in Appendix B.5. Since the update equations are provided in the paper (without derivation), the contribution of this Section is to present the derivations and to do so using consistent notation.

Likelihood

The conditional distribution for the observed values can be written as

$$\mathrm{p}\left(\boldsymbol{x}_j^{(O_j)} | \boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau\right) = \mathcal{N}\left(\boldsymbol{W}^{(O_j)} \boldsymbol{z}_j + \boldsymbol{\mu}^{(O_j)}, \sigma^2 \boldsymbol{I}_{|O_j| \times |O_j|}\right). \tag{3.79}$$

Priors

The prior specification is

$$\mathrm{p}(\boldsymbol{Z}) = \prod_{j=1}^{n} \mathrm{p}\left(\boldsymbol{z}_j\right) \tag{3.80}$$

$$\mathrm{p}\left(\boldsymbol{z}_j\right) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{q \times q}) \tag{3.81}$$

$$\mathrm{p}(\boldsymbol{\mu}) = \mathcal{N}(\boldsymbol{0}, \nu_\mu \boldsymbol{I}_{p \times p}) \tag{3.82}$$

$$\mathrm{p}(\boldsymbol{W}) = \prod_{g=1}^{q} \mathrm{p}\left(\boldsymbol{w}_g\right) \tag{3.83}$$

$$\mathrm{p}\left(\boldsymbol{w}_g\right) = \mathcal{N}(\boldsymbol{0}, \nu_{w,g} \boldsymbol{I}_{p \times p}). \tag{3.84}$$

In this algorithm, $\sigma^2, \nu_{w,g}, \nu_\mu$ are treated as hyperparameters to be estimated.

Joint distribution

The full joint model is

$$\mathrm{p}(\boldsymbol{X}, \boldsymbol{W}, \boldsymbol{Z}, \boldsymbol{\mu}) = \prod_{ij \in O} \mathrm{p}(x_{ij} | \boldsymbol{w}_g, \boldsymbol{z}_j, \mu_i) \prod_{g=1}^{q} \mathrm{p}(\boldsymbol{w}_g) \prod_{j=1}^{n} \mathrm{p}(\boldsymbol{z}_j) \prod_{i=1}^{p} \mathrm{p}(\mu_i). \tag{3.85}$$

Variational approximation

We seek to approximate the posterior distribution, $\mathrm{p}(\boldsymbol{W}, \boldsymbol{Z}, \boldsymbol{\mu} | \boldsymbol{X})$, using the mean field approximation

$$\mathrm{q}(\boldsymbol{W}, \boldsymbol{Z}, \boldsymbol{\mu}) = \prod_{i=1}^{p} \mathrm{q}(\boldsymbol{w}_i) \prod_{j=1}^{n} \mathrm{q}(\boldsymbol{z}_j) \prod_{i=1}^{p} \mathrm{q}(\mu_i), \tag{3.86}$$

similarly to Bishop [15].

Optimal q^* distributions

The optimal distributions are

$$\mathrm{q}(\boldsymbol{w}_i) = \mathcal{N}\left(\overline{\boldsymbol{w}}_i, \boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right), \tag{3.87}$$

$$\mathrm{q}(\boldsymbol{z}_j) = \mathcal{N}\left(\overline{\boldsymbol{z}}_j, \boldsymbol{\Sigma}_{\boldsymbol{z}_j}\right), \tag{3.88}$$

$$\mathrm{q}(\mu_i) = \mathcal{N}\left(\overline{\mu}_i, \widetilde{\mu}_i\right), \tag{3.89}$$

where we note that since there are missing values, the covariance matrices $\boldsymbol{\Sigma}_{\boldsymbol{w}_i}$ and $\boldsymbol{\Sigma}_{\boldsymbol{z}_j}$ are now specific to the rows and columns of $\boldsymbol{W}$ and $\boldsymbol{Z}$, respectively.

Variational updates

The variational updates are as follows

$$\widetilde{\mu}_i = \frac{\nu_\mu \sigma^2}{|O_i|(\nu_\mu + \sigma^2/|O_i|)}, \tag{3.90}$$

$$\overline{\mu}_i = \frac{\nu_\mu}{|O_i|(\nu_\mu + \sigma^2/|O_i|)} \sum_{j \in O_i} \left(x_{ij} - \overline{\boldsymbol{w}}_i^\top \overline{\boldsymbol{z}}_j\right), \tag{3.91}$$

$$\boldsymbol{\Sigma}_{z_j} = \sigma^2 \left(\sigma^2 \boldsymbol{I}_{q\times q} + \sum_{i \in O_j} \left(\boldsymbol{\Sigma}_{\boldsymbol{w}_i} + \overline{\boldsymbol{w}}_i \overline{\boldsymbol{w}}_i^\top \right) \right)^{-1}, \tag{3.92}$$

$$\overline{\boldsymbol{z}}_j = \frac{1}{\sigma^2} \boldsymbol{\Sigma}_{z_j} \left(\sum_{i \in O_j} \overline{\boldsymbol{w}}_i (x_{ij} - \overline{\mu}_i) \right), \tag{3.93}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{w}_i} = \sigma^2 \left(\sigma^2 \mathrm{diag}\left(\nu_{w,g}^{-1}\right) + \sum_{j \in O_j} \left(\boldsymbol{\Sigma}_{z_j} + \overline{\boldsymbol{z}}_j \overline{\boldsymbol{z}}_j^\top \right) \right)^{-1}, \tag{3.94}$$

$$\overline{\boldsymbol{w}}_i = \frac{1}{\sigma^2} \boldsymbol{\Sigma}_{\boldsymbol{w}_i} \left(\sum_{j \in O_i} \left(x_{ij} - \overline{\mu}_i^\top\right) \overline{\boldsymbol{z}}_j \right), \tag{3.95}$$

$$\sigma^2 = \frac{1}{np} \sum_{ij \in O} \left(x_{ij} - \overline{\boldsymbol{w}}_i^\top \overline{\boldsymbol{z}}_j - \overline{\mu}_i\right)^2 + \widetilde{\mu}_i + \overline{\boldsymbol{x}}_j^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i} \boldsymbol{z}_j + \overline{\boldsymbol{w}}_i^\top \boldsymbol{\Sigma}_{z_j} \boldsymbol{w}_i + \mathrm{Tr}\left[\boldsymbol{\Sigma}_{z_j} \boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right], \tag{3.96}$$

$$\nu_{w,g} = \frac{1}{p} \sum_{i=1}^{p} \overline{w}_{ig}^2 + \widetilde{w}_{ig}, \tag{3.97}$$

$$\nu_\mu = \frac{1}{p} \sum_{i=1}^{p} \widetilde{\mu}_i + \overline{\mu}_i^2, \tag{3.98}$$

in agreement with Ilin and Raiko [60], where $\mathrm{diag}\left(\nu_{w,g}^{-1}\right)$ indicates the $(q \times q)$ diagonal matrix whose non-zero entries correspond to each $\nu_{w,g}^{-1}$ and $\widetilde{w}_{ig}$ is the g-th diagonal element of $\boldsymbol{\Sigma}_{\boldsymbol{w}_i}$.

Implementation

The variational algorithm then proceeds by cycling through the updates in Equations (3.90), (3.91), (3.92), (3.93), (3.94), (3.95), (3.96), (3.97), and (3.98) until convergence, which can again be assessed by inspecting the variational lower bound. Note that the missing values are not imputed in this algorithm and so are computed after convergence. This algorithm may also be constructed to treat σ^2, $\nu_{w,g}$, ν_μ as random variables with their own prior distributions.

3.5 Selection of the latent dimensionality

A key challenge is the choice of q, the dimension of the manifold onto which the original data points are projected (i.e. the number of principal components). Computationally, lower values of q require fewer parameters to be estimated and smaller matrices to be manipulated, leading to faster estimation. However, selecting q to be too small leads to under-fitting. It requires diligent selection, particularly when it is desirable to interpret the underlying latent factors.

In the VB approach, the choice of q is aided by the ARD prior distribution. Instead of directly selecting the dimensionality, q can be set to a large value (or even its maximum) value and the so-called *effective* dimensionality is inferred through the amount of shrinkage on each $\boldsymbol{w}_g$. Small values of $\nu_{w,g}$ result in values of $\|\overline{\boldsymbol{w}}_g\|$ close to zero and effectively eliminate the influence of that dimension.

For the EM algorithms described in Section 3.3.2, no such automatic selection is performed within the model. Instead, cross-validation is often used to maximise some selection criteria, as in Stacklies et al. [98]. This greatly increases the computational time required to run PPCA, since it must be run for multiple of values of q before an optimal value is selected. In our situation, we are primarily interested in high-dimensional datasets. Generally, this means that the statistic used for selection will be unstable due to its high sampling error. Multiple-fold cross-validation, which is often used to improve this stability, is also unlikely to be beneficial due to the high-dimensional nature of the datasets.

For covariance estimation, the choice of q is less crucial since it need not be justified by interpretation. Rather than focussing on the latent factors as with classical PPCA, the covariance estimation perspective is primarily concerned with low estimation error. The ARD prior using $q = p - 1$ in the Bayesian approach therefore seems more attractive than the non-Bayesian approach for this purpose.

3.6 Inverse covariance estimation

Recall that the inverse covariance matrix $\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$ has an important application in network reconstruction (Section 1.5). Network reconstruction aims to identify interactions between pairs of variables under study. Significant interactions are then visualised as edges that connect the variables, represented as nodes, in a network or graph [109]. The edges that comprise the graph can then be analysed to explore the interactions between variables in the dataset. Network reconstruction covers a broad range of statistical techniques and will be treated with more detail in Chapter 4.

A great advantage of the PPCA covariance model from Equation (3.2) is that $\boldsymbol{\Omega}$ may be calculated efficiently provided $q \ll p$. The calculation of inverse of a PPCA

covariance matrix may be simplified via the Woodbury matrix identity to obtain

$$\boldsymbol{\Omega} = \sigma^{-2}(\boldsymbol{I}_{p\times p} - \boldsymbol{W}\boldsymbol{M}^{-1}\boldsymbol{W}^{\top}), \tag{3.99}$$

where $\boldsymbol{M} = \sigma^2\boldsymbol{I}_{q\times q} + \boldsymbol{W}^{\top}\boldsymbol{W}$ is a $q \times q$ matrix. The complexity of this computation is therefore reduced from $O(p^3)$ to $O(q^3)$, which is extremely beneficial whenever $p \gg q$.

3.7 Model-based simulation

In this section, we study the performance of the PPCA algorithms as covariance estimators using simulated data. Similar to the previous chapter, we generate $M = 100$ datasets of size $n \in \{p/4, p/2, 3p/4\}$ from the generative PPCA model using $p \in \{60, 80, 100\}$, $q = 3$, $\boldsymbol{\mu} = \mathbf{0}$, and $\sigma^2 = 0.25$. Recall that by construction, this means that $\boldsymbol{z}_j \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{3\times 3})$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, 0.25\boldsymbol{I}_{3\times 3})$. In addition, we also generate $\boldsymbol{w}_g \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{3\times 3})$, $g = 1,2,3$. This means that there are M covariance matrices $\boldsymbol{\Sigma}$ that we attempt to estimate.

We compare the algorithms in Sections 3.3 and 3.4 except for VB algorithm 1 (since it is not yet implemented), denoting them by their corresponding function names in `pcaNet`. EM algorithm 1 is `ppcapM`, EM algorithm 2 is `pca_full` and VB algorithm 2 is `bpca_full`. For fair comparison, we run all algorithms with the correct $q = 3$ and set the maximum number of iterations to 1000. We report the familiar PRIAL metric for performance comparison, using the sample covariance matrix as the underlying baseline for improvement. Figure 3.1 presents the PRIAL and run-time in seconds of each algorithm for $p = 100$, with $p = 60, 80$ found in Appendix B.7.

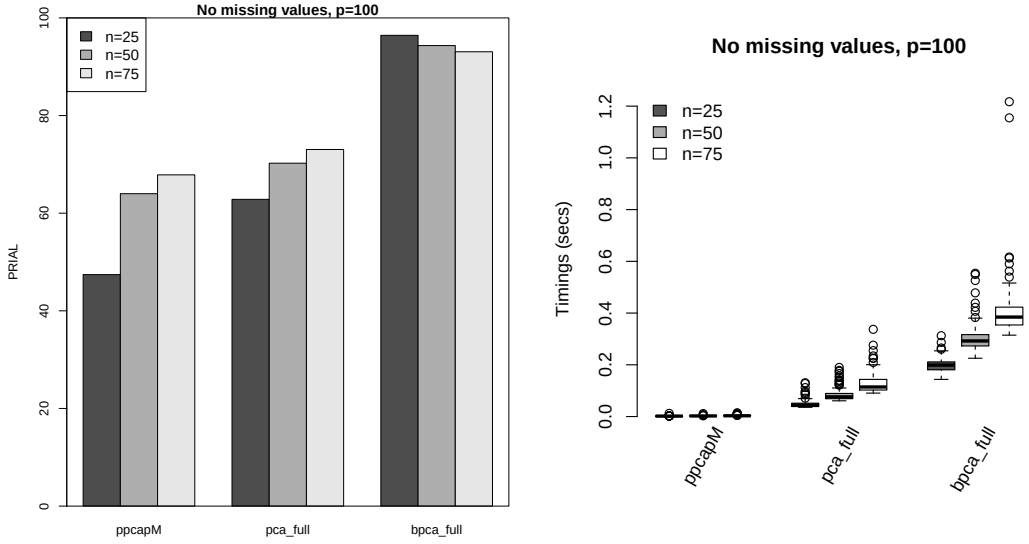


Fig. 3.1 PRIAL and timing of each PPCA algorithm for $p = 100$ with no missing values.

In terms of PRIAL, all algorithms deliver a significant performance boost over the sample covariance estimator. This is a good sanity check given that they represent the data-generating model. It is also clear that the VB algorithm outperforms the EM implementations for all values of n. The best performing algorithm is `bpca_full`. Its EM counterpart `pca_full` also outperforms `ppcapM`. However, this increased performance comes at a price in terms of computational cost, as can be seen from the timings plot of Figure 3.1. The VB algorithm does remarkably well for the very small sample size $n = 25$, attaining a PRIAL close to 100 and therefore almost recovering the true covariance matrix. Interestingly, the performance of the VB algorithm lowers as n increases, whilst the performance of the EM algorithms increases with n.

We now wish to see how the performance of each algorithm changes in the presence of missing values. We repeat the previous simulation two times, randomly excluding 30% and 50% of the values from the new datasets. With the now excluded missing values, we cannot use the PRIAL as a performance metric with the sample covariance matrix as a baseline, because it no longer exists. Instead, we report the squared Frobenius losses for each algorithm, which we recall is the underlying metric for the PRIAL. The timing results follow a similar pattern to those in the previous simulation, and so are left to Appendix B.7 along with the results for $p = 60, 80$. The results for $p = 100$ are presented in Figure 3.2.

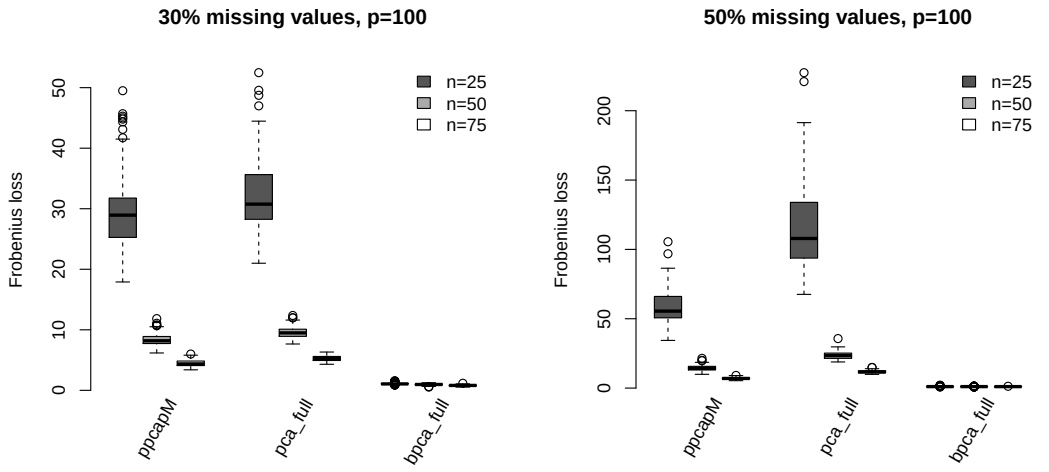


Fig. 3.2 PRIAL for the PPCA algorithms with $p = 100$ with 30% and 50% missing values.

Again we can see that the VB algorithm outperforms the EM implementations. The VB algorithm attains nearly zero Frobenius loss despite having 30% and 50% of the data removed, which is a very strong performance. For 30% missing values both EM algorithms now perform similarly. However, for 50% missing values, `ppcapM` now outperforms `pca_full`. This suggests that the performance of `pca_full` diminishes

as the percentage of missing values increases. This is somewhat surprising, since the algorithm was derived with the intention of handling missing values in a principled way through its update equations.

Overall, these results suggest that the VB algorithms perform better in situations where the data-generating model is of the same form as the PPCA model with a known number of latent dimensions with and without the presence of missing values. The `bpca_full` algorithm has the highest performance but does incur a much higher computational cost. This cost is not noticeable in these simulations since on average each run took less than one second to complete. But for situations with a large number of variables and larger latent dimensions, the smaller run-time of the EM algorithms might be preferable.

3.8 Comparison to TAS

In this section, we provide a numerical comparison between PPCA and TAS. We include the best performing PPCA algorithm from Section 3.7 to some of the TAS simulations from Chapter 2.

3.8.1 Model-based simulation

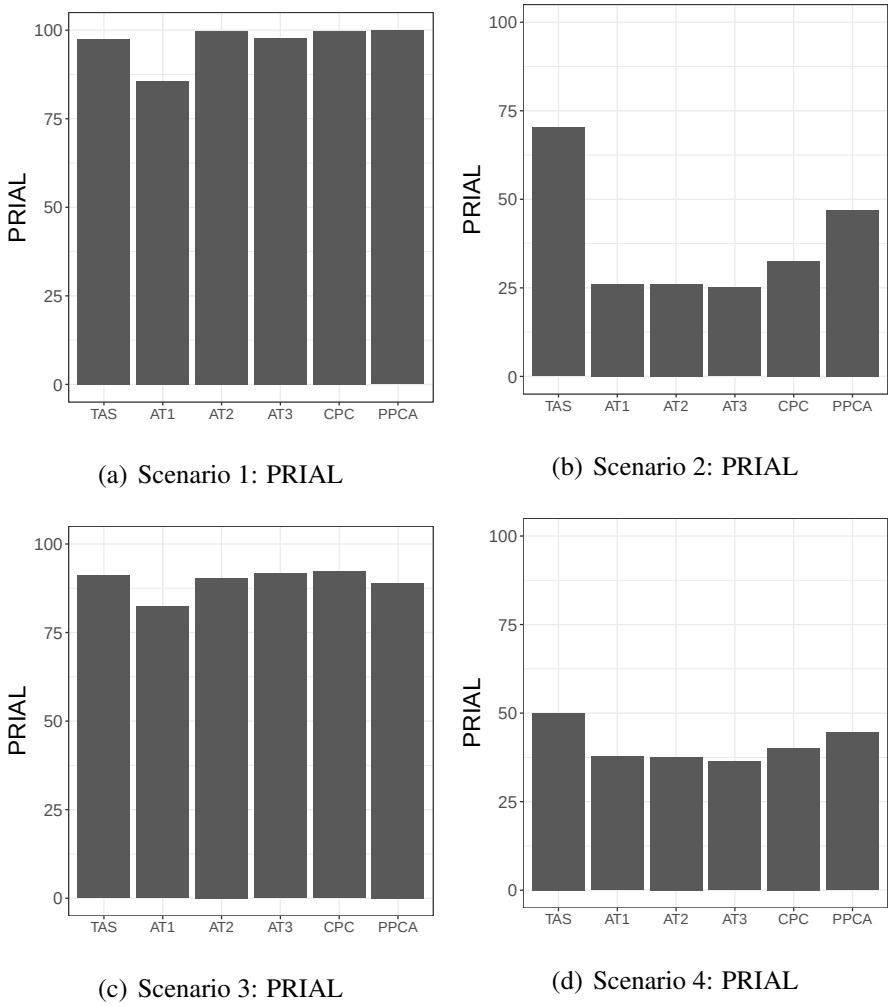
Recall the model-based simulation from Chapter 2. To recap, we generate $M = 100$ datasets of size $n \in \{25,50,75\}$ from a p-variate Gaussian distribution with zero mean vector and covariance matrix Σ, where $p = 100$. Four distinct covariance structures are considered, yielding the following four simulation scenarios:

- **Scenario 1: common variance, zero correlation.** $\boldsymbol{\Sigma}_1 = 5 \times \boldsymbol{I}_{p\times p}$,
- **Scenario 2: unit variance, constant correlation.** $\boldsymbol{\Sigma}_2 = \boldsymbol{I}_{p\times p} + 0.3 \times (\mathbf{1}_{p\times p} - \boldsymbol{I}_{p\times p})$, where $\mathbf{1}_{q\times r}$ is the $q \times r$ unit matrix with elements all equal to one.
- **Scenario 3: unequal variances, decaying correlations.** $\boldsymbol{\Sigma}_3 = \boldsymbol{D}^{1/2}\boldsymbol{C}\boldsymbol{D}^{1/2}$, where the $(i,j)^{\text{th}}$ entry of $\boldsymbol{C}$ equals $(-0.7)^{|i-j|}$ and $\boldsymbol{D} = \text{diag}(d_1,\ldots,d_p)$ with $d_i \sim \mathcal{U}(1,5)$.
- **Scenario 4: unit variance, block-diagonal correlation.** $\boldsymbol{\Sigma}_4 \sim$ Inv-Wishart, such that $\mathbb{E}[\boldsymbol{\Sigma}_4] \propto \boldsymbol{B}$, where $\boldsymbol{B}$ is a block-diagonal matrix with two identical $p/2 \times p/2$ blocks, each with the same constant correlation structure that was used in scenario 2.

We do not include the Bayesian single-target shrinkage estimators in the results as their performance was reported in Chapter 2. We apply PPCA using the `bpca_full` algorithm with $q = p - 1$, utilising the ARD prior for latent dimension shrinkage. This

algorithm was selected over the EM approaches due to its selection of q, and over `bpcapM` due to its superior performance in Section 3.7. We denote the estimator simply as 'PPCA' in this section's figures since it is the only PPCA candidate.

Figure 3.3 summarises the results obtained for $n = 25$, the results for $n \in \{50,75\}$, which are similar to that of $n = 25$, are provided in Appendix B.8. In general the PPCA



(a) Scenario 1: PRIAL

(b) Scenario 2: PRIAL

(c) Scenario 3: PRIAL

(d) Scenario 4: PRIAL

Fig. 3.3 PRIAL for $p = 100$ and $n = 25$ for scenarios 1, 2, 3, and 4..

estimator performs similarly to the best single target shrinkage estimators. Only on one occasion, scenario 1, does PPCA improve upon the performance of TAS for $n = 25$. In Appendix B, more extreme results can be seen – PPCA performs better than all alternatives in scenario 4 for $n = 50,75$. This performance suggests that PPCA (at least using `bpca_full`) is not an ideal candidate for a standalone covariance estimator when its model assumptions do not hold and there are no missing values. However, it could provide some utility as a target within the target set of TAS, which we expand upon in Section 3.10.

3.8.2 Predictive validation simulation

Recall the predictive validation simulation from Chapter 2 using the following TCGA datasets:

- **Data set 1:** p53 pathway in breast cancer ($p = 68$ genes in $N = 529$ samples)
- **Data set 2:** apoptosis pathway in ovarian cancer ($p = 86$ genes in $N = 558$ samples).

For each data set, we randomly split the full data matrix ($p \times N$) into a small sample size ($p \times n$) and a large sample size ($p \times (N-n)$) data matrix, for $n \in \{p/4, p/2, 3p/4\}$. The sample covariance matrix of the large sample size matrix is used to proxy for the underlying covariance matrix, whilst the high-dimensional estimators are applied to the high-dimensional partition. This procedure is repeated 1,000 times for data sets 1 and 2, and for the three different values of n investigated. We apply PPCA again using `bpca_full` with $q = p - 1$ to utilise the ARD prior. The results for this simulation can be found in Figure 3.4.

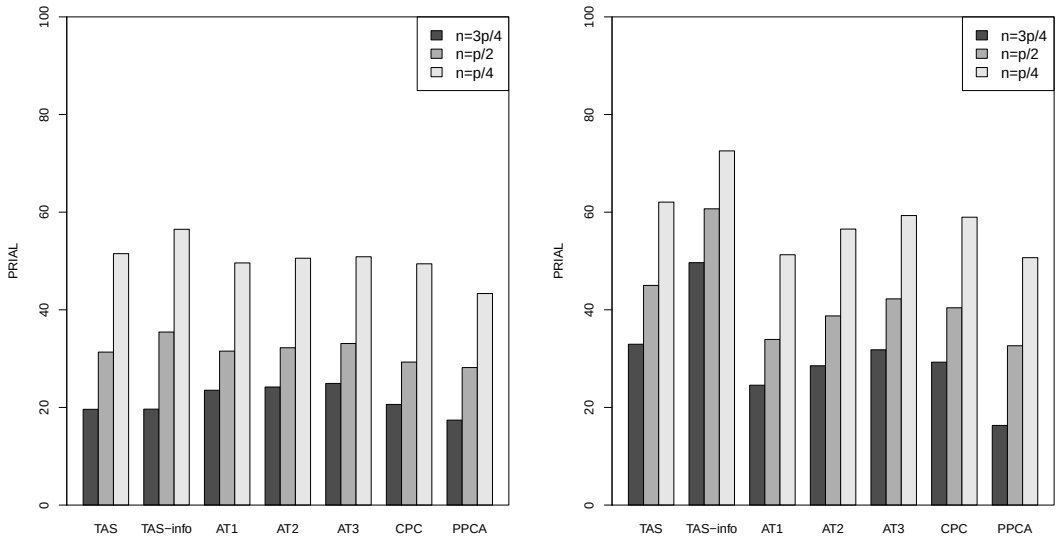


Fig. 3.4 PRIAL for the predictive validation BRCA and OV simulations.

In these simulations, PPCA fails to really achieve the same level of performance as the other estimators across all ratios of n and p. This provides further evidence that PPCA (with its `bpca_full` implementation) is simply not an effective covariance estimator by itself when the PPCA model assumption is not necessarily true and there are no missing values.

3.9 Software

We provide an open source `R` package `pcaNet` http://github.com/HGray384/pcaNet, which provides an interface to `C++` code that efficiently implements the PPCA model for EM algorithm 1 and 2 described in Section 3.3 and VB algorithm 2 from Section 3.4. Our software provides a means for high-dimensional covariance matrix estimation from incomplete data, and interfaces with existing `R` packages in order to perform network inference. Figure 3.5 provides a graphical illustration of the software.

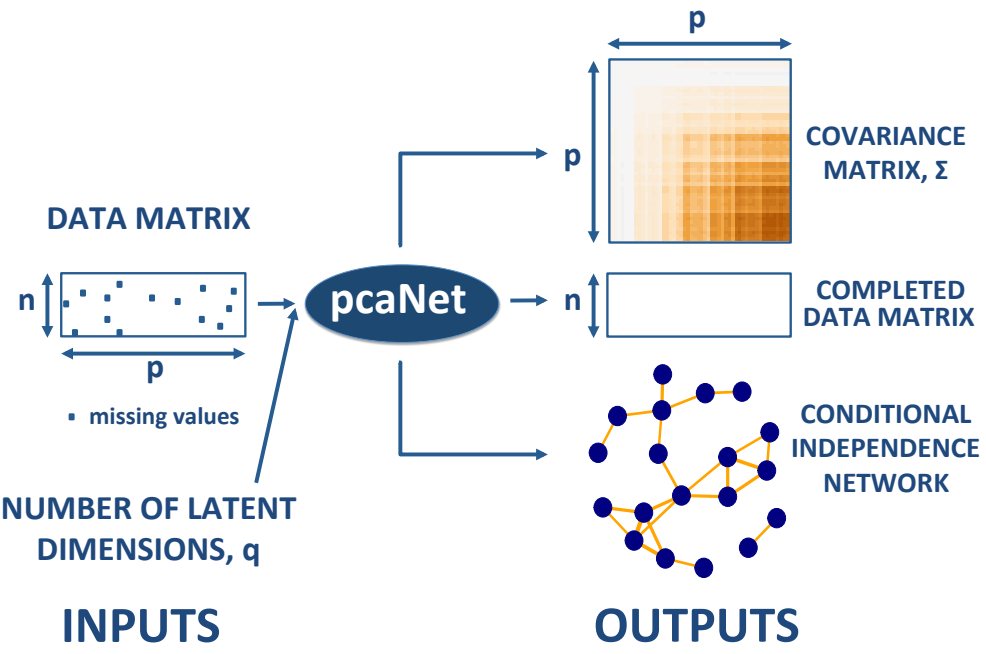


Fig. 3.5 A graphical illustration of pcaNet

In `pcaNet`, we implement accelerated versions of existing PPCA model fitting algorithms from the `pcaMethods` `R` package [98] (EM 1; demonstrated in Appendix B.6) and port the equivalent functions from the `PCAMV` `MATLAB` toolbox [60] to `R` (EM and VB algorithm 2). As our focus is the covariance estimator, `pcaNet` is designed to report the inferred covariance matrix and not just the dimensionality reduction of PPCA.

Here we demonstrate the output of `pcaNet` when applied to the Arabidopsis thaliana dataset provided in `pcaMethods` [98]. The data consist of 154 observations of 54 metabolites during a cold stress experiment, with 5% of values uniformly removed for the primary purpose of imputation assessment. Here we do not compare missing value imputation accuracy, only demonstrating how the software may be used. We choose to use EM algorithm 1 for our demonstration, referred to as `ppcapM` which is its function name in `pcaNet`.

Applying ppcapM with $q = 5$ to the dataset first yields Figure 3.6, which represents the entries of the matrix $\boldsymbol{W}$. This display is known as a Hinton diagram. Hinton diagrams

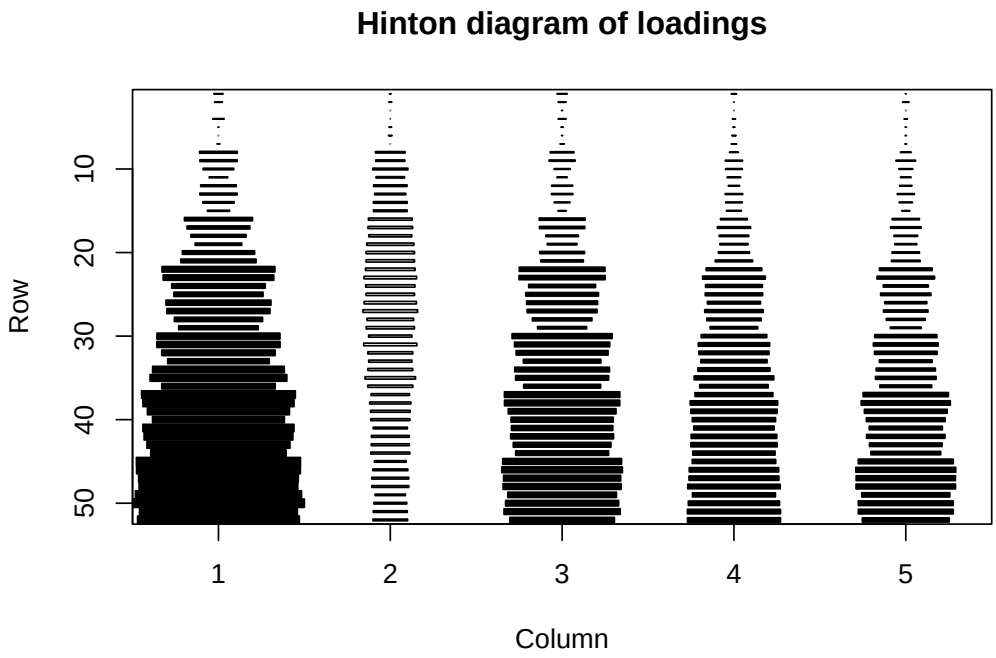


Fig. 3.6 The Hinton diagram generated by ppcapM when applied to the Arabidopsis thaliana dataset with $q = 5$.

can be useful for inspecting the form of the estimated $\boldsymbol{W}$ to see the transformation of the observed and latent variables. It can also be helpful to see the effect of an ARD prior in the VB setting.

From the output of ppcapM, it is possible to access a number of the estimated model parameters as well as the associated log-likelihood values, which can be useful for model comparison. Most notably, the estimated covariance matrix is available to use for further analysis. It can be visualised as a heat map using the ppca2Covplot function. Figure 3.7 shows the heat map for the estimated covariance matrix of the Arabidopsis thaliana dataset using ppcapM with $q = 5$.

Since the objective of pcaNet is to directly estimate and use the covariance matrix of the data, further principal component analysis functions are not provided in this package. However, if that functionality is also desirable to the user, then the output of pcaNet can be easily integrated with the PCA tools provided in the pcaMethods package [98] by accessing the pcaRes object of the output.

Since the second aim of pcaNet is to use the covariance matrix estimate to perform network reconstruction, we also provide this functionality. The function ppca2Net takes the output of any pcaNet PPCA function and returns a network. Within this process, the inverse covariance matrix is computed, partial correlations are extracted, partial correlations are tested for significance, and then significant partial correlations are added

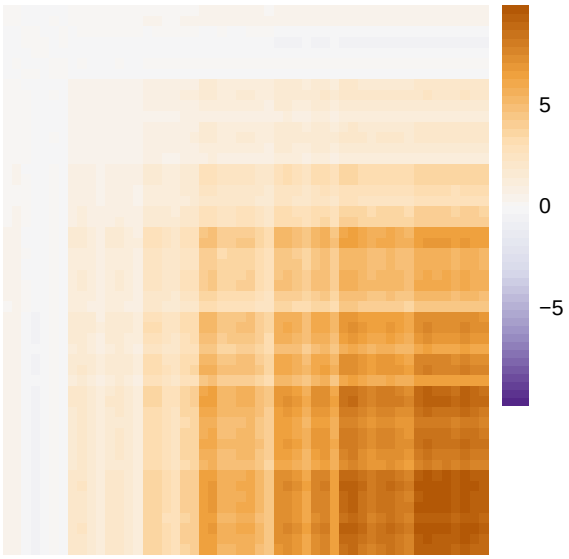


Fig. 3.7 The covariance heat map generated by `ppcapM` when applied to the Arabidopsis thaliana dataset with $q = 5$

as weighted edges to a conditional independence graph, which can be visualised by specifying an optional argument. To determine whether or not each partial correlation is significantly non-zero, `pcaNet` uses the empirical Bayes mixture model approach [37, 95] as supplied in the `fdrtool` `R` package [100]. For visualisation of the resulting network, `pcaNet` uses `igraph` [29] to display significant conditional dependencies, represented as edges in a network whose nodes are the observed variables. Note that for a large number of significant edges, it might not be useful to plot the resulting graph. Figure 3.8 shows the network the Arabidopsis thaliana dataset using `ppcapM` with $q = 5$.

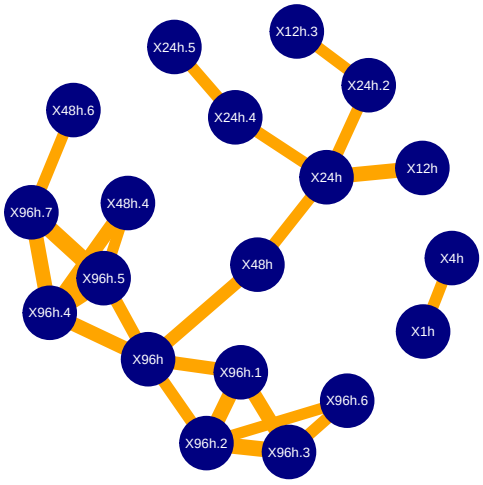


Fig. 3.8 The network plot generated by `ppcapM` when applied to the Arabidopsis thaliana dataset with $q = 5$.

`pcaNet` is designed to be easy to use for those who are new to PPCA and network reconstruction, whilst still providing sufficient functionality for those with experience (including integration with other well-established packages). For this reason the default values are used in the underlying functions from other packages with minimal user control, but the statistics for the graphical output of `ppca2Net` are also provided. This means that for those who wish to perform a different visualisation or analysis using `igraph`, we provide the estimated network as an `igraph` object. In addition, for those who would like to further investigate the partial correlation statistics (perhaps performing their own thresholding for significance), we also provide the underlying statistical output from the `fdrtools` package.

3.10 Discussion

In this Chapter, we have considered PPCA as a method for covariance estimation. We have described the EM and VB approaches for parameter estimation and how they can handle missing values. We also extended an existing algorithm to handle data situations with missing values. This, in addition to the full derivations with consistent notation and software application, represents a novel contribution to the literature. We discussed the problem of the latent dimension q and how the ARD prior in VB implementations offers an attractive solution, particularly for covariance estimation when the latent dimensions do not require interpretation. By assuming a low-dimensional latent space, a large benefit of the PPCA approach for covariance estimation is that the inverse covariance (i.e. the precision matrix) can be calculated efficiently. This presents a very appealing opportunity for multivariate methods that require the inverse covariance matrices of a large number of variables, such as network reconstruction based on partial correlations.

This chapter illustrated our `R` package `pcaNet`, which implements three of the algorithms introduced here. `pcaNet` provides an easy interface to perform covariance estimation and network reconstruction using PPCA. It also interfaces with other popular software packages such as `pcaMethods` for further analysis of the estimated parameters, `fdrtool` for evaluating statistical significance for the inferred associations, and `igraph` graphically displaying the estimated networks.

Section 3.7 provided a small comparison between the three PPCA methods implemented in `pcaNet`. We focussed on the situation where the data-generating mechanism was a PPCA model and the number of latent dimensions was known. In this case, `bpca_full` provided the best PRIAL values and Frobenius losses, although coming at an additional computational cost that might become obstructive for large datasets. However, due to their row- (column-)wise nature, both `ppca_full` and `bpca_full` using Equations (3.9), (3.18), (3.10), (3.91), (3.93), and (3.95) can easily be parallelised

in their implementation, which would reduce computational time (especially if p is large). This parallelisation is left as future work.

This model-based comparison is limited in its application to real data since number of latent dimensions is mostly not known. A more realistic comparison would be to estimate the number of latent dimensions with each method as it would be in practice; using cross validation for frequentist methods and the ARD shrinkage for the Bayesian ones.

Sections 3.8 provided a small comparison between `bpca_full` and the state-of-the-art shrinkage methods introduced in Chapter 2. Whilst providing some evidence for improvement in certain scenarios, the PPCA estimator was mostly bested by the other methods. A more comprehensive comparison will be required to fully assess the merits of each method, particularly in the presence of missing observations. PPCA has a conceptual advantage in such case, as it does not require the use of imputation techniques prior to covariance estimation. We expect that `pcaNet` will enable this comparison.

Despite these results, PPCA does still have some potential utility as a target matrix to be used within TAS, due to its good performance in certain scenarios. This would enable the target set to capture the low-dimensional structure defined by PPCA. Since, TAS is robust to misspecified target matrices, it would also overcome PPCA's otherwise mediocre performance. The main challenge would be how to construct a PPCA covariance estimator without using the same dataset twice and overfitting. One initial thought would be to employ a partition scheme, in which the PPCA model can be estimated on smaller subsets of the data. This would seem promising since the VB implementations of the model seem to perform well in high-dimensions. However, if one is already concerned with a high-dimensional dataset then further subsetting is not desirable. Another idea would be to utilise external datasets, in line with what we did in Section 2.8. In such cases, PPCA could be applied to external datasets in order to define target matrices to be used within TAS.

Chapter 4

Case study: cytokine expression in the context of traumatic brain injury

4.1 Introduction

Traumatic brain injury (TBI) is a widespread instigator of death and disability globally. Cytokines [49], which are signalling proteins involved in the immune system, are able to mediate injury following TBI by acting as anti-inflammatories in some cases. They therefore provide an interesting molecular study in the aftermath of TBI and as targets for injury treatment.

The interleukin-1 (IL1) receptor is a cytokine receptor that binds IL1, which plays a central role in the regulation of immune and inflammatory responses. Interleukin-1 receptor antagonist (IL1ra) is a cytokine that opposes the effects of IL1 by binding, and therefore limiting access, to the IL1 receptor. The drug recombinant human IL1ra is a licensed treatment for rheumatoid arthritis, has a well-defined safety profile, and has previously been trialled in stroke [38], subarachnoid haemorrhage [47, 25], and severe sepsis [83].

Antagonism at the interleukin-1 receptor has been reported to exhibit protective properties in rodent studies of brain injury [26, 27, 101, 67, 89, 10, 66]. However, little is known about how the drug penetrates the blood-brain barrier, a highly selective semipermeable membrane that separates the circulating blood from the brain and extracellular fluid in the human central nervous system.

Helmy et al. [55] provided the first-of-its-kind randomised clinical trial for IL1ra in patients with TBI, monitoring not only cytokine levels in the blood, but also in the brain to determine its efficacy at reaching the injured tissue. The overarching goal of the study of Helmy et al. [55] was to provide a comprehensive biochemical assessment of the treatment effect associated to the drug IL1ra in TBI patients. The data that we analyse in this chapter is a subset of this trial.

Applying methodology from a similar previous dataset [53], Helmy et al. [55, 56] used principal component analysis and partial least squares (PLS) discriminant analysis to group cytokine profiles based on the time after trauma, treatment status, and sampling type (brain versus blood). Due to a high number of missing values, data from multiple time points was pooled in order to reduce their burden. In addition, a liberal threshold of 50% missing values was used to exclude cytokines from the resulting statistical analysis. Remaining missing values were automatically imputed in the PLS method but were not discussed beyond that.

In this chapter, we seek to analyse the cytokine expression across each individual time point in order to gain a more accurate insight into the behaviour of cytokines as a response to TBI across time. To support this, we use a more conservative missing value threshold. We also perform a missing value imputation for each individual time point and therefore avoid the need to pool data and average out potentially interesting expression information. We demonstrate how high-dimensional covariance estimation can be useful for characterising the interactions between cytokines through cluster analysis and network reconstruction.

The chapter is organised as follows. In Section 4.2 we describe the dataset from Helmy et al. [55] that is used to conduct the statistical analysis. Section 4.3 then details a short exploratory analysis of the missing values and univariate statistics present for each cytokine, treatment status, and sample type. In Section 4.4 we describe the covariance estimation procedure using TAS and how the resulting matrix is used for the multivariate analysis technique cluster analysis and network reconstruction, we then present the results. Finally, we conclude the chapter with a discussion in Section 4.5.

4.2 Materials and methods

In this section we describe the aims, methods, and study design of the randomised control trial as outlined in Helmy et al. [55], to which we also refer the reader for further information.

4.2.1 Recruitment and treatment allocation

Strict criteria of eligibility for and exclusion from the study were imposed [55]. Recruitment resulted in a total of 20 patients with severe TBI, all of whom were recruited within 24 hours of their injury. Ten patients were allocated to the treatment arm of the study and the remaining ten patients were allocated to the control arm of the study. Allocation was decided by the randomisation of sealed envelopes whose contents contained the treatment status. The resulting treatment status was revealed to both the physician and the family of the patient. The treatment group received 100mg of IL1ra (drug named

Anakinra; Kineret) subcutaneously (injection under the skin) once per day. The control group received neither the drug nor a placebo.

4.2.2 Intervention and sampling

Upon admission, the time of trauma was recorded as well as measures of brain injury, disease, trauma severity and images of brain computerised tomography scan. After admission, patients were monitored over a period of five days. At the same time each day, the treatment was administered to the treatment group. The cytokines' expression was measured before and after treatment using three sampling modes: arterial blood, venous blood, and microdialysis fluid (microdialysate) obtained from microdialysis catheters [54] near injured areas in the brain.

One hour before and after treatment administration (or hypothetical administration in the case of the control group) a number of clinical variables and fluid samples were taken. Microdialysis vials were collected and replaced from the catheters every hour during this five day period, and so the number of microdialysis samples is greater than the number of blood samples.

We refer to Helmy et al. [55] for more details on how samples were obtained, treated and profiled. Importantly for our ensuing analysis, blood samples were taken at a sufficient volume for cytokine profiling. This means that for arterial and venous samples there were fewer samples than the microdialysate, but no pooling or dilution of the samples was required. However, the microdialysate inherently has a low volume of extraction. For this reason, the microdialysis samples required pooling into 6-hour time periods in order to achieve the necessary volume for cytokine profiling. Due to this pooling, the 5-day monitoring period can be seen as 20 microdialysis sampling time points. For fair comparison, in our analysis we only use the time points corresponding to the sampling times for the arterial and venous samples, which corresponds to time points 1, 2, 5, 6, 9, 10, 13, 14, 17, and 18. We refer to these microdialysis sampling times when mentioning specific time points.

4.2.3 Data

The samples were assayed using premixed analyte kit targeting 42 cytokines and read on a Luminex system as described in [53, 55], with resulting concentrations calculated with reference to a standardising logistic curve for each cytokine. A full list of the cytokines considered can be found in Appendix Table C.1. The blood and brain samples were run on separate plates to ensure that the control measurements were calibrated specifically for sample type. The resulting measurements of the cytokine expression are continuous values representing the abundance of each cytokine in the sample. Due to

the detection limits and random error, there are also a number of missing values, which we explore in the next section.

4.3 Statistical analysis

Our analysis focuses on the cytokine expression data generated by the above trial, comparing between trial arms (treatment and control) as well as across sampling times and types (e.g. arterial).

4.3.1 Exploratory analysis

The focus here is to explore the characteristics of the data before performing statistical inference. When inspecting the cytokine expression values, the most prominent characteristic is that the dataset contains a substantial amount of missing values.

For some cytokines, there were a very large amount of missing values. Such cytokines are problematic as most statistical models do not allow for missing data. In such cases, one could impute the missing values but this could substantially alter the structure of the data if there are many. For this reason, we consider setting a missing value threshold to exclude cytokines that have a high percentage of missing values from our analysis. There is a clear trade-off to be made when selecting a missing value threshold in order to retain as many cytokines with observed values as possible whilst excluding those with lots of missing values. We opt to consider a conservative threshold of 20%.

Although arbitrary, some justification for this threshold is given by inspecting the relationship between the average log-expression of the cytokines and their percentage of missing values, shown by Figure 4.1. It can be seen that lowly expressed cytokines give rise to more missing values. This is likely because of cytokines expressing themselves close to or below the limit of detection of the assay. Using a threshold of 20% not only removes cytokines with many missing values, but it also artificially removes the relationship between missing value percentage and mean expression - with the remaining missing values more evenly distributed along this domain. There are also a number of missing values for highly expressed cytokines that do not appear to be due to the detection limit. These have no known cause and we assume that they are missing at random. We also note that some patients have a higher amount of missing data. For example, there are no cytokine expression values recorded for patient 11 at time points 13, 14, 17, and 18, and patient 14 at time point 18, and so the number of samples for these time points is less than for the others.

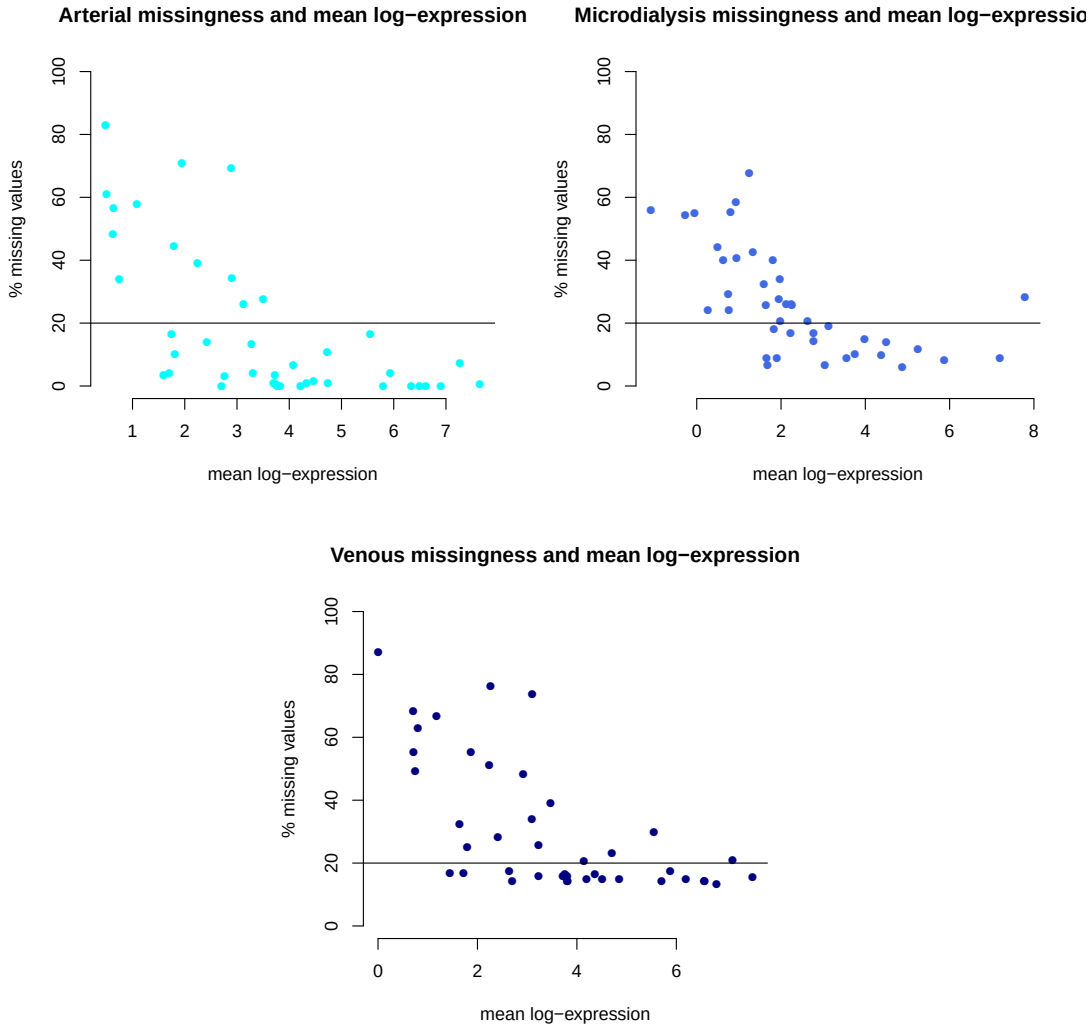


Fig. 4.1 Average log-expression of cytokines as a function of the percentage of missing values. We have log-transformed the non-missing expression values in order to reduce them to a smaller, more stable scale.

In addition to excluding cytokines with more than 20% missing values, we also decided to exclude all of the venous samples. This was done for three reasons (i) the main difference of interest was between blood and brain samples, rather than oxygenated and deoxygenated blood (ii) for biological reasons, the cytokines' expression in arterial samples are expected to be highly correlated with venous samples (iii) the majority of venous cytokine expressions had missing value percentage close to our threshold and is therefore generally considered to be of lower quality. The resulting analyses proceeded with 28 cytokines for the arterial samples and 17 cytokines for the microdialysis samples, with a total of 195 total observations (divided across 20 patients and 10 time points).

4.3.2 Univariate analysis

We explored the mean and variance for each of the remaining cytokines' expression. Figure 4.2 shows the interquartile ranges of expression levels of IL1ra in both arterial and microdialysis samples. As expected, there is a pronounced difference between its expression in the treatment and control group for the arterial samples, since the bloodstream is the primary location of administration for the cytokine. For the microdialysis samples the difference is far less apparent. The biological mechanism restricting this efficiency of expression is the blood-brain barrier. Nonetheless, this difference has been reported to be substantial enough to show that subcutaneous administration of IL1ra does lead to extra penetration of the blood-brain barrier [55].

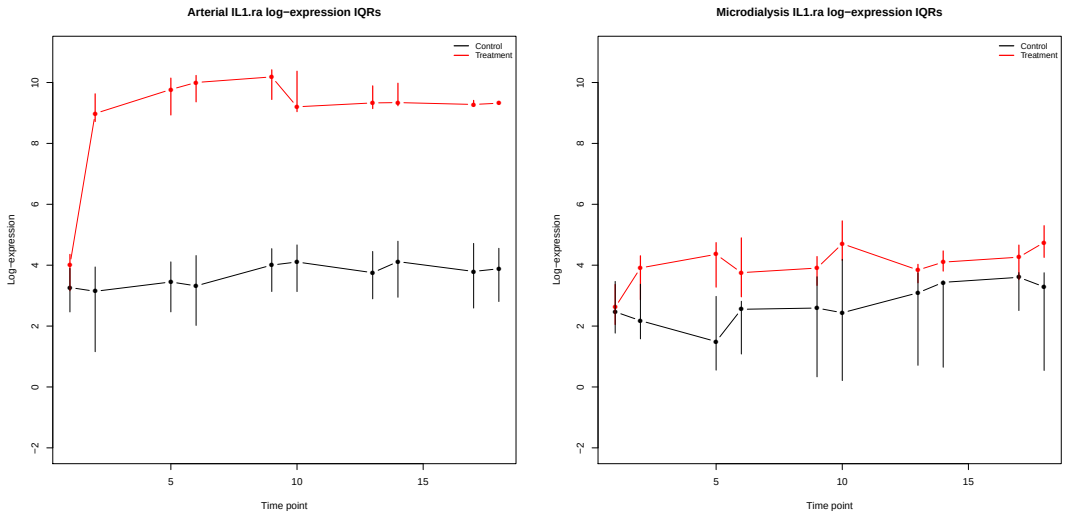


Fig. 4.2 The inter-quartile ranges of IL1ra for both arterial and microdialysis samples. The values of the control group are in black whilst those of the treatment group are in red.

Having explored the effects of IL1ra administration on the expression levels of IL1ra in blood plasma and cerebral microdialysate, we now explore the expression of the remaining cytokines to see if any other differences arise. As an illustration, Figure 4.3 shows the expression for two of the cytokines considered in the arterial samples, Eotaxin and Fractalkine. It can be seen that there are negligible differences for the median expression of these cytokines between the treatment and control groups. Any differences between the treatment and control groups are mostly covered within their inter-quartile range (IQR) bars. However, there does seem to be some differences between the size of the IQR bars for some cytokines, indicating that there is some variation. As illustrated in Figure 4.3 a similar behaviour can be observed in the microdialysis samples shown in Figure 4.4.

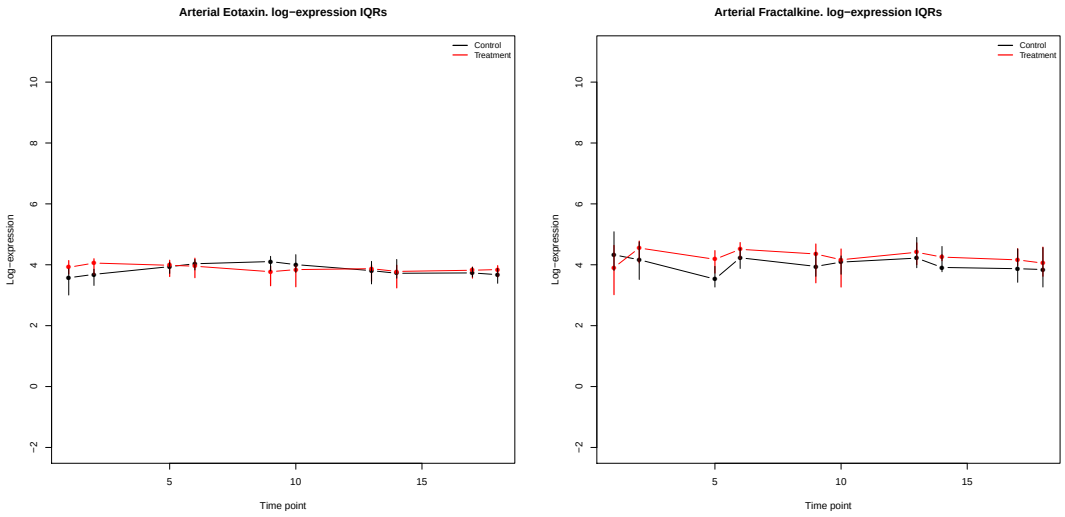


Fig. 4.3 The inter-quartile ranges for Eotaxin and Fractalkine in the arterial dataset. The values of the control group are in black whilst those of the treatment group are in red.

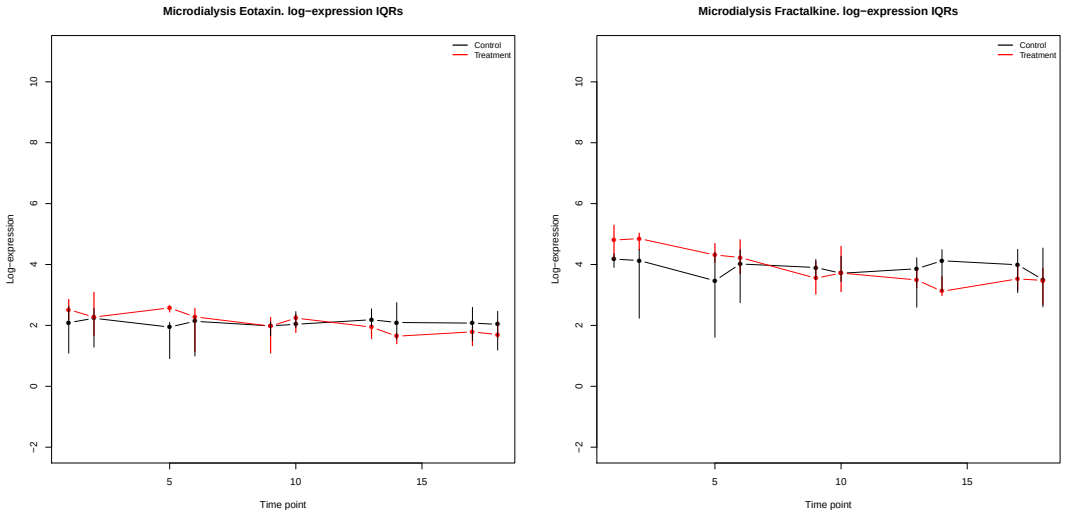


Fig. 4.4 The inter-quartile ranges for Eotaxin and Fractalkine in the microdialysis dataset. The values of the control group are in black whilst those of the treatment group are in red.

Due to the marginal differences in the means of these cytokines, we next explored their variances. For each patient and time point, we calculated the variance of the observed expression values (log-scale) across all cytokines (excluding IL1ra). The median and IQR of these variances across patients can be found in Figure 4.5. We can see that for the arterial samples, the cytokines exhibit similar variances and inter-quartile ranges. For the microdialysis samples there is a large difference in variability between the treatment and control groups. This is particularly interesting because the difference is present as of time point 1, which is before any treatment is administered.

This suggests that this is not a cause of the treatment, but is rather a baseline difference between the two groups. It is also not present in the arterial samples, which would suggest that it is not caused by some fundamental difference between the patients in each group.

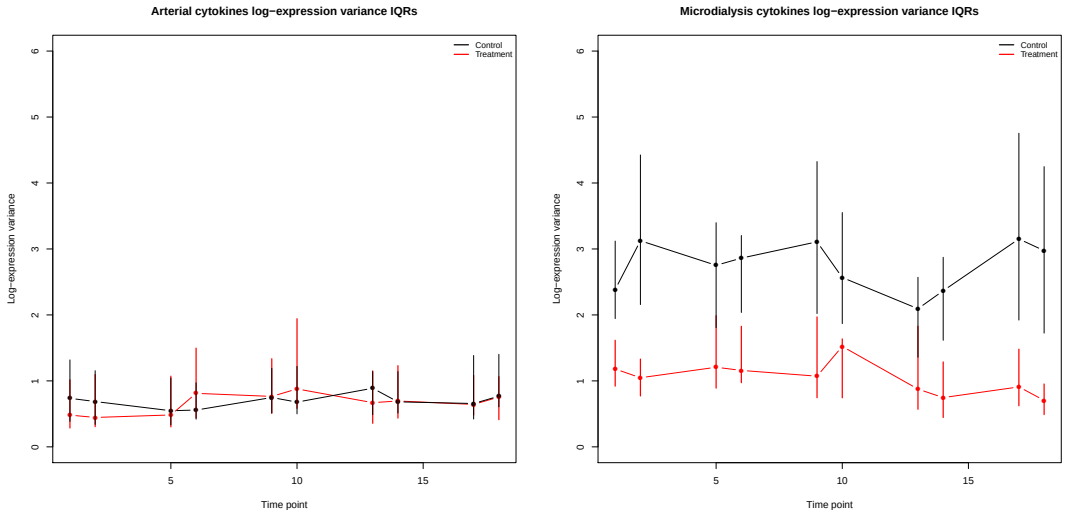


Fig. 4.5 Variance IQRs of the observed expression values (log-scale) across all cytokines in both the arterial and microdialysis dataset. The values of the control group are in black whilst those of the treatment group are in red.

This exploratory analysis further motivates a multivariate analysis of the cytokines, since there does not appear to be differences between the univariate expression levels of the cytokines, yet there does seem to be differences in variability across cytokines. Next, we explore potential differences in the associations between the cytokines through their covariance structure.

4.4 Results

Our aim is to obtain a covariance matrix estimate at each time point, and then extract the resulting marginal and partial correlations to investigate the cytokine interactions (see Figure 4.6). We then apply cluster analysis and network reconstruction (explained in the following sections) to find groups of cytokines that frequently interact, indicated through clusters or edges in the analyses. We summarise the resulting clusters and networks across all time points in order to categorise changes or stabilities over time for the treatment and control group in both arterial and microdialysis samples. Here we describe our methodology and present the results.

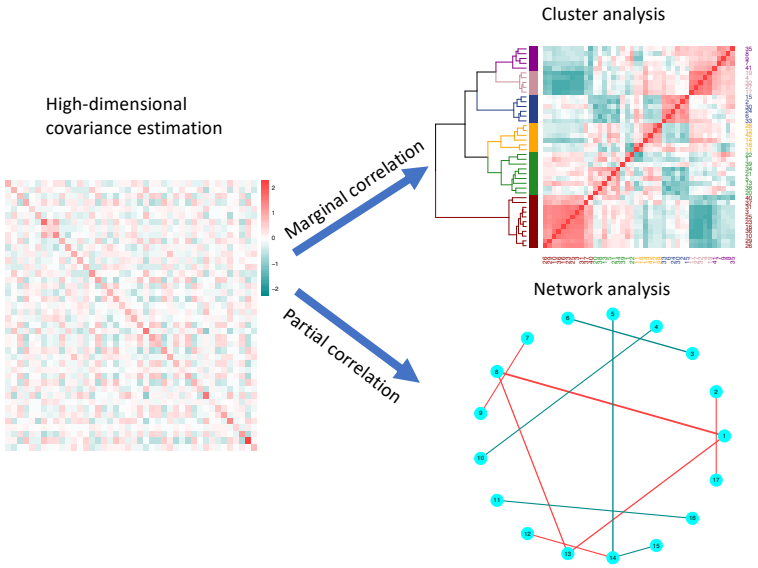


Fig. 4.6 The covariance estimate can be used to perform cluster analysis and network reconstruction by extracting the marginal and partial correlations, respectively.

Covariance estimation using TAS

Here, we explore the use of TAS in order to infer the covariance structure among cytokines, within treatment groups as well as across different time points. The main motivation for using TAS instead of PPCA is because of the sequential structure of the dataset. In this context, TAS enables the use of covariance estimates obtained for earlier time points as prior information (in the form of target matrices) when using TAS on later time points.

Before we can apply TAS, however, we recall that there are still a small number of missing values in the dataset. An important point to note is that some of the missing values are censored observations due to being below the limit of detection of the assay. However, as lowly expressed cytokines were removed due to our stringent threshold (>20% missingness), we assume that the remaining values are missing at random. We choose to impute these missing values independently for each time point using the well-established k-Nearest Neighbours (kNN) algorithm [104]. This algorithm computes the similarity between cytokines by computing the Euclidean distance of their (non-missing) log-expression values from each other across all patients in a chosen stratum. The closest k cytokines are identified and then the missing values are imputed as an average of their neighbours' log-expression values for that patient. We use kNN with a

small number of neighbours, $k = 3$. The rationale behind a small k is that the imputation of censored values will still be governed by similarly lowly expressed cytokines and nothing else, and therefore the imputation inaccuracy should be limited. The choice of $k = 3$ is arbitrary and as such we also present all analysis results for $k = 5,7$ (Appendix C.2) to assess the robustness of our analysis.

Having imputed the missing data for each time point, we centre the resulting log-expression values by subtracting their mean. Our strategy is to use TAS to its maximum potential, by including informative prior information. For this dataset, that manifests as using the covariance estimates from previous time points in the target set for the current time point. This is derived from our expectation that the covariance structure from previous time points should have relevant information to share with that of the current time point. Remaining loyal to the structure of the data, we treat the time points as if they were sequential, so that at time point 5, we use information from time points 1, 2, and 4, but not of the others since they have not yet occurred. We also treat the arterial and microdialysis samples completely separately, so that no covariance information is shared across sample types.

For the first time point there are no previous data-derived covariance matrices to use. In this case, we use the nine default targets from Section 2. For every time point thereafter, we construct the target set using these nine default targets as well as all TAS covariance estimates from previous time points. For example, for the estimation of the arterial cytokine covariance matrix at time point 18, a total of eighteen target matrices are used to inform its estimation. This corresponds to the nine default target matrices in addition to the covariance estimates from the nine previous time points.

Applying TAS to each time point yields both a covariance matrix estimate and the associated weights for each target matrix used in the target set. For example, Figure 4.7 shows the weights from TAS for the arterial samples of the treatment group at time point 2. Here, it can be seen that the weights are mostly shared between the TAS estimate for time point 1 (60%) and the sample covariance matrix for time point 2 (40%).

A similar pattern is observed for the majority of arterial samples (both control and treatment), where the TAS covariance estimate from the previous time point is allocated a high weight when used as a target matrix for the current time point (not shown). This result supports the use of TAS in this longitudinal data collection process, in which earlier time points have relevant information to improve inference in later time points.

A different behaviour is observed for the microdialysis treatment samples, for which TAS weights tend to prefer the regular default shrinkage targets (see Figure 4.8). This is further justification behind using our multi target approach. Our intuition about covariance estimates from previous time points does not hold in this case and would have led to increased estimation error had we decided to use this target in single target linear shrinkage.

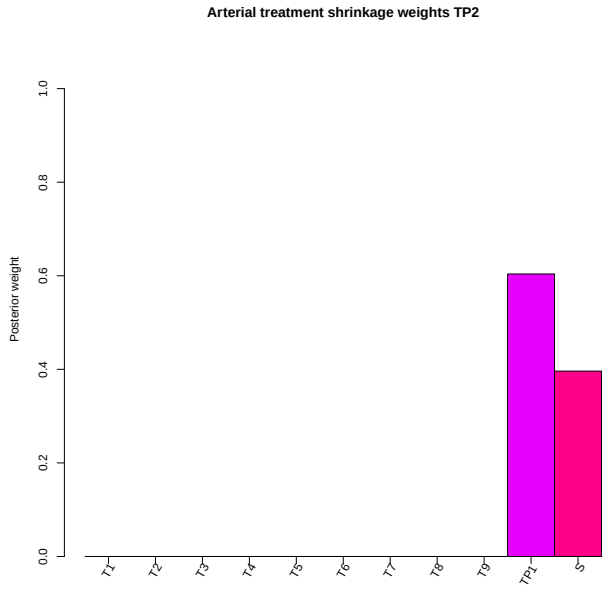


Fig. 4.7 TAS weights for the arterial samples of the treatment group for time point 2.

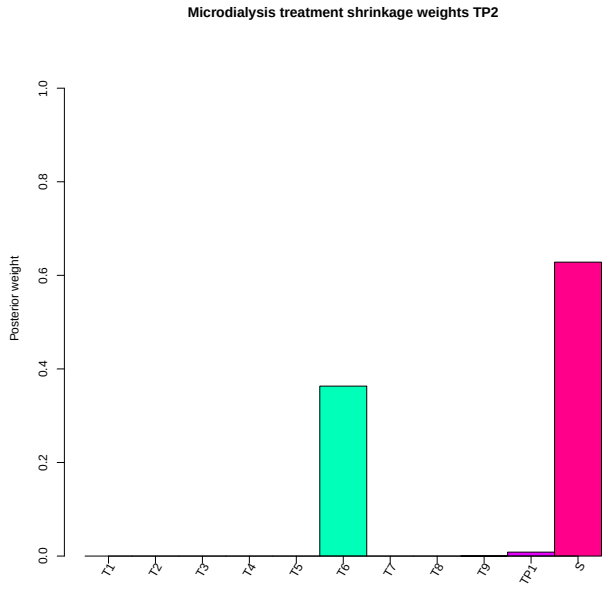


Fig. 4.8 TAS weights for the microdialysis samples of the treatment group for time point 2.

Cluster analysis

We wish to demonstrate how the estimate of a covariance matrix may be used for downstream multivariate analyses. For the current dataset, we are interested in how cytokines interact with each other so that we may better characterise the differences

between the treatment and control groups. Cluster analysis is a widely-used method in statistics and machine learning for grouping together items with similar quantitative characteristics. It is 'unsupervised' in the sense that the algorithms are given no ground truth to measure their success and adapt, meaning that they must find their own structure in the data. It can therefore be a useful aid in exploratory analyses in which the underlying system is not fully understood, as with cytokines in TBI.

Cluster analysis is the category name for a large number of methods, none of which is perfect in all situations. The decision to choose a method is largely context-specific and can depend upon decisions such as how to define similarity between measurements, how clusters should be defined based on this similarity, and how the number of clusters should be selected. For example, one might choose a clustering strategy for which the distance between items in a cluster is minimised whilst the distance between clusters is maximised.

Since we wish to investigate cytokine interactions, we use the Pearson (marginal) correlation of cytokine expressions as the feature to be used as input for clustering. The estimates for these coefficients are readily extracted from the covariance matrix estimated by TAS. Agglomerative hierarchical clustering was then applied to these correlation matrices using the Euclidean distance to define similar correlations and Ward's criterion [78] to distinguish between clusters of these correlations. The number of clusters at each time point was determined by optimising the silhouette index [91]. The silhouette index is a metric based on the ratio of distances of points, maximised by the clustering that shows maximum similarity within and discrepancy between clusters. Given a clustering of k clusters labelled C_i, $i = 1,\ldots,k$, to which we have assigned data points x_n, $n = 1,\ldots,N$, then define the quantity

$$a(n) = \frac{1}{|C_i| - 1} \sum_{x_j \in C_i, n \neq j} d(n, j), \tag{4.1}$$

for $x_n \in C_i$ with $d(n, j)$ equal to some distance metric between two data points x_n, x_j (in our case Euclidean). This makes $a(n)$ the mean distance between x_n and all other data points assigned to the same cluster $x_j \in C_i$. Further, define

$$b(n) = \min_{C_l} \frac{1}{|C_l|} \sum_{x_j \in C_l} d(n, j), \tag{4.2}$$

where the minimum is computed over all clusters C_l of which x_n is not a member. This makes $b(n)$ the minimum mean distance to the points in other clusters from x_n if its current cluster is excluded from the computation. The silhouette of point x_n is then

$$s(n) = \frac{b(n) - a(n)}{\max\{a(n), b(n)\}}. \tag{4.3}$$

It can be seen that $-1 \leq s(n) \leq 1$, with positive values arising when the intra-cluster mean distance is small relative to the inter-cluster one, and vice versa for negative values. The silhouette of a cluster C_i is

$$S_i = \frac{1}{|C_i|} \sum_{x_n \in C_i} s(n) \tag{4.4}$$

and a clustering's Silhouette Index is

$$S = \frac{1}{k} \sum_i S_i, \tag{4.5}$$

which can be interpreted as the average silhouette value across all of the N data points. The clustering that maximises this value is selected as the number of clusters to proceed with in the analysis. We note that other metrics could also have been used, but that our aim is to demonstrate how covariance estimation can facilitate clustering, rather than the clustering interpretation itself.

Applying cluster analysis to each stratified collection of patient-cytokine expression values across time points gives different clusterings for each sample type and treatment status. We use Cluster Of Cluster Analysis (COCA) [80, 20] to summarise multiple clusterings of the same set of cytokines, thereby identifying which cytokines' correlations were often clustered together over time. COCA proceeds as follows:

1. For each combination of sample type and treatment status, create a Matrix of Clusters (MOC), which contains the clustering allocations obtained across the time points. The rows of the MOC represent the cytokines and each column represents a cluster for a specific time point. The entries are binary, taking the value 1 only if that cytokine was in that cluster at that time point. For example, if there were only two time points and three clusters were identified for the first and two for the second, then the resulting MOC would have five columns.

2. Randomly subsample (without replacement) the rows of the MOC and apply hierarchical clustering to each subsampled matrix.

3. Create a square consensus matrix whose rows and columns represent the considered cytokines. The entries of the consensus matrix are the proportion of times two cytokines were clustered together when they were both subsampled in step 2.

4. Apply hierarchical clustering to the consensus matrix in step 3 and output the corresponding clustering labels.

We ran COCA using 0.8 as the sampling fraction (step 2 above) of rows and columns to include in 1000 subsamples (these last two values are default parameters in the software

[20]) and by again optimising the silhouette index to determine the number of clusters to use for step 4.

Steps 2 and 3 of COCA correspond to a method known as consensus clustering. Consensus clustering aims to assess the robustness of a clustering structure both to perturbations in the data and stochasticity of the clustering algorithm. In this way, the clustering labels that COCA outputs can be seen as a robust global clustering of multiple clustering structures.

As an illustration, Figure 4.9 shows the clustered correlation matrix for the arterial samples of the treatment group at time point 2. In this case, the silhouette index has selected three clusters of cytokines, with two being quite well-defined and the larger one being more heterogeneous. The heterogeneous cluster can be seen as an example of uncertainty in the identified clustering structure, which we expand upon in Section 4.5.

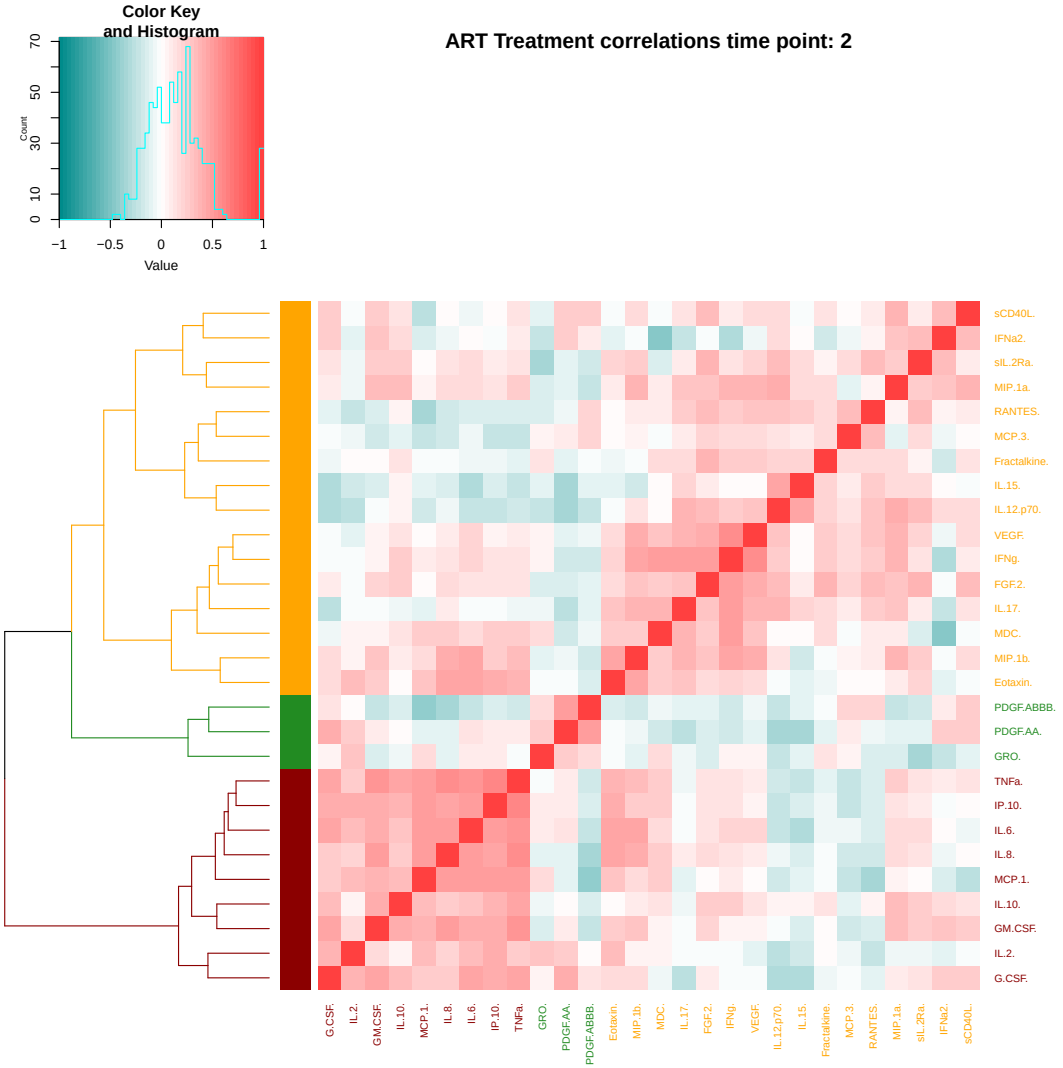


Fig. 4.9 Correlation clustering for the arterial samples of the treatment group for time point 2. The number of clusters identified by the silhouette index is shown by the number of colours used in the dendrogram of the rows.

We now summarise the clusterings for each sample type and treatment group using COCA. Figures 4.10 and 4.11 show the result of COCA for the arterial samples of the control and treatment groups, respectively.

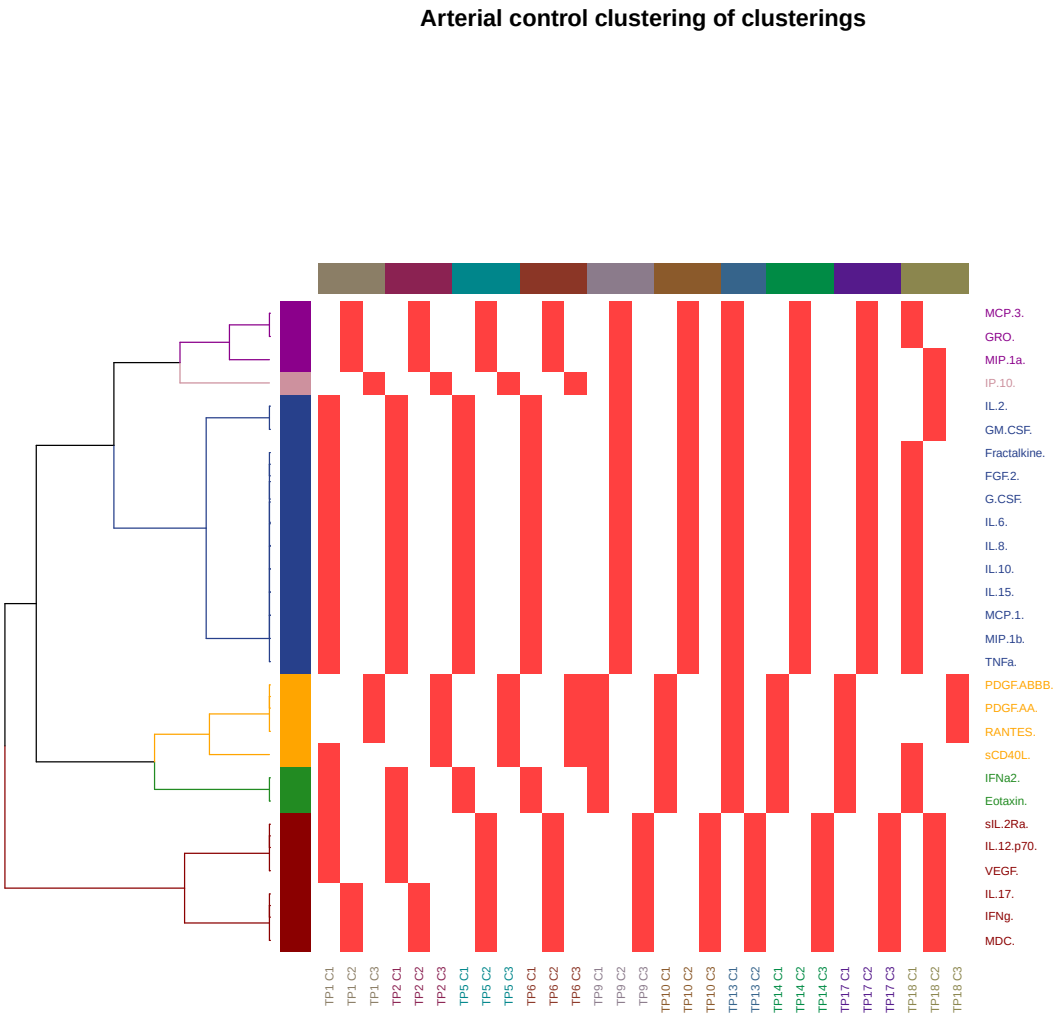


Fig. 4.10 COCA for the arterial samples of the control group using TAS. The labels of the columns indicate a cluster allocation for a given time point, e.g. time point 10 cluster 2 is denoted as TP10 C2. Each time point has been given its own colour for clarity.

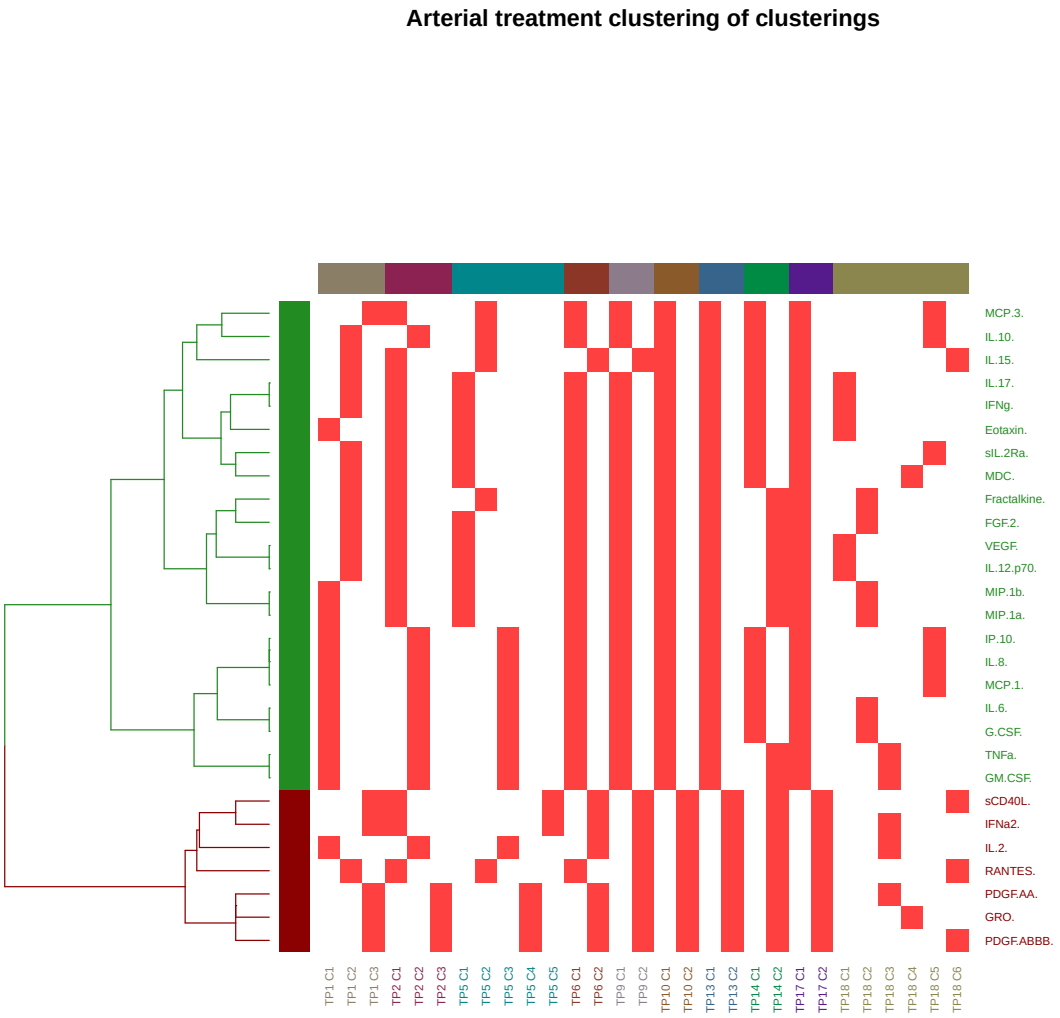


Fig. 4.11 COCA for the arterial samples of the treatment group using TAS. The labels of the columns indicate a cluster allocation for a given time point, e.g. time point 10 cluster 2 is denoted as TP10 C2. Each time point has been given its own colour for clarity.

In this case, COCA identifies six clusters of cytokines that are consistently grouped together across the time points for the arterial control group cytokines but only two clusters for the arterial treatment group. However, there is considerable overlap between the maroon cluster in the treatment group (Figure 4.11) and the combined yellow and green clusters in the control group (Figure 4.10). The common cytokines between these clusters are PDGF.ABBB, PDGF.AA, RANTES, sCD40L, and IFNa2, whereas IL2 and GRO appear in the treatment cluster, but not in the analogous control cluster, and vice versa for Eotaxin. Even though these clusters contain common cytokines, the biggest difference is the position in their dendrograms. The position in the treatment dendrogram is farthest away from the other cytokines. In the control group, their position is with

the blue, pink, and purple clusters after the first branch of the dendrogram, and so they are considered as more similar to the rest of the cytokines. Instead it is the maroon cluster of the control cytokines that are considered as most different to the others, which includes sIL.2Ra, IL12p70, VEGF, IL17, IFNg, and MDC. These cytokines are still considered to be similar in the treatment group (appearing close to each other in the dendrogram), but are also part of the large green cluster and thus deemed to cluster similarly to the majority of other cytokines. It is unclear whether these contrasts in clusterings are due to the treatment status of the patients, but it could provide some target groups of cytokines to analyse in further studies.

Figures 4.12 and 4.13 display the outputs of COCA for the microdialysis samples.

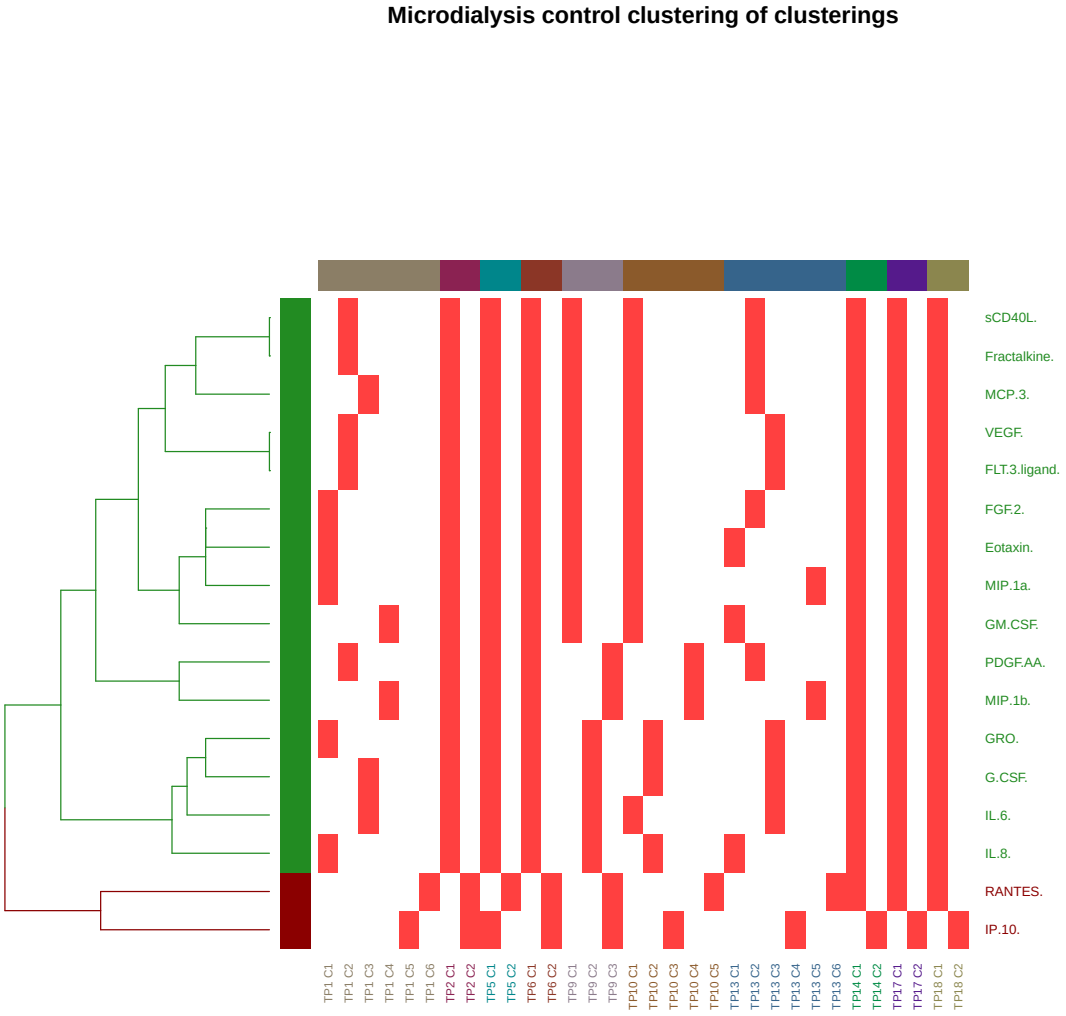


Fig. 4.12 COCA for the microdialysis samples of the control group using TAS. The labels of the columns indicate a cluster allocation for a given time point, e.g. time point 10 cluster 2 is denoted as TP10 C2. Each time point has been given its own colour for clarity.

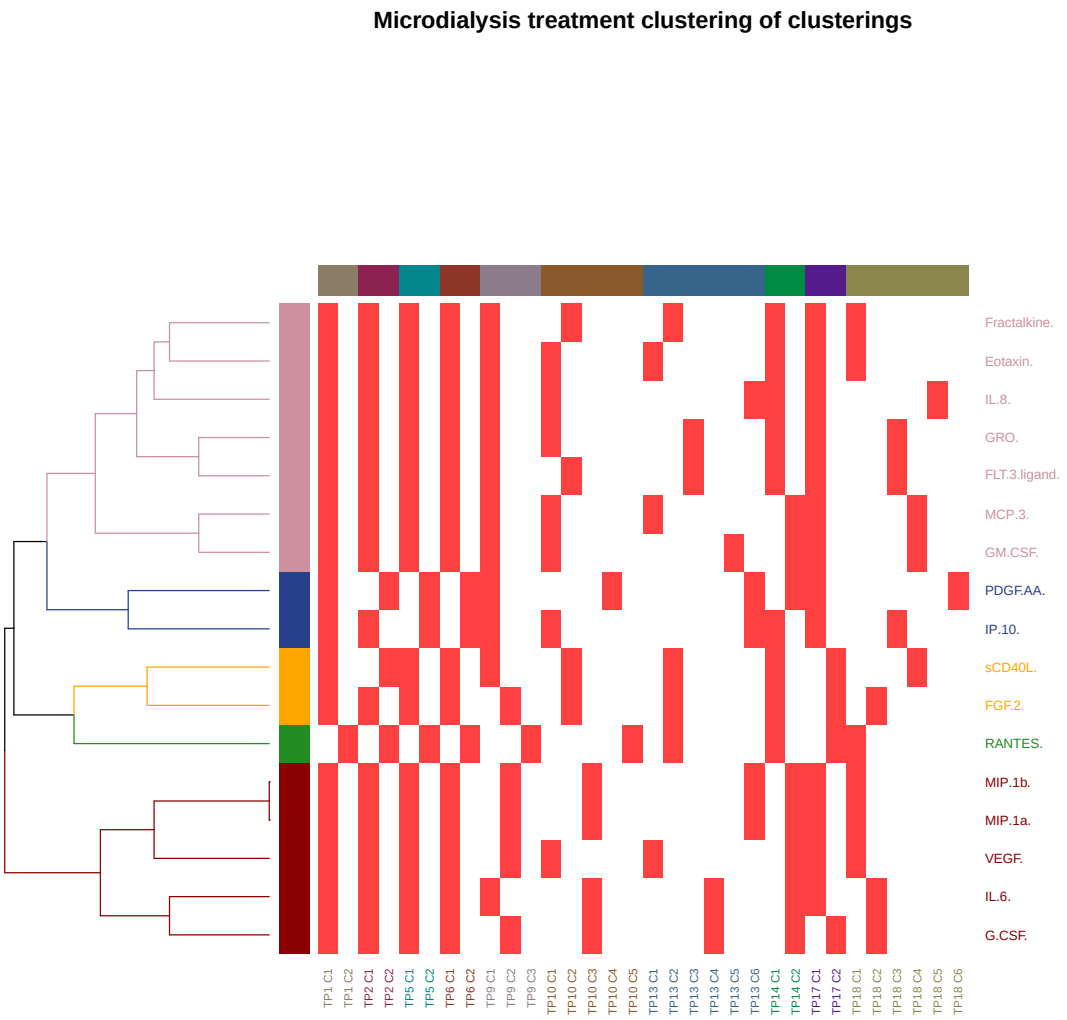


Fig. 4.13 COCA for the microdialysis samples of the treatment group using TAS. The labels of the columns indicate a cluster allocation for a given time point, e.g. time point 10 cluster 2 is denoted as TP10 C2. Each time point has been given its own colour for clarity.

This time, two clusters have been identified for the control group and five clusters for the treatment group. However, unlike for the arterial samples, there does not seem to be much overlap between the clusters or apparent similarity in dendrogram distances. For the control group, the small cluster that is distinguished from the other cytokines consists of RANTES and IP10. In the treatment group, these cytokines differ by a couple of branches of separation. Their closest cytokines in the treatment group are FGF2 and sCD40L for RANTES and PDGF.AA for IP10, all of which are largely separated in the control group. The most different cluster in the treatment group consists of MIP1b, MIP1a, VEGF, IL6, and GCSF. The only dendrogram distance of those cytokines that remain within one branch of separation in the control group is between

GCSF and IL6. This large difference in similarity for the microdialysis samples is another avenue for future exploration.

Network reconstruction

As mentioned in Section 1.5 and Chapter 3, network reconstruction aims to identify significant pairwise interactions between objects of interest, in our case cytokines. The resulting significant interactions can then be visualised as edges between cytokines (nodes) that represent cytokines in what is collectively termed a network. If desired, groups of cytokines may then be manually extracted through consideration of the resulting edges in the network.

Similar to cluster analysis, there are many different ways to define an interaction (edge) in the network and estimate its significance. In this analysis, we assume a Gaussian Graphical Model (GGM). By imposing an assumption of Gaussianity on the log-expression values, the entries of the inverse covariance matrix (precision matrix) encode the conditional dependencies of the cytokines through a statistic called the partial correlation coefficient (Section 1.5). The partial correlation coefficient in this analysis quantifies the interaction of two cytokines whilst conditioning on the interactions with all others, inducing a more accurate measure of the direct pairwise relationship of each cytokine. This powerful interpretation of interaction is a result of the Gaussian assumption and can thus be thought of as a trade-off with model simplicity. For an introduction to GGMs, see [32, 36].

Estimates for partial correlation coefficients can be obtained by inverting an estimated covariance matrix, obtained here using TAS. In order to determine which interactions are statistically significant, we employ the popular empirical Bayes hypothesis testing approach of Schäfer and Strimmer [94]. It assumes that the empirical distribution of partial correlations is a mixture of a null and alternative distribution, corresponding to non-interesting interactions (partial correlation not significantly different to zero) and interesting ones (partial correlation significantly different from zero) their method determines a partial correlation threshold adaptively from the data, controlling the estimated false positive rate. This generates a p-value for each interaction and corresponding q-value after correction for multiple testing. These statistics enable network reconstruction by forming edges from the significant interactions. This method is not only capable of high-dimensional data but actually benefits from it, since the more interactions there are the better it is able to fit its null and alternative models.

Similar to the cluster analysis, we estimate time-point specific cytokine networks each category of sample type and treatment status. To summarise the results over time, we opted to count the number of times a pair of cytokines has a significant interaction across all time points for each sample type. The counts for the control and treatment

groups were then compared. The rationale behind this summary and comparison is to see which interactions are consistently present across time points. In this comparison, there would be three interesting situations: (i) if a pair of cytokines has a consistently present edge over time in the control group only (ii) vice versa for the treatment group only (iii) or for both groups. However, care must be taken in this case to not compare across sample types, since the partial correlation coefficient conditions on all other observed cytokines and the set of observed cytokines is different for the arterial and microdialysis group (due to our quality control step).

As an example, Figure 4.14 shows the reconstructed networks for the arterial samples of both treatment groups at time point 10. In this case, the network shows that for this time point there are only two edges in common between control and treatment groups, IL6-GCSF and PDGF.AA-PDGF.ABBB.

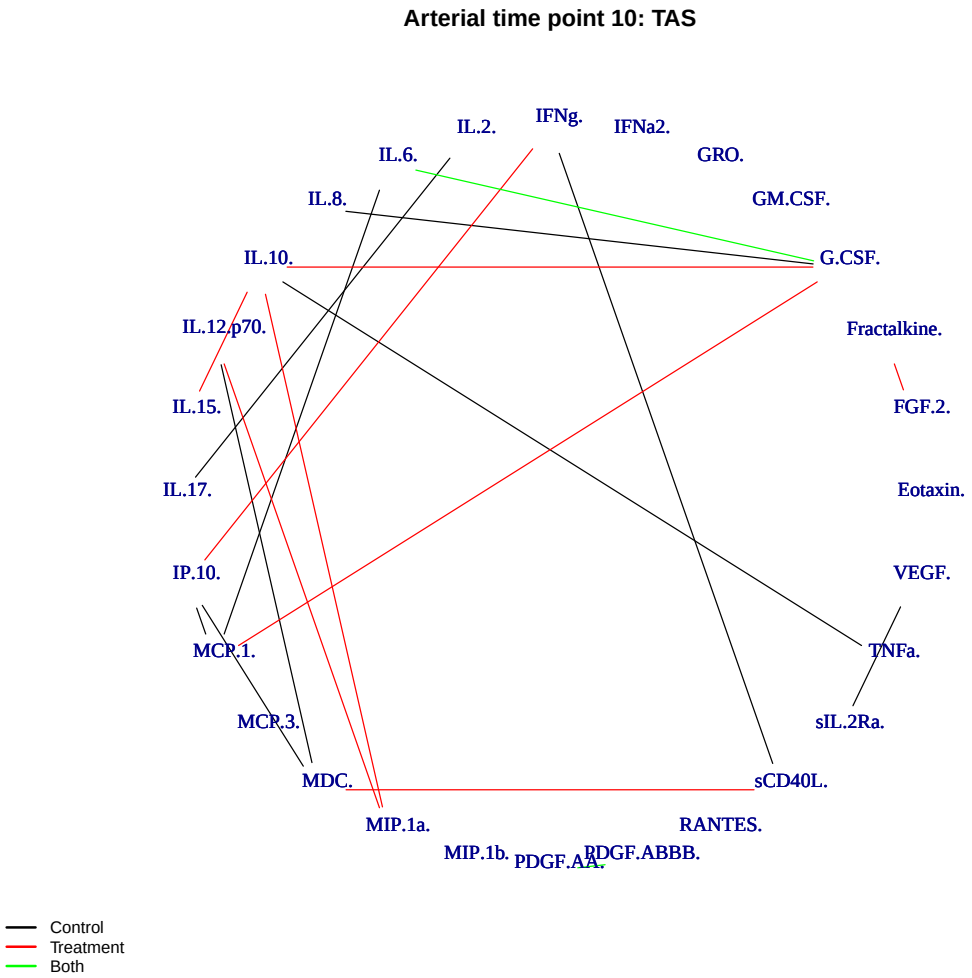


Fig. 4.14 Reconstructed networks for arterial samples at time point 10 using TAS.

Here, we summarise the networks for each sample type and treatment status by counting the number of times an edge appears between each pair of cytokines across the time points. Figure 4.15 displays the count of each edge between a pair of cytokines for the arterial samples.

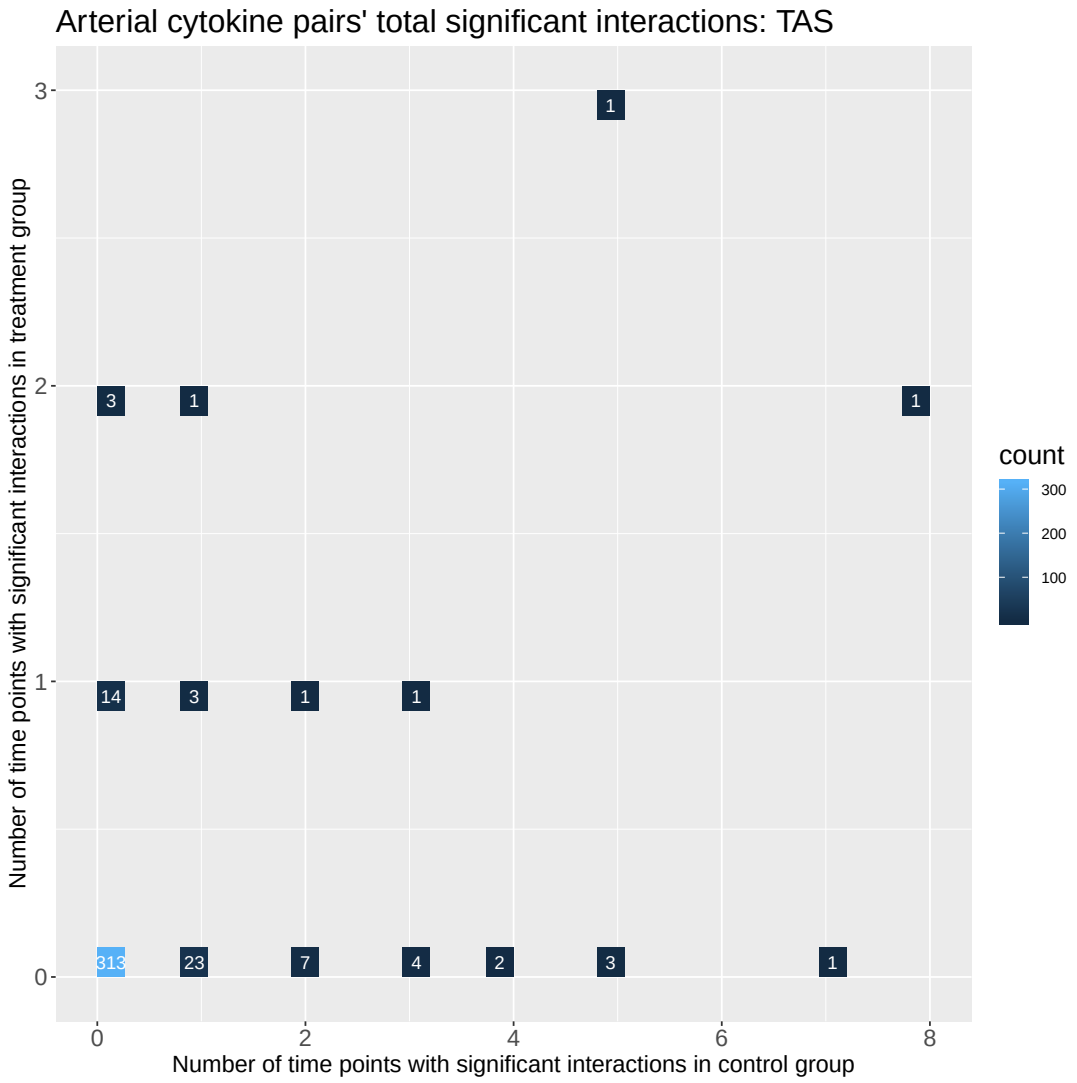


Fig. 4.15 Network edge summaries for arterial samples using TAS. The x-axis shows the number of times an edge appeared in the control group networks, whilst the y-axis does the same for the treatment group. The number inside the blue boxes shows how many pairs of cytokines had an edge appear that many times.

The edges that stand out from this figure are GCSF-IL6 (control=5, treatment=3), PDGF.AA-PDGF.ABBB (control=8, treatment=2), sIL.2Ra-VEGF (control=7, treatment=0) and IL.10-TNFa, IP.10-MCP.1, IP.10-MDC (control=5, treatment=0). It seems as though the interactions between GCSF-IL6 and PDGF.AA-PDGF.ABBB are present in both the treatment and control group. It also appears that no cytokine interactions are

specific to the treatment group, but that sIL.2Ra-VEGF, IL.10-TNFa, IP.10-MCP.1, and IP.10-MDC are specific to the control group.

Figure 4.16 displays the count of each edge between a pair of cytokines for the microdialysis samples.

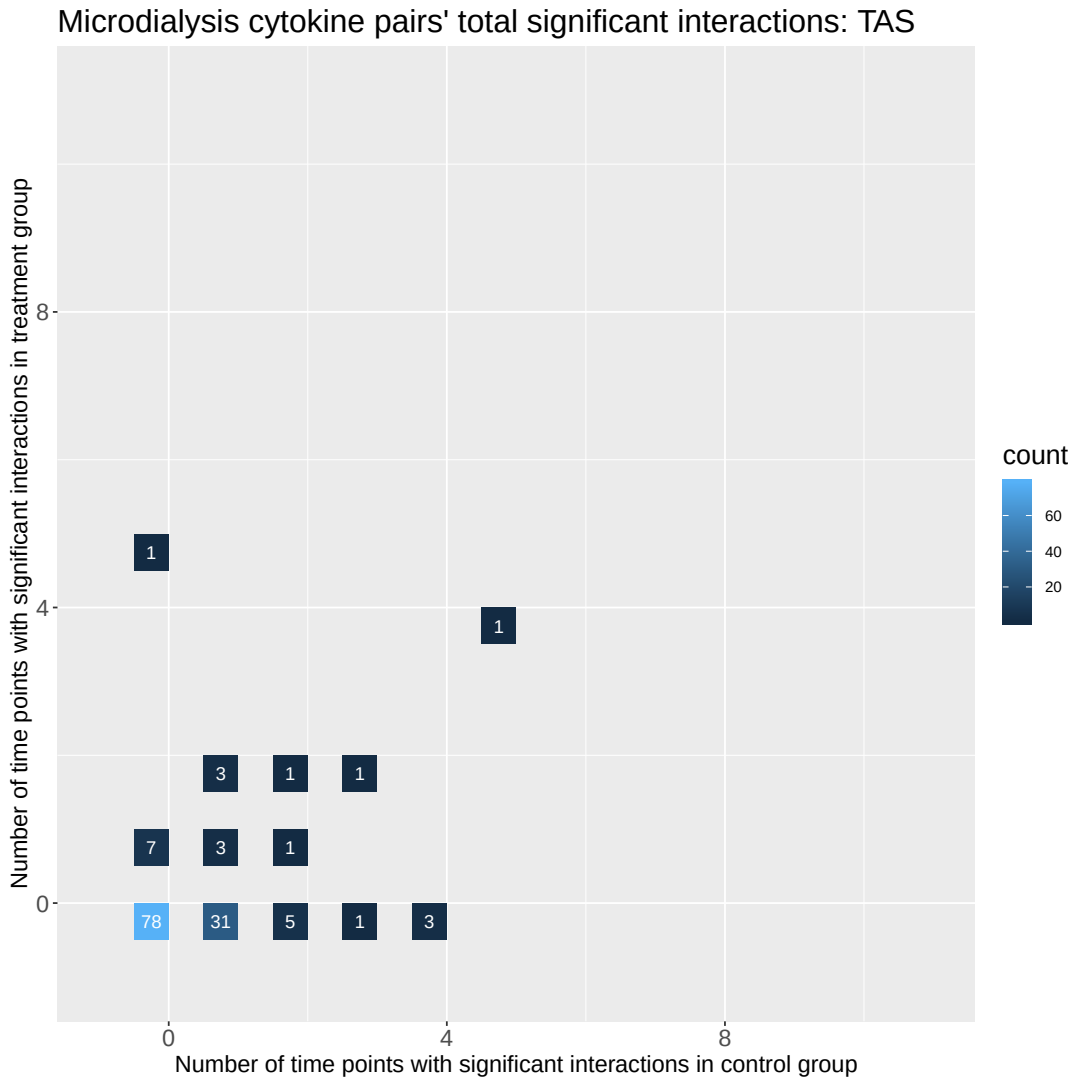


Fig. 4.16 Network edge summaries for microdialysis samples using TAS. The x-axis shows the number of times an edge appeared in the control group networks, whilst the y-axis does the same for the treatment group. The number inside the blue boxes shows how many pairs of cytokines had an edge appear that many times

The edges that stand out here are IP.10-PDGF.AA (control=0, treatment=5) and MIP.1a-MIP.1b (control=5, treatment=4). In this case, MIP.1a-MIP.1b appears often in both treatment cases, whereas IP.10-PDGF.AA seems to be specific to the treatment group. As with the cluster analysis, it is unclear whether these differences are caused by the treatment, but they could be useful for further analysis.

4.5 Discussion

In this chapter, we have described the analysis of a cytokine expression dataset for patients with TBI. This analysis has improved upon Helmy et al. [56] who did not perform high-dimensional covariance estimation and pooled data between time points, potentially averaging out meaningful effects. We have shown how TAS can overcome the problem of high-dimensional cytokine datasets. We have also shown through TAS that covariance information can be shared between time points to facilitate the multivariate analysis of a time-series dataset. Using clustering and network reconstruction, we have identified some groups (and pairs) of cytokines that often interact with each other over time and how these groups differ between sample type and treatment status. This sheds some light on the largely uncharacterised multivariate molecular behaviour of cytokines and their response to both TBI and IL1ra treatment and therefore presents some potential avenues for further analysis and studies.

As with any real data analysis there are limitations to this study. Firstly, we have used a fairly arbitrary cut-off of 20% for missing values in both sample types. Although this can be considered as conservative and has some basis for justification through our explanation in Section 4.3, it does exclude many cytokines from the analysis and potentially useful information. In particular, the set of cytokines considered for the partial correlation calculation is crucial. As such, if cytokines that are central to the underlying interactions are excluded, then false positive edges will be included in the estimated network. One possible route to overcoming this drawback is to perform more diligent missing value imputation. Since our collaborators highlighted the similarity between the arterial and venous datasets, one avenue would be to use the venous samples to inform the imputation of the missing arterial values. The relationships between the arterial and venous samples, including any correlation of missing values, would first need to be established in order to do this.

The cluster analysis also comes with a number of potential limitations. The method used to cluster, the metric used for distance, the metric used to define clusters, and the method to determine the number of clusters are all subjective decisions that have other alternatives. A more robust procedure for choosing the number of clusters could be to run the clustering using multiple difference methods and then select an aggregate for the number of clusters that was chosen by them, e.g. the most common number of clusters. Whilst this, and other methods exist for the performing a cluster analysis, we highlight that the purpose of this analysis was to present a pipeline of statistical methods to analyse this type of dataset, rather than necessarily drawing biological interpretation from the results. Therefore, whilst selecting other ways of the performing each individual step, the process of using TAS to facilitate multivariate analysis remains unchanged.

The decision to not use PPCA for covariance estimation was based upon the missing values of the dataset being censored, rather than missing at random. One way to overcome this limitation is to explore PPCA for censored data [19]. PPCA could be used to generate a target matrix for TAS in this case. Since the venous datasets were not used and have been said to be biologically associated with the arterial samples, PPCA could be trained on the venous dataset and used as a target matrix for the same time point in the arterial dataset. The target that PPCA provides will contain latent factor covariance information that is identified within the venous samples, and therefore unable to be captured by the other target matrices used by TAS.

Another unused part of the dataset in our analysis was the abundance of clinical variables that are available. We chose to only use the treatment status to simplify our analysis. However, it would be interesting to explore how other quantitative metrics are related to the cytokine expression values. These include multiple scores for TBI severity, systemic infection status, and C-reactive protein and white blood cell count. We view the analysis pipeline presented here as a starting point for tackling these questions, but note that the dataset does not have enough samples to simultaneously stratify by all of these metrics.

Chapter 5

Conclusions and further work

In this thesis, we considered the problem of high-dimensional covariance matrix estimation from high-throughput molecular data. In particular, we focussed on regularised covariance estimators that are computationally efficient, can handle missing values, and are relatively scalable for large sample sizes and dimensionality. We have also shown how to incorporate prior knowledge into covariance estimation. We finally aspired to translate this research into practice by providing publicly available software packages that will enable the adoption of our methods. While we focussed on applications in the context of functional genomics, our methods are generic and can also be applied to other types of high-dimensional data.

In Chapter 2 we focussed on linear shrinkage estimators, adopting a conjugate Bayesian framework that enables an efficient computational implementation. We introduced TAS, a new estimator that incorporates multiple shrinkage targets. We showed that the proposed estimator can perform similarly to the best single target shrinkage estimator, but that TAS is less prone to target misspecification because it is possible to specify a set of shrinkage target instead of a single one. Our estimator is therefore particularly useful in situations where there is uncertainty around the choice of target matrix. Another advantage of TAS is that target matrices from external datasets can be included into the target set without degradation in performance. For example, we showed that on data from The Cancer Genome Atlas this had a very beneficial effect when analysing the apoptosis pathway from ovarian cancer samples whereas this had no beneficial effect when analysing the p53 pathway for breast cancer samples, yet it did not harm the estimation. As such, TAS provides a low-risk method of including external information without inducing large amounts error when the external information is not relevant. TAS is implemented as an `R` software package and is publicly available at http://github.com/HGray384/TAS.

In Chapter 3 we considered PPCA as an alternative method for covariance estimation. While PPCA has been widely explored as a dimensionality reduction method, its use in

the context of covariance estimation has largely been overlooked in the literature. We evaluated three different algorithms for model fitting in the presence of missing values based on EM and VB implementations and proposed our own extension to another. We also highlighted how this estimator represents a large computational benefit for inverting the covariance matrix due to its low dimensional representation - especially useful for reconstructing networks of a large number of variables. We introduced the software package `pcaNet` that is publicly available at http://github.com/HGray384/pcaNet. `pcaNet` implements all three algorithms and contains functions for performing network reconstruction. The software was illustrated on a real Arabidopsis thaliana dataset. Moreover, we showed that the VB implementations outperform the standard EM ones through simulation study. Combined with an automatic selection of the latent dimension through the ARD prior, we advocate the use of VB for PPCA-based covariance estimation. We showed that the PPCA estimator outperforms TAS with its default target set when the underlying covariance structure is from a PPCA model, otherwise TAS appears to perform better. This motivates the potential use of a PPCA-structured target matrix within TAS, and we discussed how this may be constructed without reusing any data.

In Chapter 4 we presented a real data analysis of a case-control cytokine expression dataset for patients with TBI. We highlighted some of the difficulties associated with data of this type, such as small sample sizes, multiple time points, and missing values. We discussed that the nature of these missing values is complex, as they can be censored due to technology limitations. We therefore adopted a conservative criterion to remove cytokines with more than 20% missing values, imputing the remaining cytokines using a nearest neighbours procedure. This overcame some limitations in previous studies as it allowed us to study individual time points instead of pooling data across time points, albeit at the expense of excluding more cytokines from the analysis. We showed how multiple time points could be exploited by TAS to share information about their covariance structure. We also performed cluster analysis and network reconstruction in order to identify groups of cytokines that behave similarly over time and how these groups might differ between arterial/microdialysis samples and control/treatment groups. Whilst this has its limitations, and our results should be interpreted with caution, our proposed pipeline introduces a novel approach to perform multivariate analyses based on this type of data. This might also be relevant in other situations in which high-dimensional data is collected over a time course.

Throughout our discussion sections at the end of each chapter, we have identified promising directions for future work.

For TAS, the ability to include multiple target matrices has been shown to be greatly beneficial. A natural direction for future work would be to design a more comprehensive default target set when no external information is available or relevant. Steps towards

this could involve identifying more general forms of covariance matrices whose parameters could be adaptively estimated for the specific dataset at hand. Theoretically, this can be done by parametrising target matrices (with parameters that have their own prior distributions) and performing Bayesian inference over these parameters too. Practically, this causes significant increases in computational complexity and arguably negates the simplicity of a linear shrinkage model. A simpler empirical Bayes approach to this problem in the single-target model has been outlined in Hannart and Naveau [52], though it was shown that the solutions are the default targets for common parametrisations (e.g. when constrained to a diagonal matrix with equal or unequal entries). The implementation has yet to be applied to more complex parametrisations or when considering multiple targets, though this is a promising avenue for future work.

Validation of a default target set could be done by performing a large-scale simulation study on a range of different datasets. One potential target matrix to consider would be the PPCA estimator. The main challenge would be how to use the available data to estimate the PPCA model, since subsetting might not be optimal for high-dimensional datasets and using external data may bias the model training.

Another useful extension to the TAS model would be its applicability to non-Gaussian settings, so that it could potentially be applied to count-based sequencing data. Research in this direction would first be required to compile existing methods in the literature, such as those presented at the end of Chapter 2. A simulation study with non-Gaussian data-generating processes could then be used to determine the suitability of each method.

A robust and comprehensive simulation study would also be greatly beneficial in determining the advantages and disadvantages of the PPCA implementations explored in Chapter 3. By varying factors such as the data-generating model and its parameter settings, missing value percentage, the number of true latent dimensions (if any), and the dimensionality of the data, it would then be possible to create some guidelines as to which algorithm should be used for covariance estimation in which situation.

For both TAS and PPCA, another avenue for comparison with other methods would be to assess their network inference performance. This can be done by using the inverse of their estimated covariance matrix in combination with an edge selection method such as Strimmer [100] to enforce sparsity, as implemented in `pcaNet`. In particular, it would be interesting to theoretically assess the application of Strimmer [100] to such a structured partial correlation matrix induced via the PPCA model. Obtaining these sparse inverse covariance estimates would then allow the comparison against simultaneous estimation and variable selection methods such as the popular graphical lasso [45], and is another way of assessing the practical utility of accurate covariance estimation.

Future work on the cytokines analysis presented in Chapter 4 should be focussed towards incorporating the excluded data from the venous samples of the experiment. Since these samples are thought to biologically correlate with the arterial samples, they could be used to improve the initial imputation of missing data. Since this imputation would be biologically meaningful, the thresholding step taken in our analysis might then need to be relaxed, allowing more cytokines to be analysed and more insights could be gained. The venous dataset could also be used to train the PPCA model, which could then be included as an extra target in TAS and potentially improve estimation.

In sum, the methodology presented in this thesis extends the existing pool of available techniques for covariance estimation, not only through new methodology, but also by providing open-source well-documented analysis tools.We expect that these tools could be useful in a variety of contexts and, as discussed above, motivate further methodological developments.

References

[1] Agarwal, S. and Bishop, C. M. (2003). Improved variational approximation for bayesian pca. Technical report, Microsoft Research, Cambridge.

[2] Aguilar, O., Huerta, G., Prado, R., and West, M. (1998). Bayesian inference on latent structure in time series. *Bayesian Statistics*, 6(1):1–16.

[3] Aitchison, J. and Ho, C.-H. (1989). The multivariate Poisson-log normal distribution. *Biometrika*, 76(4):643–653.

[4] Ansari, A. and Jedidi, K. (2000). Bayesian factor analysis for multilevel binary observations. *Psychometrika*, 65(4):475–496.

[5] Bai, J. and Shi, S. (2011). Estimating high dimensional covariance matrices and its applications. *Annals of Economics and Finance*, 12(2):199–215.

[6] Bartz, D., Höhne, J., and Müller, K.-R. (2014). Multi-Target Shrinkage. *ArXiv e-prints*.

[7] Berger, J. O. (2013). *Statistical decision theory and Bayesian analysis*. Springer Science & Business Media.

[8] Bernardo, J., Bayarri, M., Berger, J., Dawid, A., Heckerman, D., Smith, A., and West, M. (2003). Bayesian factor regression models in the "large p, small n" paradigm. *Bayesian statistics*, 7:733–742.

[9] Bernardo, J. M. and Smith, A. F. (2009). *Bayesian theory*, volume 405. John Wiley & Sons.

[10] Betz, A. L., Yang, G.-Y., and Davidson, B. L. (1995). Attenuation of stroke size in rats using an adenoviral vector to induce overexpression of interleukin-1 receptor antagonist in brain. *Journal of Cerebral Blood Flow & Metabolism*, 15(4):547–551.

[11] Bhattacharya, A. and Dunson, D. B. (2011). Sparse bayesian infinite factor models. *Biometrika*, pages 291–306.

[12] Bickel, P. J. and Levina, E. (2008). Covariance regularization by thresholding. *Ann. Statist.*, 36(6):2577–2604.

[13] Bickel, P. J. and Li, B. (2006). Regularization in statistics. *Test*, 15(2):271–344.

[14] Bien, J. and Tibshirani, R. J. (2011). Sparse estimation of a covariance matrix. *Biometrika*, 98(4):807–820.

[15] Bishop, C. (1999). Bayesian PCA. In *Advances in Neural Information Processing Systems*, pages 382–388. MIT Press.

[16] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

[17] Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877.

[18] Bochao, J., Suwa, X., Guanghua, X., Vishal, L., and Faming, L. (2017). Learning gene regulatory networks from next generation sequencing data. *Biometrics*, 73(4):1221–1230.

[19] Buettner, F., Moignard, V., Göttgens, B., and Theis, F. J. (2014). Probabilistic pca of censored data: accounting for uncertainties in the visualization of high-throughput single-cell qpcr data. *Bioinformatics*, 30(13):1867–1875.

[20] Cabassi, A. and Kirk, P. D. (2019). Multiple kernel learning for integrative consensus clustering of genomic datasets. *arXiv preprint arXiv:1904.07701*.

[21] Cai, T. and Liu, W. (2011). Adaptive thresholding for sparse covariance matrix estimation. *J. Amer. Statist. Assoc.*, 106(494):672–684.

[22] Cai, T. T., Ma, Z., and Wu, Y. (2013). Sparse pca: Optimal rates and adaptive estimation. *The Annals of Statistics*, 41(6):3074–3110.

[23] Chen, C. F. (1979). Bayesian inference for a normal dispersion matrix and its application to stochastic multiple regression analysis. *J. Roy. Statist. Soc. Ser. B*, 41(2):235–248.

[24] Chen, Y., Wiesel, A., Eldar, Y. C., and Hero, A. O. (2010). Shrinkage algorithms for mmse covariance estimation. *IEEE Transactions on Signal Processing*, 58(10):5016–5029.

[25] Clark, S. R., McMahon, C. J., Gueorguieva, I., Rowland, M., Scarth, S., Georgiou, R., Tyrrell, P. J., Hopkins, S. J., and Rothwell, N. J. (2008). Interleukin-1 receptor antagonist penetrates human brain at experimentally therapeutic concentrations. *Journal of Cerebral Blood Flow & Metabolism*, 28(2):387–394.

[26] Clausen, F., Hånell, A., Björk, M., Hillered, L., Mir, A. K., Gram, H., and Marklund, N. (2009). Neutralization of interleukin-1β modifies the inflammatory response and improves histological and cognitive outcome following traumatic brain injury in mice. *European Journal of Neuroscience*, 30(3):385–396.

[27] Clausen, F., Hånell, A., Israelsson, C., Hedin, J., Ebendal, T., Mir, A. K., Gram, H., and Marklund, N. (2011). Neutralization of interleukin-1β reduces cerebral edema and tissue loss and improves late cognitive outcome following traumatic brain injury in mice. *European Journal of Neuroscience*, 34(1):110–123.

[28] Cloonan, N., Forrest, A. R., Kolle, G., Gardiner, B. B., Faulkner, G. J., Brown, M. K., Taylor, D. F., Steptoe, A. L., Wani, S., Bethel, G., et al. (2008). Stem cell transcriptome profiling via massive-scale mRNA sequencing. *Nature methods*, 5(7):613–619.

[29] Csardi, G. and Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal, Complex Systems*, 1695(5):1–9.

[30] Daniels, M. J. and Kass, R. E. (1999). Nonconjugate bayesian estimation of covariance matrices and its use in hierarchical models. *Journal of the American Statistical Association*, 94(448):1254–1263.

[31] Daniels, M. J. and Kass, R. E. (2001). Shrinkage estimators for covariance matrices. *Biometrics*, 57(4):1173–1184.

[32] Dempster, A. P. (1972). Covariance selection. *Biometrics*, pages 157–175.

[33] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

[34] Eddelbuettel, D., François, R., Allaire, J., Ushey, K., Kou, Q., Russel, N., Chambers, J., and Bates, D. (2011). Rcpp: Seamless r and c++ integration. *Journal of Statistical Software*, 40(8):1–18.

[35] Eddelbuettel, D. and Sanderson, C. (2014). Rcpparmadillo: Accelerating r with high-performance c++ linear algebra. *Computational Statistics & Data Analysis*, 71:1054–1063.

[36] Edwards, D. (2012). *Introduction to graphical modelling*. Springer Science & Business Media.

[37] Efron, B., Tibshirani, R., Storey, J., and Tusher, V. (2001). Empirical Bayes analysis of a microarray experiment. *Journal Of The American Statistical Association*, 96(456):1151–1160.

[38] Emsley, H., Smith, C., Georgiou, R., Vail, A., Hopkins, S., Rothwell, N., and Tyrrell, P. (2005). A randomised phase ii study of interleukin-1 receptor antagonist in acute stroke patients. *Journal of Neurology, Neurosurgery & Psychiatry*, 76(10):1366–1372.

[39] Engel, J., Buydens, L., and Blanchet, L. (2017). An overview of large-dimensional covariance and precision matrix estimators with applications in chemometrics. *Journal of Chemometrics*, 31(4).

[40] Ettinger, D. S., Wood, D. E., Aisner, D. L., Akerley, W., Bauman, J., Chirieac, L. R., D'Amico, T. A., DeCamp, M. M., Dilling, T. J., Dobelbower, M., et al. (2017). Non–small cell lung cancer, version 5.2017, nccn clinical practice guidelines in oncology. *Journal of the National Comprehensive Cancer Network*, 15(4):504–535.

[41] Fan, J., Liao, Y., and Liu, H. (2016). An overview of the estimation of large covariance and precision matrices. *Econom. J.*, 19(1):C1–C32.

[42] Fan, J., Liao, Y., and Mincheva, M. (2011). High dimensional covariance matrix estimation in approximate factor models. *Annals of statistics*, 39(6):3320.

[43] Fan, J., Liao, Y., and Mincheva, M. (2013). Large covariance estimation by thresholding principal orthogonal complements. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(4):603–680.

[44] Fisher, T. J. and Sun, X. (2011). Improved Stein-type shrinkage estimators for the high-dimensional multivariate normal covariance matrix. *Comput. Statist. Data Anal.*, 55(5):1909–1918.

[45] Friedman, J., Hastie, T., and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441.

[46] Gaiteri, C., Ding, Y., French, B., Tseng, G. C., and Sibille, E. (2014). Beyond modules and hubs: the potential of gene coexpression networks for investigating molecular mechanisms of complex brain disorders. *Genes, Brain and Behavior*, 13(1):13–24.

[47] Galea, J., Ogungbenro, K., Hulme, S., Greenhalgh, A., Aarons, L., Scarth, S., Hutchinson, P., Grainger, S., King, A., Hopkins, S. J., et al. (2011). Intravenous anakinra can achieve experimentally effective concentrations in the central nervous system within a therapeutic time window: results of a dose-ranging study. *Journal of Cerebral Blood Flow & Metabolism*, 31(2):439–447.

[48] Gallopin, M., Rau, A., and Jaffrézic, F. (2013). A hierarchical Poisson log-normal model for network inference from RNA sequencing data. *PloS one*, 8(10):e77503.

[49] Ghirnikar, R. S., Lee, Y. L., and Eng, L. F. (1998). Inflammation in traumatic brain injury: role of cytokines and chemokines. *Neurochemical research*, 23(3):329–340.

[50] Gray, H., Leday, G. G., Vallejos, C. A., and Richardson, S. (2018). Shrinkage estimation of large covariance matrices using multiple shrinkage targets. *arXiv preprint arXiv:1809.08024*.

[51] Hahn, P. R., He, J., and Lopes, H. (2018). Bayesian factor model shrinkage for linear iv regression with many instruments. *Journal of Business & Economic Statistics*, 36(2):278–287.

[52] Hannart, A. and Naveau, P. (2014). Estimating high dimensional covariance matrices: a new look at the Gaussian conjugate framework. *J. Multivariate Anal.*, 131:149–162.

[53] Helmy, A., Antoniades, C. A., Guilfoyle, M. R., Carpenter, K. L., and Hutchinson, P. J. (2012). Principal component analysis of the cytokine and chemokine response to human traumatic brain injury. *PloS one*, 7(6):e39677.

[54] Helmy, A., Carpenter, K. L., Skepper, J. N., Kirkpatrick, P. J., Pickard, J. D., and Hutchinson, P. J. (2009). Microdialysis of cytokines: methodological considerations, scanning electron microscopy, and determination of relative recovery. *Journal of neurotrauma*, 26(4):549–561.

[55] Helmy, A., Guilfoyle, M. R., Carpenter, K. L., Pickard, J. D., Menon, D. K., and Hutchinson, P. J. (2014). Recombinant human interleukin-1 receptor antagonist in severe traumatic brain injury: a phase ii randomized control trial. *Journal of Cerebral Blood Flow & Metabolism*, 34(5):845–851.

[56] Helmy, A., Guilfoyle, M. R., Carpenter, K. L., Pickard, J. D., Menon, D. K., and Hutchinson, P. J. (2016). Recombinant human interleukin-1 receptor antagonist promotes m1 microglia biased cytokines and chemokines following human traumatic brain injury. *Journal of Cerebral Blood Flow & Metabolism*, 36(8):1434–1448.

[57] Hoeting, J. A., Madigan, D., Raftery, A. E., and Volinsky, C. T. (1999). Bayesian model averaging: a tutorial. *Statist. Sci.*, 14(4):382–417.

[58] Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417.

[59] Ikeda, Y., Kubokawa, T., and Srivastava, M. S. (2016). Comparison of linear shrinkage estimators of a large covariance matrix in normal and non-normal distributions. *Computational Statistics & Data Analysis*, 95:95–108.

[60] Ilin, A. and Raiko, T. (2010). Practical Approaches to Principal Component Analysis in the Presence of Missing Values. *Journal of Machine Learning Research*, 11:1957–2000.

[61] Jacobson, A. (2015). R-based api for accessing the mskcc cancer genomics data server. r package version 1.2. 5.

[62] Kass, R. E. and Raftery, A. E. (1995). Bayes factors. *J. Amer. Statist. Assoc.*, 90(430):773–795.

[63] King, M.-C., Marks, J. H., Mandell, J. B., et al. (2003). Breast and ovarian cancer risks due to inherited mutations in brca1 and brca2. *Science*, 302(5645):643–646.

[64] Lancewicki, T. and Aladjem, M. (2014). Multi-target shrinkage estimation for covariance matrices. *IEEE Transactions on Signal Processing*, 62(24):6380–6390.

[65] Law, C. W., Chen, Y., Shi, W., and Smyth, G. K. (2014). voom: Precision weights unlock linear model analysis tools for rna-seq read counts. *Genome biology*, 15(2):R29.

[66] Lawrence, C. B., Allan, S. M., and Rothwell, N. J. (1998). Interleukin-1β and the interleukin-1 receptor antagonist act in the striatum to modify excitotoxic brain damage in the rat. *European Journal of Neuroscience*, 10(3):1188–1195.

[67] Lazovic, J., Basu, A., Lin, H.-W., Rothstein, R. P., Krady, J. K., Smith, M. B., and Levison, S. W. (2005). Neuroinflammation and both cytotoxic and vasogenic edema are reduced in interleukin-1 type 1 receptor-deficient mice conferring neuroprotection. *Stroke*, 36(10):2226–2231.

[68] Leday, G. G. R., de Gunst, M. C. M., Kpogbezan, G. B., van der Vaart, A. W., van Wieringen, W. N., and van de Wiel, M. A. (2017). Gene network reconstruction using global-local shrinkage priors. *Ann. Appl. Statist.*, 11(1):41–68.

[69] Ledoit, O. and Wolf, M. (2003). Improved estimation of the covariance matrix of stock returns with an application to portfolio selection. *Journal of Empirical Finance*, 10(5):603 – 621.

[70] Ledoit, O. and Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. *J. Multivariate Anal.*, 88(2):365–411.

[71] Little, R. J. A. and Rubin, D. B. (1987). Statistical analysis with missing data. *New York: Wiley*.

[72] Ludmir, E. B., Stephens, S. J., Palta, M., Willett, C. G., and Czito, B. G. (2015). Human papillomavirus tumor infection in esophageal squamous cell carcinoma. *Journal of gastrointestinal oncology*, 6(3):287.

[73] Ma, Z. (2013). Sparse principal component analysis and iterative thresholding. *The Annals of Statistics*, 41(2):772–801.

[74] MacKay, D. J. (1995). Probable networks and plausible predictions? a review of practical bayesian methods for supervised neural networks. *Network: computation in neural systems*, 6(3):469–505.

[75] McLachlan, G. and Krishnan, T. (2007). *The EM algorithm and extensions*, volume 382. John Wiley & Sons.

[76] Min, E. J., Chang, C., and Long, Q. (2018). Generalized bayesian factor analysis for integrative clustering with applications to multi-omics data. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 109–119. IEEE.

[77] Murphy, K. P. et al. (2006). Naive bayes classifiers. *University of British Columbia*, 18:60.

[78] Murtagh, F. and Legendre, P. (2014). Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion? *Journal of classification*, 31(3):274–295.

[79] Network, C. G. A. et al. (2012a). Comprehensive molecular characterization of human colon and rectal cancer. *Nature*, 487(7407):330.

[80] Network, C. G. A. et al. (2012b). Comprehensive molecular portraits of human breast tumours. *Nature*, 490(7418):61.

[81] Oba, S., Sato, M.-a., and Ishii, S. (2003a). Prior hyperparameters in bayesian pca. In *Artificial Neural Networks and Neural Information Processing—ICANN/ICONIP 2003*, pages 271–279. Springer.

[82] Oba, S., Sato, M.-a., Takemasa, I., Monden, M., Matsubara, K.-i., and Ishii, S. (2003b). A Bayesian missing value estimation method for gene expression profile data. *Bioinformatics (Oxford, England)*, 19(16):2088–2096.

[83] Opal, S. M., Fisher, C. J., Dhainaut, J.-F. A., Vincent, J.-L., Brase, R., Lowry, S. F., Sadoff, J. C., Slotman, G. J., Levy, H., Balk, R. A., et al. (1997). Confirmatory interleukin-1 receptor antagonist trial in severe sepsis: a phase iii, randomized, doubleblind, placebo-controlled, multicenter trial. *Critical care medicine*, 25(7):1115–1124.

[84] O'Malley, A. J. and Zaslavsky, A. M. (2008). Domain-level covariance analysis for multilevel survey data with structured nonresponse. *Journal of the American Statistical Association*, 103(484):1405–1418.

[85] Parisi, G. (1988). *Statistical field theory*. Addison-Wesley.

[86] Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

[87] Porta, J. M., Verbeek, J. J., and Kröse, B. J. A. (2005). Active Appearance-Based Robot Localization Using Stereo Vision. *Auton. Robots*.

[88] Pourahmadi, M. (2013). *High-dimensional covariance estimation*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., Hoboken, NJ.

[89] Relton, J. K. and Rothwell, N. J. (1992). Interleukin-1 receptor antagonist inhibits ischaemic and excitotoxic neuronal damage in the rat. *Brain research bulletin*, 29(2):243–246.

[90] Ročková, V. and George, E. I. (2016). Fast bayesian factor analysis via automatic rotations to sparsity. *Journal of the American Statistical Association*, 111(516):1608–1622.

[91] Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

[92] Roweis, S. T. (1998). EM Algorithms for PCA and SPCA. In Jordan, M. I., Kearns, M. J., and Solla, S. A., editors, *Advances in Neural Information Processing Systems 10*, pages 626–632. MIT Press.

[93] Sanderson, C. and Curtin, R. (2016). Armadillo: a template-based c++ library for linear algebra. *Journal of Open Source Software*, 1(2):26.

[94] Schäfer, J. and Strimmer, K. (2004). An empirical Bayes approach to inferring large-scale gene association networks. *Bioinformatics (Oxford, England)*, 21(6):754–764.

[95] Schäfer, J., Strimmer, K., et al. (2005). A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical applications in genetics and molecular biology*, 4(1):32.

[96] Schilling, S. and Bock, R. D. (2005). High-dimensional maximum marginal likelihood item factor analysis by adaptive quadrature. *Psychometrika*, 70(3):533–555.

[97] Şenbabaoğlu, Y., Sümer, S. O., Sánchez-Vega, F., Bemis, D., Ciriello, G., Schultz, N., and Sander, C. (2016). A multi-method approach for proteomic network inference in 11 human cancers. *PLoS computational biology*, 12(2):e1004765.

[98] Stacklies, W., Redestig, H., Scholz, M., Walther, D., and Selbig, J. (2007). pcaMethods–a bioconductor package providing PCA methods for incomplete data. *Bioinformatics (Oxford, England)*, 23(9):1164–1167.

[99] Stein, C. (1956). Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. Technical report, Stanford University Stanford United States.

[100] Strimmer, K. (2008). fdrtool: a versatile r package for estimating local and tail area-based false discovery rates. *Bioinformatics*, 24(12):1461–1462.

[101] Tehranian, R., Andell-Jonsson, S., Beni, S. M., Yatsiv, I., Shohami, E., Bartfai, T., Lundkvist, J., and Iverfeldt, K. (2002). Improved recovery and delayed cytokine induction after closed head injury in mice with central overexpression of the secreted isoform of the interleukin-1 receptor antagonist. *Journal of neurotrauma*, 19(8):939–951.

[102] Tipping, M. E. and Bishop, C. M. (1999). Probabilistic Principal Component Analysis. *Journal Of The Royal Statistical Society Series B-Statistical Methodology*, 61(3):611–622.

[103] Touloumis, A. (2015). Nonparametric Stein-type shrinkage covariance matrix estimators in high-dimensional settings. *Comput. Statist. Data Anal.*, 83:251–261.

[104] Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. (2001). Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525.

[105] Van de Wiel, M. A., Leday, G. G., Pardo, L., Rue, H., Van der Vaart, A. W., and Van Wieringen, W. N. (2013). Bayesian analysis of RNA sequencing data by estimating multiple shrinkage priors. *Biostatistics*, 14(1):113–128.

[106] van Wieringen, W. N. and Peeters, C. F. W. (2016). Ridge estimation of inverse covariance matrices from high-dimensional data. *Comput. Statist. Data Anal.*, 103:284–303.

[107] Walboomers, J. M., Jacobs, M. V., Manos, M. M., Bosch, F. X., Kummer, J. A., Shah, K. V., Snijders, P. J., Peto, J., Meijer, C. J., and Muñoz, N. (1999). Human papillomavirus is a necessary cause of invasive cervical cancer worldwide. *The Journal of pathology*, 189(1):12–19.

[108] Warton, D. I. (2008). Penalized normal likelihood and ridge regularization of correlation and covariance matrices. *J. Amer. Statist. Assoc.*, 103(481):340–349.

[109] West, D. B. et al. (2001). *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River.

[110] Whittaker, J. (2009). *Graphical models in applied multivariate statistics*. Wiley Publishing.

[111] Won, J.-H., Lim, J., Kim, S.-J., and Rajaratnam, B. (2013). Condition-number-regularized covariance estimation. *J. R. Stat. Soc. Ser. B. Stat. Methodol.*, 75(3):427–450.

[112] Yang, R. and Berger, J. O. (1994). Estimation of a covariance matrix using the reference prior. *The Annals of Statistics*, pages 1195–1211.

[113] Zhang, Q. (2017). Classification of RNA-Seq data via Gaussian copulas. *Stat*, 6:171–183.

Appendix A

Target-Averaged linear Shrinkage Estimation

A.1 Uncertainty around the empirical Bayes estimate of α

Here, we illustrate the statistical uncertainty surrounding the empirical Bayes estimate α^*, defined as the value of α that maximises the marginal likelihood defined in A.2 for fixed $\boldsymbol{\Delta}$. We generate a data matrix $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ using $\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{0}, 2 * \boldsymbol{I}_{p\times p})$, $p = 200$ and $n = 20$. For the generated data set, we observe that a range of values of α lead to similar marginal likelihood values. Figure A.1 displays the Bayes factor

$$\text{BF}(\alpha) = \frac{\text{p}(\boldsymbol{X}|\alpha^*, \boldsymbol{I}_{p\times p})}{\text{p}(\boldsymbol{X}|\alpha, \boldsymbol{I}_{p\times p})},$$

which quantifies evidence in favour of α^* when compared to alternative values of $\alpha \in (0,1)$.

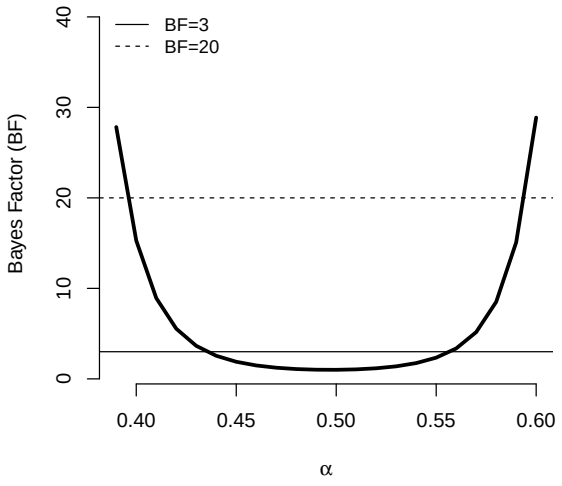


Fig. A.1 Bayes factor quantifying the strength of support for α^* (the empirical Bayes estimate for α) compared to alternative values of $\alpha \in (0,1)$. Horizontal lines correspond to the heuristic rules of Kass and Raftery (1995) for which BF < 3 is "not worth more than a bare mention" and BF < 20 provides "less than strong evidence".

A.2 Marginal likelihood of the Gaussian conjugate model

To derive Equation (2.7), we begin by using the original parametrisation of the Inverse Wishart distribution. The marginal likelihood is found by calculating

$$\mathrm{p}(\boldsymbol{X}|\alpha,\boldsymbol{\Delta}) = \int \mathrm{p}(\boldsymbol{X}|\boldsymbol{\Sigma})\mathrm{p}(\boldsymbol{\Sigma}|\alpha,\boldsymbol{\Delta})\mathrm{d}\boldsymbol{\Sigma} \tag{A.1}$$

$$= \int \frac{1}{(2\pi)^{\frac{np}{2}}}|\boldsymbol{\Sigma}|^{-\frac{n}{2}}\exp\left\{-\frac{1}{2}\mathrm{Tr}\left[n\boldsymbol{S}\boldsymbol{\Sigma}^{-1}\right]\right\}\frac{|\boldsymbol{\Psi}|^{\frac{\nu}{2}}}{2^{\frac{\nu p}{2}}\Gamma_p\left\{\frac{\nu}{2}\right\}}|\boldsymbol{\Sigma}|^{-\frac{1}{2}(\nu+p+1)}$$

$$\exp\left\{-\frac{1}{2}\mathrm{Tr}\left[\boldsymbol{\Psi}\boldsymbol{\Sigma}^{-1}\right]\right\}\mathrm{d}\boldsymbol{\Sigma} \tag{A.2}$$

$$= \frac{1}{(\pi)^{\frac{np}{2}}}\frac{|\boldsymbol{\Psi}|^{\frac{\nu}{2}}}{2^{\frac{(\nu+n)p}{2}}\Gamma_p\left\{\frac{\nu}{2}\right\}}\int |\boldsymbol{\Sigma}|^{-\frac{1}{2}(\nu+n+p+1)}\exp\left\{-\frac{1}{2}\mathrm{Tr}\left[(\boldsymbol{\Psi}+n\boldsymbol{S})\boldsymbol{\Sigma}^{-1}\right]\right\}\mathrm{d}\boldsymbol{\Sigma}. \tag{A.3}$$

The term inside the integral is the kernel of an inverse Wishart distribution with degree of freedom parameter $\nu+n$ and scale matrix $\boldsymbol{\Psi}+n\boldsymbol{S}$. Multiplying the inside of the integral by its normalising constant makes it evaluate to 1. Therefore dividing by the

normalising constant outside of the integral and substituting for $\boldsymbol{\Psi}$ leaves

$$\mathrm{p}(\boldsymbol{X}|\alpha,\boldsymbol{\Delta}) = \frac{1}{(\pi)^{\frac{np}{2}}} \frac{|\boldsymbol{\Psi}|^{\frac{\nu}{2}}}{2^{\frac{(\nu+n)p}{2}}\Gamma_p\left\{\frac{\nu}{2}\right\}} \frac{2^{\frac{(\nu+n)p}{2}}\Gamma_p\left\{\frac{\nu+n}{2}\right\}}{|\boldsymbol{\Psi}+n\boldsymbol{S}|^{\frac{\nu+n}{2}}} \tag{A.4}$$

$$= \frac{1}{(\pi)^{\frac{np}{2}}} \frac{\Gamma_p\left\{\frac{\nu+n}{2}\right\}}{\Gamma_p\left\{\frac{\nu}{2}\right\}} \frac{|\frac{\alpha n}{1-\alpha}\boldsymbol{\Delta}|^{\frac{\nu}{2}}}{|\frac{\alpha n}{1-\alpha}\boldsymbol{\Delta}+n\boldsymbol{S}|^{\frac{\nu+n}{2}}} \tag{A.5}$$

$$= \frac{1}{(\pi)^{\frac{np}{2}}} \frac{\Gamma_p\left\{\frac{\nu+n}{2}\right\}}{\Gamma_p\left\{\frac{\nu}{2}\right\}} \frac{n^{\frac{\nu p}{2}}|\frac{\alpha}{1-\alpha}\boldsymbol{\Delta}|^{\frac{\nu}{2}}}{n^{\frac{(\nu+n)p}{2}}|\frac{\alpha}{1-\alpha}\boldsymbol{\Delta}+\boldsymbol{S}|^{\frac{\nu+n}{2}}} \tag{A.6}$$

$$= \frac{1}{(n\pi)^{\frac{np}{2}}} \frac{\Gamma_p\left\{\frac{\nu+n}{2}\right\}}{\Gamma_p\left\{\frac{\nu}{2}\right\}} \frac{|\frac{\alpha}{1-\alpha}\boldsymbol{\Delta}|^{\frac{\nu}{2}}}{|\frac{\alpha}{1-\alpha}\boldsymbol{\Delta}+\boldsymbol{S}|^{\frac{\nu+n}{2}}}. \tag{A.7}$$

Substituting in for ν then gives the expression

$$\mathrm{p}(\boldsymbol{X}|\alpha,\boldsymbol{\Delta}) = \frac{\Gamma_p\left\{\frac{1}{2}(\frac{n}{1-\alpha}+p+1)\right\}|\frac{\alpha}{1-\alpha}\boldsymbol{\Delta}|^{\frac{\alpha n}{1-\alpha}+p+1}}{(n\pi)^{\frac{np}{2}}\Gamma_p\left\{\frac{1}{2}(\frac{\alpha n}{1-\alpha}+p+1)\right\}|\boldsymbol{S}+\frac{\alpha}{1-\alpha}\boldsymbol{\Delta}|^{\frac{n}{1-\alpha}+p+1}}. \tag{A.8}$$

A.3 Cardinality for the support of α

Our approach assigns a discrete prior distribution with support $\mathcal{A}$ to the shrinkage intensity parameter α. As $0 < \alpha < 1$, a natural support for this prior is an equidistant grid of values within the $(0,1)$ interval. Here, we study the stability of the multi-target estimate for different choices of support. We generate 100 data sets of size $n = 25$ from $\mathcal{N}(\mathbf{0},\mathbf{\Sigma})$ with $p = 100$ and $\mathbf{\Sigma} = 4 * \boldsymbol{I}_{100\times100}$. Subsequently, we compute Equation (2.10) using $\mathcal{D} = \{\boldsymbol{T}_1,\ldots,\boldsymbol{T}_9\}$ (see Table 1 in main text) and $d \in \{0.2,0.1,0.05,0.01,0.005,0.001\}$, where d denotes the distance between consecutive elements of $\mathcal{A}$. Figure A.2 shows the PRIAL of estimator (2.10) as a function of the cardinality $\text{card}_d(\mathcal{A}) = d^{-1} - 1$ of $\mathcal{A}$ and shows that for sufficiently small values of d, i.e. large values of $\text{card}_d(\mathcal{A})$, minimal improvement is obtained beyond $d = 0.01$ ($\text{card}(\mathcal{A}) = 99$).

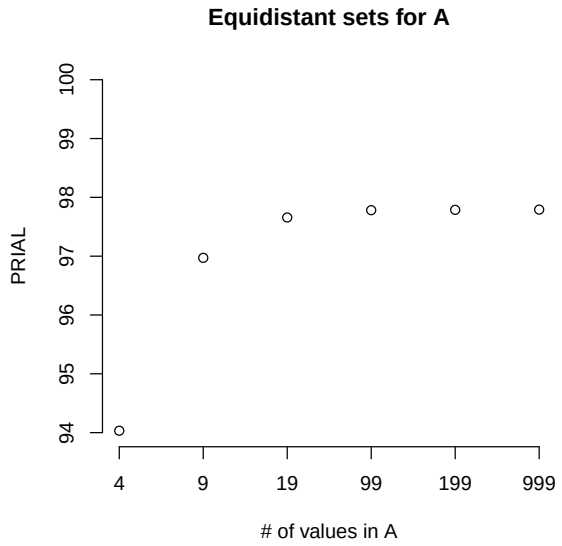
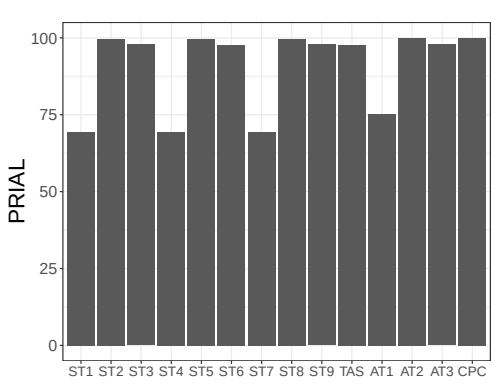


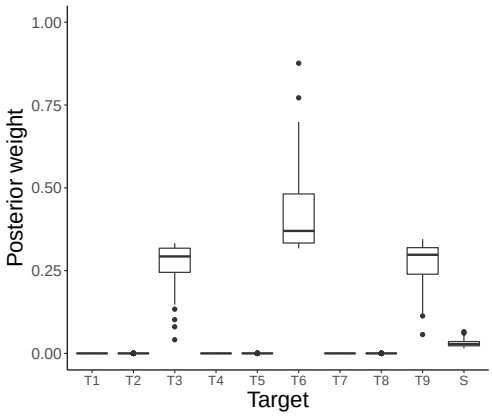
Fig. A.2 PRIAL associated to the TAS estimator ($\mathcal{D} = \{\boldsymbol{T}_1,\ldots,\boldsymbol{T}_9\}$, see Table 2.1) across different cardinalities of $\mathcal{A}$. Results are associated to the simulation setup described in Section A.3.

A.4 Model-based simulation: additional results

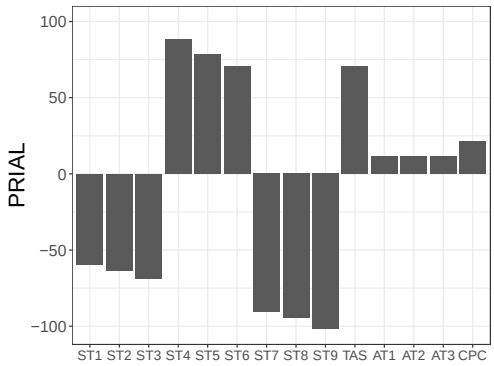
Figures A.3, A.4, A.5 and A.6 complement Figures 2.1 and 2.2 in Section 2.6 by providing results for $n \in \{50,75\}$.



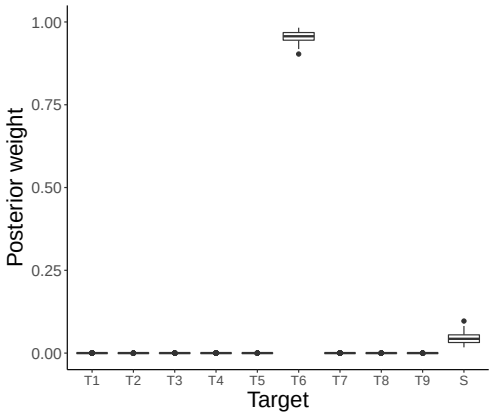
(a) Scenario 1: PRIAL



(b) Scenario 1: target-specific posterior weights

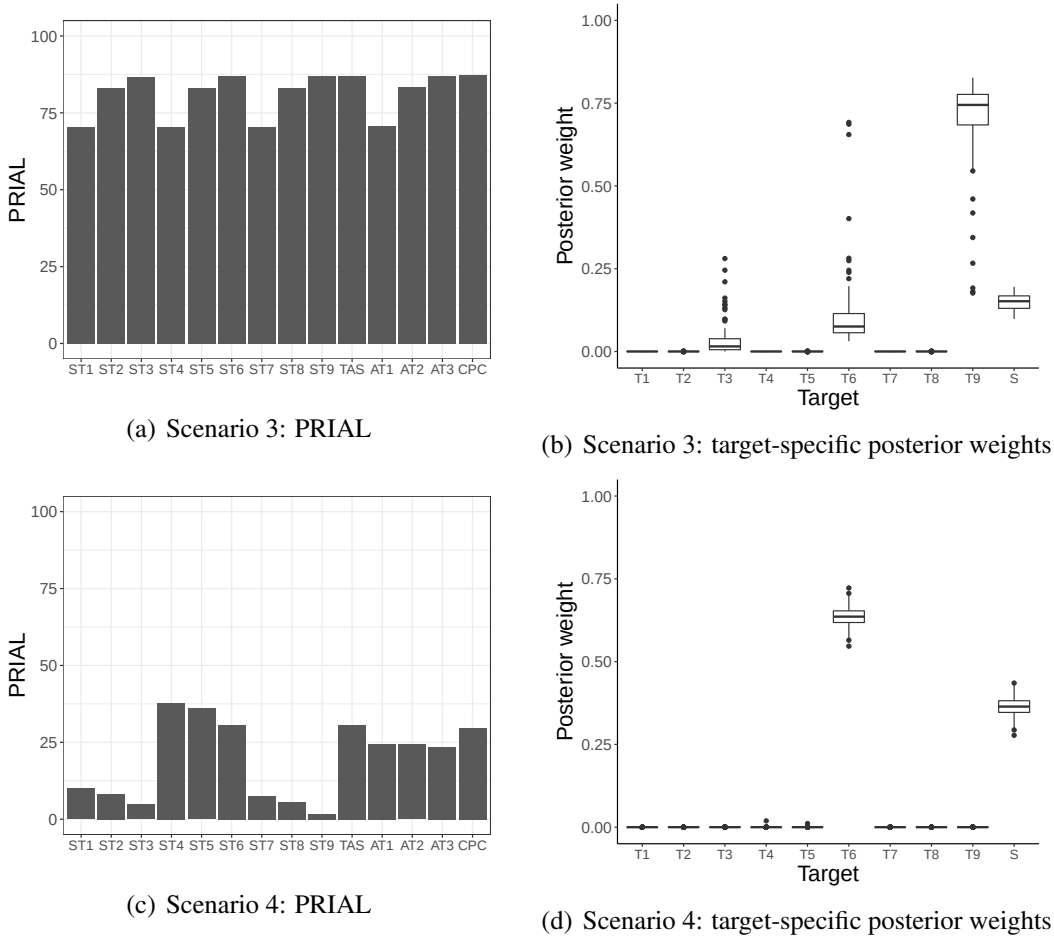


(c) Scenario 2: PRIAL



(d) Scenario 2: target-specific posterior weights

Fig. A.3 Simulation results for scenarios 1 and 2 when $n = 50$. Barplots display the PRIAL for each estimator and boxplots display target-specific posterior weights (see Equation (2.11)) of the TAS estimator. ST1, ..., ST9 refer to the nine STS estimators, TAS to the estimator in Equation (2.10), AT1, ..., AT3 to the three estimators of Touloumis [103] and CPC to the estimator of Schäfer et al. [95].



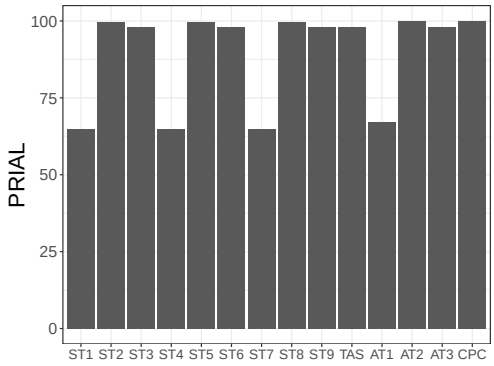
(a) Scenario 3: PRIAL

(b) Scenario 3: target-specific posterior weights

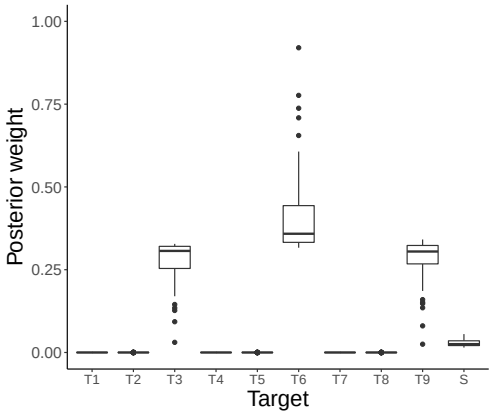
(c) Scenario 4: PRIAL

(d) Scenario 4: target-specific posterior weights

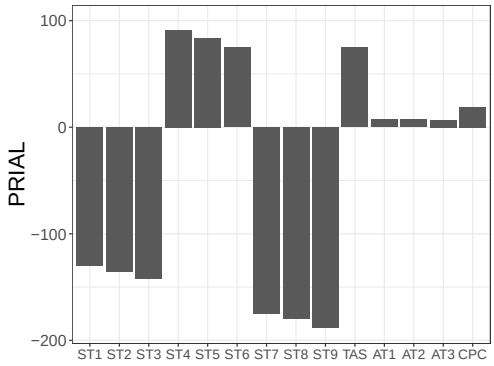
Fig. A.4 Simulation results for scenarios 3 and 4 when $n = 50$. Barplots display the PRIAL for each estimator and boxplots display target-specific posterior weights (see Equation (2.11)) of the TAS estimator. ST1, ..., ST9 refer to the nine STS estimators, TAS to the estimator in Equation (2.10), AT1, ..., AT3 to the three estimators of Touloumis [103] and CPC to the estimator of Schäfer et al. [95].



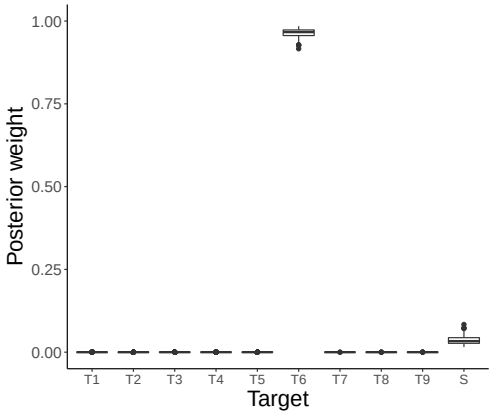
(a) Scenario 1: PRIAL



(b) Scenario 1: target-specific posterior weights

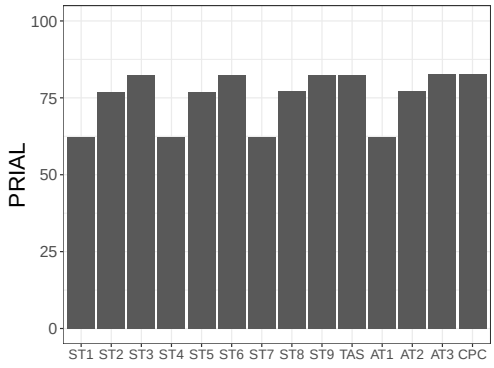


(c) Scenario 2: PRIAL

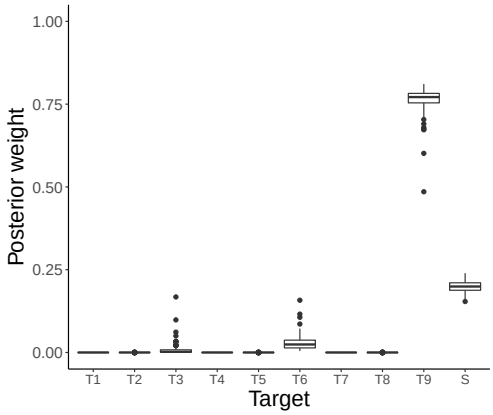


(d) Scenario 2: target-specific posterior weights

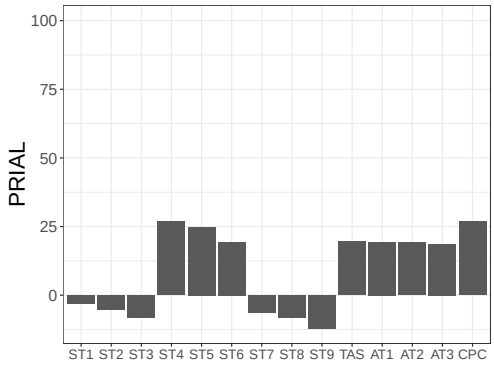
Fig. A.5 Simulation results for scenarios 1 and 2 when $n = 75$. Barplots display the PRIAL for each estimator and boxplots display target-specific posterior weights (see Equation (2.11)) of the TAS estimator. ST1, ..., ST9 refer to the nine STS estimators, TAS to the estimator in Equation (2.10), AT1, ..., AT3 to the three estimators of Touloumis [103] and CPC to the estimator of Schäfer et al. [95].



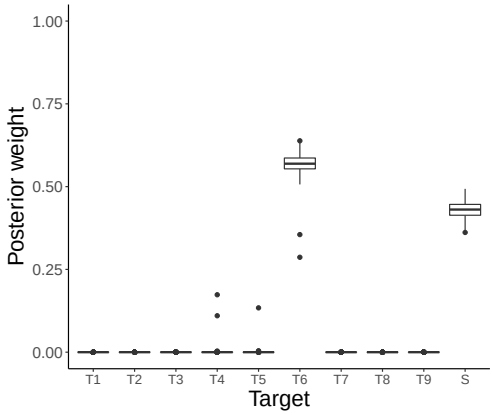
(a) Scenario 3: PRIAL



(b) Scenario 3: target-specific posterior weights

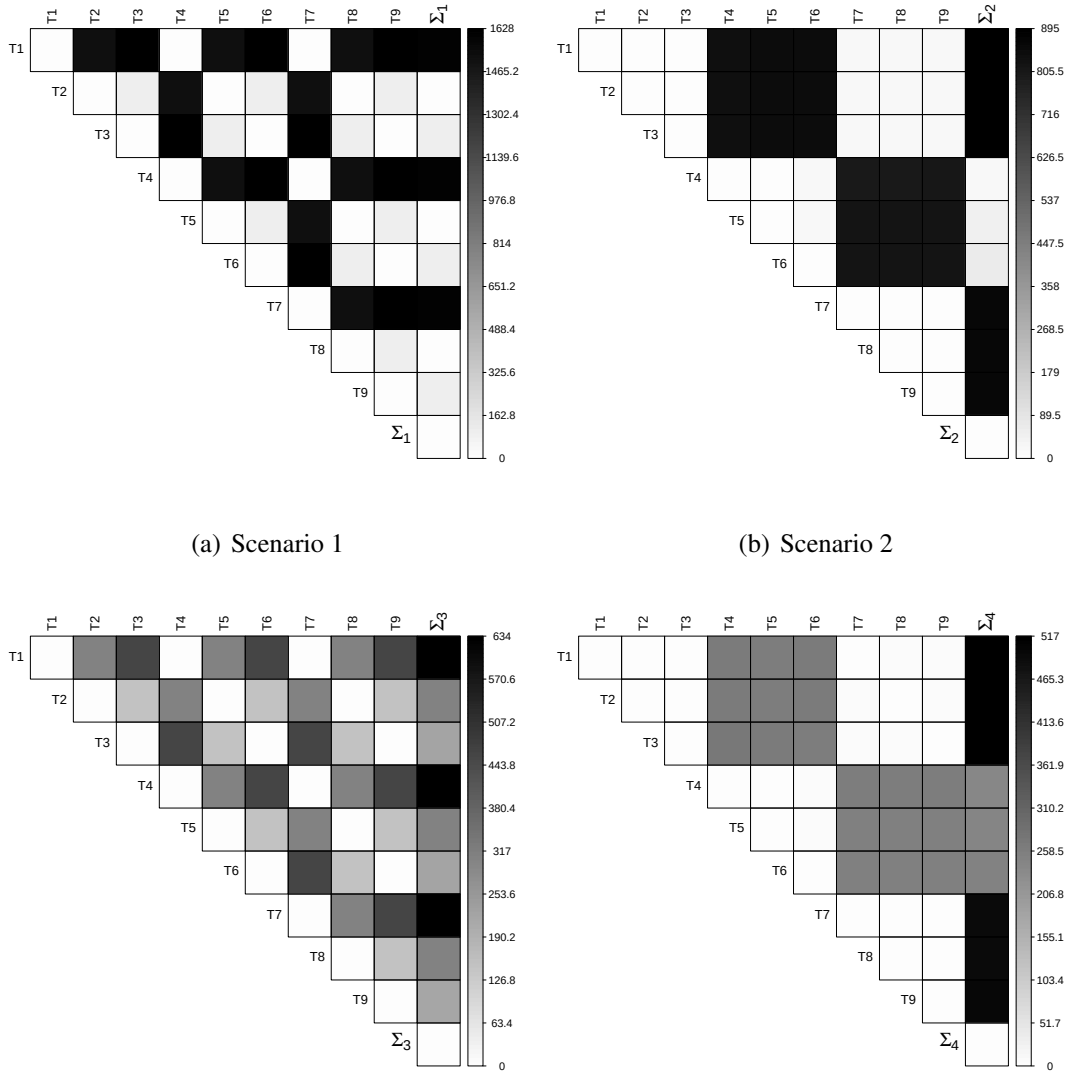


(c) Scenario 4: PRIAL



(d) Scenario 4: target-specific posterior weights

Fig. A.6 Simulation results for scenarios 3 and 4 when $n = 75$. Barplots display the PRIAL for each estimator and boxplots display target-specific posterior weights (see Equation (2.11)) of the TAS estimator. ST1, ..., ST9 refer to the nine STS estimators, TAS to the estimator in Equation (2.10), AT1, ..., AT3 to the three estimators of Touloumis [103] and CPC to the estimator of Schäfer et al. [95].



(a) Scenario 1

(b) Scenario 2

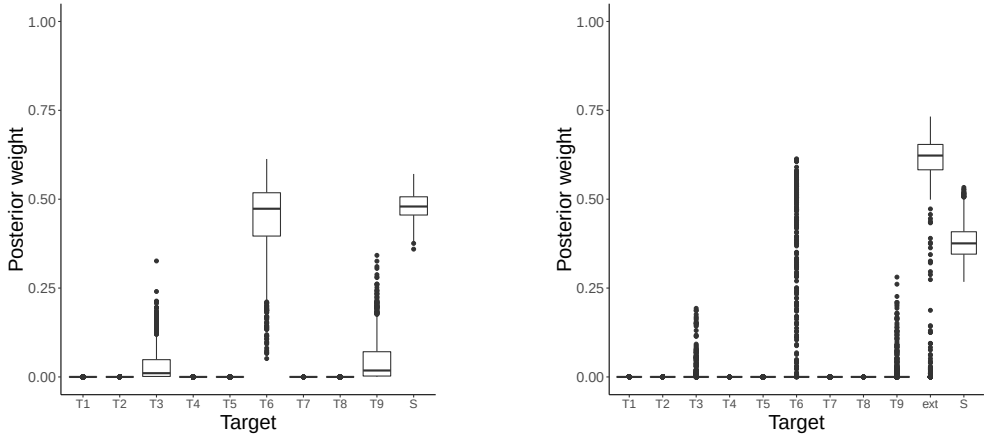
(c) Scenario 3

(d) Scenario 4

Fig. A.7 Heatmaps displaying the average Frobenius norm (over the 100 simulated data sets) between all pairs of shrinkage targets in Table 2.1 for simulation scenarios 1, 2, 3 and 4 when $n = 50$ (results are omitted for $n \in \{25,75\}$ as they are identical). The true covariance matrices $\Sigma_1,\ldots,\Sigma_4$ were also included in the comparison. Light (dark) colors indicate that the shrinkage targets are (dis-)similar.

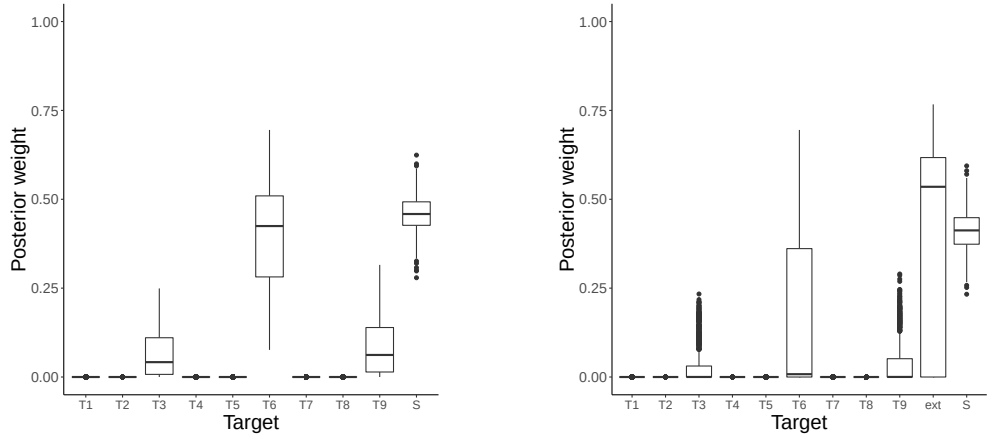
A.5 predictive validation simulation strategy

Figure A.8 complements Figure 2.5 by providing results for $n = p/4$ and $n = 3p/4$.



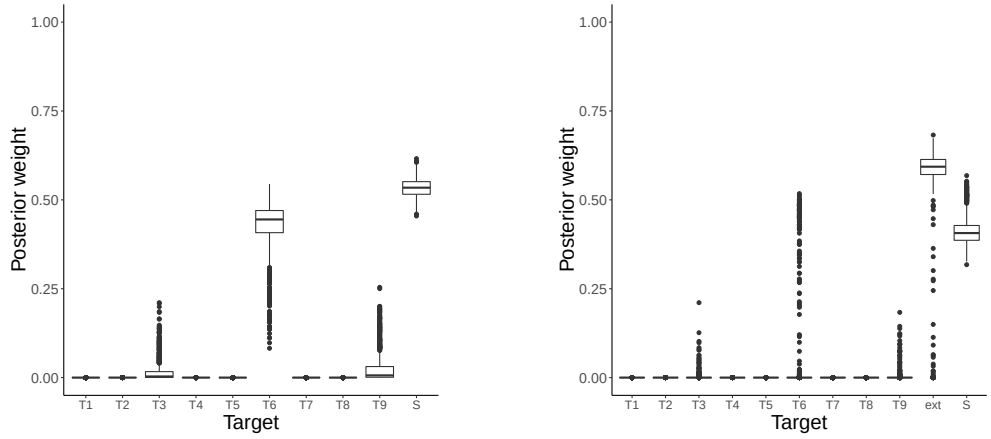
(a) $\hat{\mathbf{\Sigma}}_{\text{TAS}}$ $(n = p/2)$ (b) $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$ $(n = p/2)$

Fig. A.8 Target-specific posterior weights (see Equation (2.11)) obtained for estimators $\hat{\mathbf{\Sigma}}_{\text{TAS}}$ and $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$ across the 1,000 random data partitions of the breast cancer data set when $n \in \{p/2\}$. The target "ext" in $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$ stands for the shrinkage target $\hat{\mathbf{\Sigma}}_{\text{ext}}$ estimated from external data.



(a) $\hat{\boldsymbol{\Sigma}}_{\text{TAS}}$ ($n = p/4$)

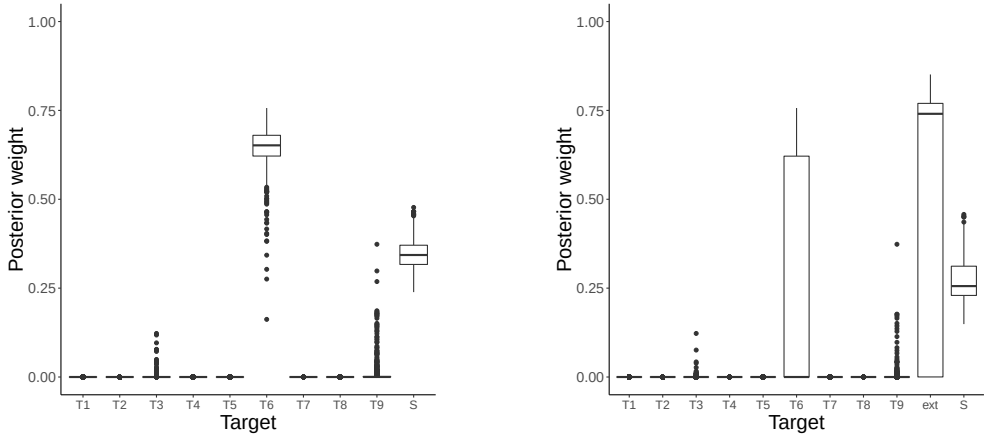
(b) $\hat{\boldsymbol{\Sigma}}_{\text{TAS-info}}$ ($n = p/4$)



(c) $\hat{\boldsymbol{\Sigma}}_{\text{TAS}}$ ($n = 3p/4$)

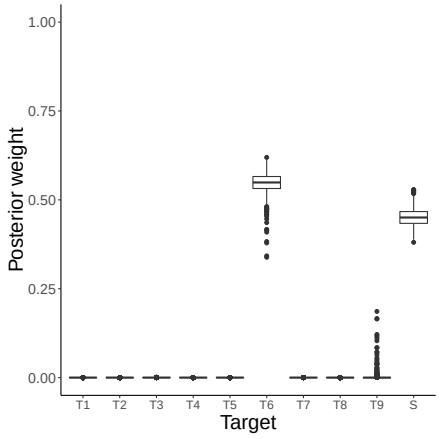
(d) $\hat{\boldsymbol{\Sigma}}_{\text{TAS-info}}$ ($n = 3p/4$)

Fig. A.9 Target-specific posterior weights (see Equation (2.11)) obtained for estimators $\hat{\boldsymbol{\Sigma}}_{\text{TAS}}$ and $\hat{\boldsymbol{\Sigma}}_{\text{TAS-info}}$ across the 1,000 random data partitions of the breast cancer data set when $n \in \{p/4, 3p/4\}$. The target "ext" in $\hat{\boldsymbol{\Sigma}}_{\text{TAS-info}}$ stands for the shrinkage target $\hat{\boldsymbol{\Sigma}}_{\text{ext}}$ estimated from external data.

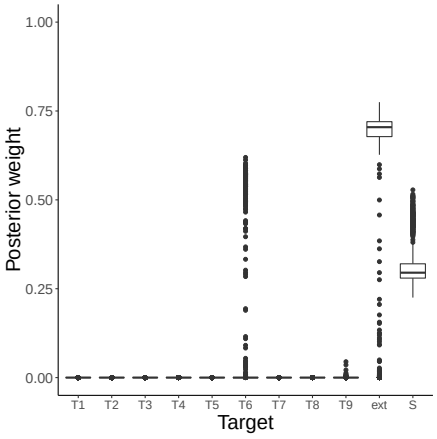


(a) $\hat{\mathbf{\Sigma}}_{\text{TAS}}$ $(n = p/4)$

(b) $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$ $(n = p/4)$

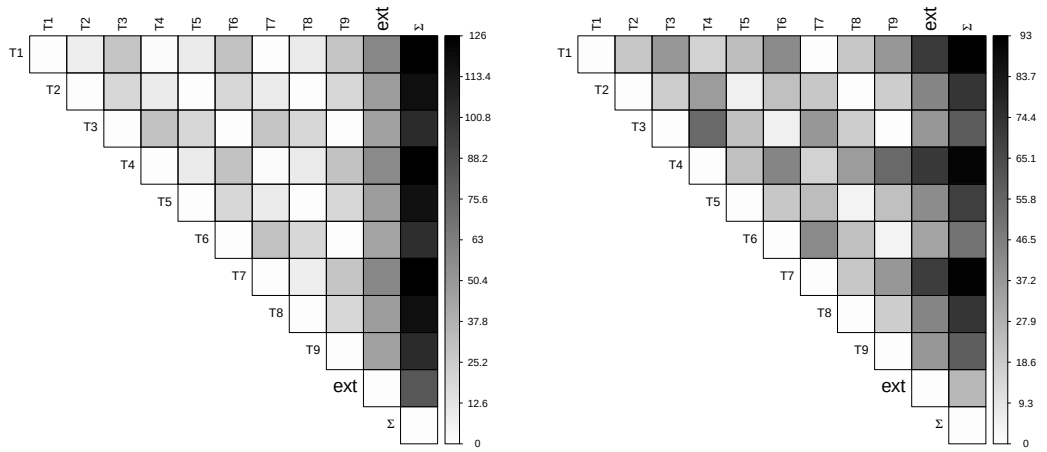


(c) $\hat{\mathbf{\Sigma}}_{\text{TAS}}$ $(n = 3p/4)$



(d) $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$ $(n = 3p/4)$

Fig. A.10 Target-specific posterior weights (see Equation (2.11)) obtained for estimators $\hat{\mathbf{\Sigma}}_{\text{TAS}}$ and $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$ across the 1,000 random data partitions of the ovarian cancer data set when $n \in \{p/4, 3p/4\}$. The target "ext" in $\hat{\mathbf{\Sigma}}_{\text{TAS-info}}$ stands for the shrinkage target $\hat{\mathbf{\Sigma}}_{\text{ext}}$ estimated from external data.



(a) Breast cancer ($n = p/2$)

(b) Ovarian cancer ($n = p/2$)

Fig. A.11 Heatmaps displaying the average Frobenius norm (over the 1000 data partitions) between all pairs of shrinkage targets in Table 2.1 for Breast and Ovarian cancer datasets when $n = p/2$ (results are omitted for $n \in \{p/4, 3p/4\}$ as they are identical). The true covariance matrix Σ for each cancer type is also included in the comparison. Light (dark) colors indicate that the shrinkage targets are (dis-)similar.

A.6 Assumption of normality

Figure A.12 provides normal Quantile-Quantile plots for the expression levels of four different genes in two different cancer data sets from TCGA. This provides strong evidence to suggest that the Gaussian assumption does not hold (even for individual genes).

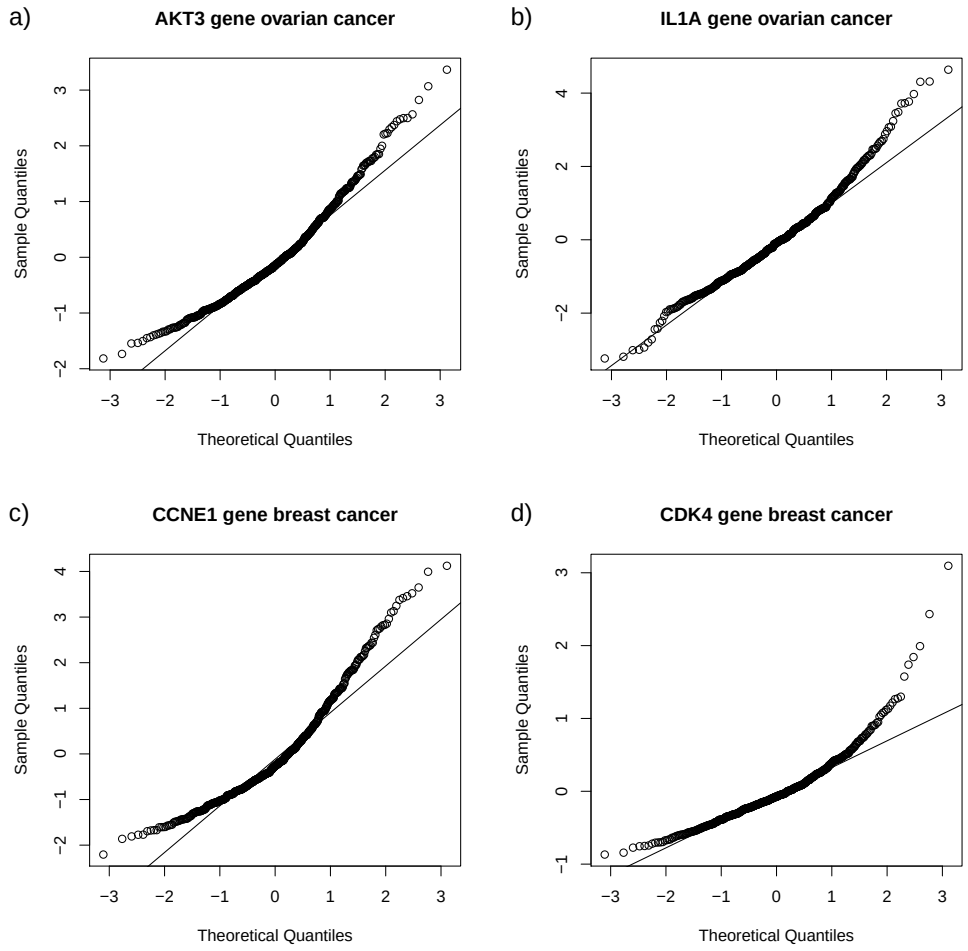


Fig. A.12 Normal Quantile-Quantile plots for two genes from TCGA datasets. Sub-figures (a) and (b) show the departure from normality for genes AKT3 and IL1A in the ovarian cancer data whereas, sub-figures (c) and (d) show the departure from normality for genes CCNE1 and CDK4 in the breast cancer data.

A.7 The PANCAN32 data set

	Cancer type	TCPA acronym	*n*
1	Adrenocortical carcinoma	ACC	46
2	Bladder urothelial carcinoma	BLCA	344
3	Breast invasive carcinoma	BRCA	874
4	Cervical squamous cell carcinoma and endocervical adenocarcinoma	CESC	171
5	Cholangiocarcinoma	CHOL	30
6	Colon adenocarcinoma	COAD	357
7	Lymphoid neoplasm niffuse large B-cell lymphoma	DLBC	33
8	Esophageal carcinoma	ESCA	126
9	Glioblastoma multiforme	GBM	205
10	Head and neck squamous cell carcinoma	HNSC	346
11	Kidney chromophobe	KICH	63
12	Kidney renal clear cell carcinoma	KIRC	445
13	Kidney renal papillary cell carcinoma	KIRP	208
14	Brain lower grade glioma	LGG	427
15	Liver hepatocellular carcinoma	LIHC	184
16	Lung adenocarcinoma	LUAD	362
17	Lung squamous cell carcinoma	LUSC	325
18	Mesothelioma	MESO	61
19	Ovarian serous cystadenocarcinoma	OV	411
20	Pancreatic adenocarcinoma	PAAD	105
21	Pheochromocytoma and paraganglioma	PCPG	80
22	Prostate adenocarcinoma	PRAD	351
23	Rectum adenocarcinoma	READ	130
24	Sarcoma	SARC	221
25	Skin cutaneous melanoma	SKCM	353
26	Stomach adenocarcinoma	STAD	392
27	Testicular germ cell tumors	TGCT	118
28	Thyroid carcinoma	THCA	372
29	Thymoma	THYM	90
30	Uterine corpus endometrial carcinoma	UCEC	404
31	Uterine carcinosarcoma	UCS	48
32	Uveal melanoma	UVM	12

Table A.1 Cancer types and number of samples in the PANCAN32 protein expression data set from The Cancer Proteome Atlas.

Appendix B

Covariance estimation through Probabilistic Principal Component Analysis

B.1 EM algorithm in PPCA without missing values - derivation

The equation for the joint log likelihood of the observed and latent variables is given by

$$\ln \mathrm{p}(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\mu},\boldsymbol{W},\sigma^2) = \sum_{j=1}^{n} \Big(\ln \mathrm{p}(\boldsymbol{x}_j|z_j,\boldsymbol{\mu},\boldsymbol{W},\sigma^2) + \ln \mathrm{p}(z_j) \Big). \tag{B.1}$$

From the assumption that $z_j \sim \mathcal{N}(\boldsymbol{0},\boldsymbol{I}_{q\times q})$, we have

$$\ln \mathrm{p}(z_j) = -\frac{q}{2}\ln(2\pi) - \frac{1}{2} z_j^\top z_j, \tag{B.2}$$

while from $\boldsymbol{x}_j|z_j,\boldsymbol{\mu},\boldsymbol{W},\sigma^2 \sim \mathcal{N}(\boldsymbol{W}z_j+\boldsymbol{\mu},\sigma^2 I)$ we have

$$\begin{aligned} \ln \mathrm{p}(\boldsymbol{x}_j|z_j,\boldsymbol{\mu},\boldsymbol{W},\sigma^2) &= -\frac{p}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}\left((\boldsymbol{x}_j-\boldsymbol{\mu}) - \boldsymbol{W}z_j\right)^\top \left((\boldsymbol{x}_j-\boldsymbol{\mu}) - \boldsymbol{W}z_j\right) \\ &= -\frac{p}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}\Big((\boldsymbol{x}_j-\boldsymbol{\mu})^\top(\boldsymbol{x}_j-\boldsymbol{\mu}) + z_j^\top \boldsymbol{W}^\top \boldsymbol{W} z_j \\ &\quad - z_j^\top \boldsymbol{W}^\top(\boldsymbol{x}_j-\boldsymbol{\mu}) - (\boldsymbol{x}_j-\boldsymbol{\mu})^\top \boldsymbol{W} z_j \Big). \end{aligned} \tag{B.3}$$

Note that $(\boldsymbol{x}_j-\boldsymbol{\mu})^\top \boldsymbol{W} z_j$ is a scalar, so $(\boldsymbol{x}_j-\boldsymbol{\mu})^\top \boldsymbol{W} z_j = ((\boldsymbol{x}_j-\boldsymbol{\mu})^\top \boldsymbol{W} z_j)^\top = z_j^\top \boldsymbol{W}^\top(\boldsymbol{x}_j - \boldsymbol{\mu})$. Moreover, $z_j^\top \boldsymbol{W}^\top \boldsymbol{W} z_j = \mathrm{Tr}(z_j^\top \boldsymbol{W}^\top \boldsymbol{W} z_j) = \mathrm{Tr}(z_j z_j^\top \boldsymbol{W}^\top \boldsymbol{W})$, where we have used the

cyclic property of the trace in the final equality. Equation (B.3) then becomes

$$\ln \mathrm{p}(\boldsymbol{x}_j|\boldsymbol{z}_j,\boldsymbol{\mu},\boldsymbol{W},\sigma^2) = -\frac{p}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}\Big(\|\boldsymbol{x}_j-\boldsymbol{\mu}\|^2 + \mathrm{Tr}(\boldsymbol{z}_j\boldsymbol{z}_j^\top\boldsymbol{W}^\top\boldsymbol{W}) - 2\boldsymbol{z}_j^\top\boldsymbol{W}^\top(\boldsymbol{x}_j-\boldsymbol{\mu})\Big), \tag{B.4}$$

where $\|\cdot\|$ denotes the Euclidean norm. Substituting Equations (B.2) and (B.3) into Equation (B.1) then gives

$$\begin{aligned}\ln \mathrm{p}(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\mu},\boldsymbol{W},\sigma^2) = \sum_{j=1}^{n} &-\frac{p}{2}\ln(2\pi\sigma^2) - \frac{q}{2}\ln(2\pi) - \frac{1}{2}\mathrm{Tr}(\boldsymbol{z}_j\boldsymbol{z}_j^\top) \\ &- \frac{1}{2\sigma^2}\Big(\|\boldsymbol{x}_j-\boldsymbol{\mu}\|^2 + \mathrm{Tr}(\boldsymbol{z}_j\boldsymbol{z}_j^\top\boldsymbol{W}^\top\boldsymbol{W}) \\ &-2\boldsymbol{z}_j^\top\boldsymbol{W}^\top(\boldsymbol{x}_j-\boldsymbol{\mu})\Big).\end{aligned} \tag{B.5}$$

B.1.1 E step

In the E step, we find the expectation of the complete-date log likelihood with respect to the posterior distribution of the latent factors (given the most recently estimated values for the parameters, $\boldsymbol{\mu},\boldsymbol{W}$ and σ^2). This is given by

$$\begin{aligned}\mathbb{E}\left[\ln \mathrm{p}(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\mu},\boldsymbol{W},\sigma^2)\right] = \sum_{j=1}^{n} &-\frac{p}{2}\ln(2\pi\sigma^2) - \frac{q}{2}\ln(2\pi) - \frac{1}{2}\mathrm{Tr}\left(\mathbb{E}\left[\boldsymbol{z}_j\boldsymbol{z}_j^\top\right]\right) \\ &- \frac{1}{2\sigma^2}\Big(\|\boldsymbol{x}_j-\boldsymbol{\mu}\|^2 + \mathrm{Tr}\left(\mathbb{E}\left[\boldsymbol{z}_j\boldsymbol{z}_j^\top\right]\boldsymbol{W}^\top\boldsymbol{W}\right) \\ &-2\mathbb{E}\left[\boldsymbol{z}_j\right]^\top\boldsymbol{W}^\top(\boldsymbol{x}_j-\boldsymbol{\mu})\Big),\end{aligned} \tag{B.6}$$

from which it is clear that we require expressions for $\mathbb{E}\left[\boldsymbol{z}_j\right]$ and $\mathbb{E}\left[\boldsymbol{z}_j\boldsymbol{z}_j^\top\right]$. Recall that $\boldsymbol{z}_j \sim \mathcal{N}(\boldsymbol{0},\boldsymbol{I}_{q\times q})$ and $\boldsymbol{x}_j|\boldsymbol{z}_j \sim \mathcal{N}(\boldsymbol{W}\boldsymbol{z}_j+\boldsymbol{\mu},\sigma^2\boldsymbol{I}_{p\times p})$, and so it follows that

$$\boldsymbol{z}_j|\boldsymbol{x}_j \sim \mathcal{N}\left(\boldsymbol{M}^{-1}\boldsymbol{W}^\top(\boldsymbol{x}_j-\boldsymbol{\mu}),\sigma^2\boldsymbol{M}^{-1}\right), \tag{B.7}$$

where $\boldsymbol{M} = \boldsymbol{W}^\top\boldsymbol{W} + \sigma^2\boldsymbol{I}_{q\times q}$ and from Equation (B.7) it follows that

$$\mathbb{E}\left[\boldsymbol{z}_j\right] = \boldsymbol{M}^{-1}\boldsymbol{W}^\top(\boldsymbol{x}_j-\boldsymbol{\mu}) \tag{B.8}$$

$$\mathbb{E}\left[\boldsymbol{z}_j\boldsymbol{z}_j^\top\right] = \sigma^2\boldsymbol{M}^{-1} + \mathbb{E}\left[\boldsymbol{z}_j\right]\mathbb{E}[\boldsymbol{z}_j]^\top. \tag{B.9}$$

B.1.2 M step

In the M step, we maximise Equation (B.6) with respect to the parameters. Setting to zero the derivative of Equation (B.6) with respect to $\boldsymbol{\mu}$ gives the following expression:

$$0 = \frac{\partial}{\partial \boldsymbol{\mu}} \mathbb{E}[\ln \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} | \boldsymbol{\mu}, \boldsymbol{W}, \sigma^2)] \tag{B.10}$$

$$= -\frac{1}{2\sigma^2} \frac{\partial}{\partial \boldsymbol{\mu}} \sum_{i=1}^{n} \left((\boldsymbol{x}_j - \boldsymbol{\mu})^\top (\boldsymbol{x}_j - \boldsymbol{\mu}) + 2\mathbb{E}[z_j]^\top \boldsymbol{W}^\top \boldsymbol{\mu} \right) \tag{B.11}$$

$$= -\frac{1}{2\sigma^2} \sum_{i=1}^{n} \left(-2(\boldsymbol{x}_j - \boldsymbol{\mu}) + 2\boldsymbol{W}\mathbb{E}[z_j] \right). \tag{B.12}$$

Multiplying through by σ^2 and solving for $\boldsymbol{\mu}$ gives

$$\boldsymbol{\mu}_{\text{new}} = \frac{1}{n} \sum_{i=1}^{n} \left(\boldsymbol{x}_j - \boldsymbol{W}\mathbb{E}[z_j] \right). \tag{B.13}$$

Setting to zero the derivative of Equation (B.6) with respect to $\boldsymbol{W}$ gives the following expression:

$$0 = \frac{\partial}{\partial \boldsymbol{W}} \mathbb{E}[\ln \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} | \boldsymbol{\mu}, \boldsymbol{W}, \sigma^2)] \tag{B.14}$$

$$= -\sum_{i=1}^{n} \left(\frac{\partial}{\partial \boldsymbol{W}} \mathrm{Tr}\left(\mathbb{E}[z_j z_j^\top] \boldsymbol{W}^\top \boldsymbol{W} \right) - 2 \frac{\partial}{\partial \boldsymbol{W}} \mathbb{E}[z_j]^\top \boldsymbol{W}^\top (\boldsymbol{x}_j - \boldsymbol{\mu}) \right). \tag{B.15}$$

Using the results $\frac{\partial}{\partial \boldsymbol{W}} \mathrm{Tr}(\boldsymbol{B}\boldsymbol{W}^\top \boldsymbol{W}) = \boldsymbol{W}(\boldsymbol{B} + \boldsymbol{B}^\top)$ and $\frac{\partial}{\partial \boldsymbol{W}} (\boldsymbol{a}^\top \boldsymbol{W}^\top \boldsymbol{b}) = \boldsymbol{b}\boldsymbol{a}^\top$, Equation (B.15) becomes

$$0 = -\sum_{i=1}^{n} \left(\boldsymbol{W} \left(\mathbb{E}[z_j z_j^\top] + \mathbb{E}[z_j z_j^\top]^\top \right) - 2(\boldsymbol{x}_j - \boldsymbol{\mu}) \mathbb{E}[z_j]^\top \right) \tag{B.16}$$

$$= 2 \sum_{i=1}^{n} (\boldsymbol{x}_j - \boldsymbol{\mu}) \mathbb{E}[z_j]^\top - 2\boldsymbol{W} \sum_{i=1}^{n} \mathbb{E}[z_j z_j^\top], \tag{B.17}$$

which we may rearrange to give the update equation for $\boldsymbol{W}$ as

$$\boldsymbol{W}_{\text{new}} = \left(\sum_{i=1}^{n} (\boldsymbol{x}_j - \boldsymbol{\mu}) \mathbb{E}[z_j]^\top \right) \left(\sum_{i=1}^{n} \mathbb{E}[z_j z_j^\top] \right)^{-1}. \tag{B.18}$$

Setting to zero the derivative of Equation (B.6) with respect to σ^2 gives the following expression:

$$0 = \frac{\partial}{\partial \sigma^2}\mathbb{E}[\ln \mathrm{p}(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\mu},\boldsymbol{W},\sigma^2)] \tag{B.19}$$

$$= -\sum_{i=1}^{n} \frac{p}{2\sigma^2} - \frac{1}{2\sigma^4}\left(||\mathbf{x}_j - \boldsymbol{\mu}||^2 + \mathrm{Tr}\left(\mathbb{E}[z_j z_j^\top]\boldsymbol{W}^\top \boldsymbol{W}\right)\right.$$
$$\left. -2\mathbb{E}[z_j]^\top \boldsymbol{W}^\top(\boldsymbol{x}_j - \boldsymbol{\mu})\right). \tag{B.20}$$

After multiplying through by $2\sigma^4$ and rearranging, we obtain

$$np\sigma^2 = \sum_{i=1}^{n}\left(||\boldsymbol{x}_j - \boldsymbol{\mu}||^2 + \mathrm{Tr}\left(\mathbb{E}[z_j z_j^\top]\boldsymbol{W}^\top \boldsymbol{W}\right) - 2\mathbb{E}[z_j]^\top \boldsymbol{W}^\top(\boldsymbol{x}_j - \boldsymbol{\mu})\right), \tag{B.21}$$

from which we may straightforwardly obtain the update equation for σ^2 as

$$\sigma^2_{\text{new}} = \frac{1}{np}\sum_{i=1}^{n} ||\boldsymbol{x}_j - \boldsymbol{\mu}||^2 + \mathrm{Tr}(\mathbb{E}[z_j z_j^\top]\boldsymbol{W}_{\text{new}}^\top \boldsymbol{W}_{\text{new}})$$
$$- 2\mathbb{E}[z_j]^\top \boldsymbol{W}_{\text{new}}^\top(\boldsymbol{x}_j - \boldsymbol{\mu}). \tag{B.22}$$

B.2 EM algorithm 1 - derivation

Collecting the p-dimensional data vectors as $\boldsymbol{X} = (\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n)$, we begin by assuming that their shared mean vector $\boldsymbol{\mu} = \overline{\boldsymbol{x}} = \frac{1}{n}\sum_{j=1}^{n}\boldsymbol{x}_j$ has been subtracted, i.e. that their mean is known and equal to its sample estimate, which is then subtracted as a preprocessing step. This means that the conditional distribution is now $\boldsymbol{x}_j|z_j \sim \mathcal{N}(\boldsymbol{W}z_j, \sigma^2\boldsymbol{I}_{p\times p})$, with the $\boldsymbol{\mu}$ having been removed.

B.2.1 Handling of missing values

Assuming now that missing values are present, each $\boldsymbol{x}_j$ can be partitioned (without loss of generality) as follows:

$$\boldsymbol{x}_j = \begin{bmatrix} \boldsymbol{x}_j^{(O_j)} \\ \boldsymbol{x}_j^{(M_j)} \end{bmatrix}, \tag{B.23}$$

where the notation M_j is use for all indices not in O_j, i.e. the missing indices as the complement of the observed ones. Similarly, the loadings matrix can be partitoned as

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}^{(O_j)} \\ \boldsymbol{W}^{(M_j)} \end{bmatrix}, \tag{B.24}$$

where we note that this defines multiple partitions of the matrix $\boldsymbol{W}$, one for each data vector $\boldsymbol{x}_j$ corresponding to its (not necessarily) unique observed and missing indices. More explicitly, for the matrices $\boldsymbol{W}^{(O_j)}$ and $\boldsymbol{W}^{(M_j)}$, we have retained only those rows of $\boldsymbol{W}$ which correspond to $\boldsymbol{x}_j^{(O_j)}$ and $\boldsymbol{x}_j^{(M_j)}$, respectively. The entries $w_{ij}^{(O_j)} = \{w_{ij} | i = 1,\ldots,|O_j|, j = 1,\ldots,q\}$ and $w_{ij}^{(M_j)} = \{w_{ij} | i = |O_j|+1,\ldots,p, j = 1,\ldots,q\}$ so that the matrices $\boldsymbol{W}^{(O_j)}$ and $\boldsymbol{W}^{(M_j)}$ are of dimension $|O_j| \times q$ and $(p-|O_j|) \times q$, respectively.

The conditional distribution can now be written in a partitioned form as

$$\begin{bmatrix} \boldsymbol{x}_j^{(O_j)} \\ \boldsymbol{x}_j^{(M_j)} \end{bmatrix} | z \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{W}^{(O_j)} \boldsymbol{z}_j \\ \boldsymbol{W}^{(M_j)} \boldsymbol{z}_j \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}^{(O_j,O_j)} & \boldsymbol{\Sigma}^{(O_j,M_j)} \\ \boldsymbol{\Sigma}^{(M_j,O_j)} & \boldsymbol{\Sigma}^{(M_j,M_j)} \end{bmatrix} \right). \tag{B.25}$$

From our knowledge of the conditional distribution, we see that $\boldsymbol{\Sigma}^{(O_j,M_j)} = \boldsymbol{0}_{|O_j| \times (p-|O_j|)}$, $\boldsymbol{\Sigma}^{(M_j,O_j)} = \boldsymbol{0}_{(p-|O_j|) \times |O_j|}$, $\boldsymbol{\Sigma}^{(O_j,O_j)} = \sigma^2 \boldsymbol{I}_{|O_j| \times |O_j|}$, and $\boldsymbol{\Sigma}^{(M_j,M_j)} = \sigma^2 \boldsymbol{I}_{(p-|O_j|) \times (p-|O_j|)}$. From Equation (B.25), it can be seen that

$$\boldsymbol{x}_j^{(O_j)} | \boldsymbol{z}_j \sim \mathcal{N}\left(\boldsymbol{W}^{(O_j)} \boldsymbol{z}_j, \sigma^2 \boldsymbol{I}_{|O_j| \times |O_j|} \right) \tag{B.26}$$

and

$$\boldsymbol{x}_j^{(M_j)} | \boldsymbol{z}_j \sim \mathcal{N}\left(\boldsymbol{W}^{(M_j)} \boldsymbol{z}_j, \sigma^2 \boldsymbol{I}_{(p-|O_j|) \times (p-|O_j|)} \right). \tag{B.27}$$

It is worth noting that that $\boldsymbol{x}_j^{(O_j)}$ and $\boldsymbol{x}_j^{(M_j)}$ are assumed to be conditionally independent given the value of $\boldsymbol{z}_j$ so that $\mathrm{p}\left(\boldsymbol{x}_j^{(O_j)} | \boldsymbol{x}_j^{(M_j)}, \boldsymbol{z}_j, \boldsymbol{W} \right) = \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)} | \boldsymbol{z}_j, \boldsymbol{W} \right)$.

Using this notation, the missing values may now be modelled as latent variables for implementation of the EM algorithm. For the EM specification, we have the observed data $\left\{ \boldsymbol{x}_j^{(O_j)} \right\}$, the latent variables $\left\{ \boldsymbol{x}_j^{(M_j)}, \boldsymbol{z}_j \right\}$, and the model parameters $\left\{ \boldsymbol{W}, \sigma^2 \right\}$. The logarithm of the joint likelihood for the observed and latent variables can be expressed

as

$$\sum_{j=1}^{n} \log \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)}, \boldsymbol{x}_j^{(M_j)}, \boldsymbol{z}_j | \boldsymbol{W}, \sigma^2\right) = \sum_{j=1}^{n} \log \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)} | \boldsymbol{x}_j^{(M_j)}, \boldsymbol{z}_j, \boldsymbol{W}, \sigma^2\right) + \sum_{j=1}^{n} \log \mathrm{p}\left(\boldsymbol{x}_j^{(M_j)} | \boldsymbol{z}_j, \boldsymbol{W}, \sigma^2\right) + \sum_{j=1}^{n} \log \mathrm{p}\left(\boldsymbol{z}_j | \boldsymbol{W}, \sigma^2\right). \tag{B.28}$$

Using the distributional assumption of the $\boldsymbol{z}_j$ in the PPCA model, as well as Equations (B.26) and (B.27), and rearranging terms, Equation (B.28) becomes

$$\begin{aligned}\sum_{j=1}^{n} \log \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)}, \boldsymbol{x}_j^{(M_j)}, \boldsymbol{z}_j | \boldsymbol{W}, \sigma^2\right) = \sum_{j=1}^{n} & -\left(\frac{|O_j|}{2} + \frac{|M_j|}{2}\right) \log 2\pi\sigma^2 - \frac{q}{2} \log 2\pi - \frac{1}{2} \boldsymbol{z}_j^\top \boldsymbol{z}_j \\ & - \frac{1}{2\sigma^2} \left(\boldsymbol{x}_j^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} + \boldsymbol{x}_j^{(M_j)\top} \boldsymbol{x}_j^{(M_j)} \right. \\ & + \mathrm{Tr}\left[\boldsymbol{z}_j \boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top} \boldsymbol{W}^{(O_j)}\right] \\ & + \mathrm{Tr}\left[\boldsymbol{z}_j \boldsymbol{z}_j^\top \boldsymbol{W}^{(M_j)\top} \boldsymbol{W}^{(M_j)}\right] \\ & \left. - 2\boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} - 2\boldsymbol{z}_j^\top \boldsymbol{W}^{(M_j)\top} \boldsymbol{x}_j^{(M_j)} \right) \end{aligned} \tag{B.29}$$

$$\begin{aligned} = & -\frac{np}{2} \log 2\pi\sigma^2 - \frac{nq}{2} \log 2\pi - \sum_{j=1}^{n} \frac{1}{2} \boldsymbol{z}_j^\top \boldsymbol{z}_j \\ & - \frac{1}{2\sigma^2} \left(\boldsymbol{x}_j^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} + \mathrm{Tr}\left[\boldsymbol{x}_j^{(M_j)} \boldsymbol{x}_j^{(M_j)\top}\right] \right. \\ & + \mathrm{Tr}\left[\boldsymbol{z}_j \boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top} \boldsymbol{W}^{(O_j)}\right] \\ & + \mathrm{Tr}\left[\boldsymbol{z}_j \boldsymbol{z}_j^\top \boldsymbol{W}^{(M_j)\top} \boldsymbol{W}^{(M_j)}\right] \\ & \left. - 2\boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} - 2\boldsymbol{z}_j^\top \boldsymbol{W}^{(M_j)\top} \boldsymbol{x}_j^{(M_j)} \right). \end{aligned} \tag{B.30}$$

B.2.2 E step

For the E step we want to compute the expectation of Equation (B.30) over the joint distribution of the latent variables $\left\{z_j, \boldsymbol{x}_j^{(M_j)}\right\}$, which can be given as

$$\begin{aligned}\sum_{j=1}^{n} \mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}} \left[\log \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)}, \boldsymbol{x}_j^{(M_j)}, z_j | \boldsymbol{W}, \sigma^2\right)\right] = &-\frac{np}{2}\log 2\pi\sigma^2 - \frac{nq}{2}\log 2\pi \\ &- \sum_{j=1}^{n}\frac{1}{2}\mathrm{Tr}\left[\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j^\top z_j\right]\right] \\ &- \frac{1}{2\sigma^2}\Big(\boldsymbol{x}_j^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} \\ &+ \mathrm{Tr}\left[\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\boldsymbol{x}_j^{(M_j)\top}\right]\right] \\ &+ \mathrm{Tr}\left[\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j z_j^\top\right]\boldsymbol{W}^{(O_j)\top}\boldsymbol{W}^{(O_j)}\right] \\ &+ \mathrm{Tr}\left[\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j z_j^\top\right]\boldsymbol{W}^{(M_j)\top}\boldsymbol{W}^{(M_j)}\right] \\ &- 2\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j^\top\right]\boldsymbol{W}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} \\ &- 2\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j^\top\right]\boldsymbol{W}^{(M_j)\top}\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right]\Big).\end{aligned} \tag{B.31}$$

We therefore require expressions for $\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j\right]$, $\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j z_j^\top\right]$, $\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right]$, and $\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\boldsymbol{x}_j^{(M_j)\top}\right]$. For the first, we have

$$\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j\right] = \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\mathbb{E}_{z_j | \boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right] \tag{B.32}$$

$$= \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{M}^{-1}\boldsymbol{W}^\top \boldsymbol{x}_j\right] \tag{B.33}$$

$$= \boldsymbol{M}^{-1}\boldsymbol{W}^\top \widetilde{\boldsymbol{x}}_j, \tag{B.34}$$

where we have used the law of total expectation, $\boldsymbol{M} = \boldsymbol{W}^\top\boldsymbol{W} + \sigma^2\boldsymbol{I}_{q\times q}$, and $\widetilde{\boldsymbol{x}}_j$ is $\boldsymbol{x}_j$ for $\boldsymbol{x}_j^{(O_j)}$ and $\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right]$ for $\boldsymbol{x}_j^{(M_j)}$, i.e. with missing values replaced by their expectation.

We also have

$$\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j z_j^\top\right] = \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\mathbb{E}_{z_j|\boldsymbol{x}_j^{(M_j)}}\left[z_j z_j^\top\right]\right] \tag{B.35}$$

$$= \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\sigma^2 \boldsymbol{M}^{-1} + \mathbb{E}_{z_j|\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right]\mathbb{E}_{z_j|\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right]^\top\right] \tag{B.36}$$

$$= \sigma^2 \boldsymbol{M}^{-1} + \mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top \tag{B.37}$$

For the expectations involving the missing values, we first have

$$\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right] = \mathbb{E}_{z_j}\left[\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}|z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]\right] \tag{B.38}$$

$$= \mathbb{E}_{z_j}\left[\boldsymbol{W}^{(M_j)} z_j\right] \tag{B.39}$$

$$= \boldsymbol{W}^{(M_j)}\mathbb{E}_{z_j}\left[z_j\right]. \tag{B.40}$$

We also have

$$\mathbb{E}_{z_j, \boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\boldsymbol{x}_j^{(M_j)\top}\right] = \mathbb{E}_{z_j}\left[\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}|z_j}\left[\boldsymbol{x}_j^{(M_j)}\boldsymbol{x}_j^{(M_j)\top}\right]\right] \tag{B.41}$$

$$= \mathbb{E}_{z_j}\left[\sigma^2 \boldsymbol{I}_{(p-|O_j|)\times(p-|O_j|)} + \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}|z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}|z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]^\top\right] \tag{B.42}$$

$$= \sigma^2 \boldsymbol{I}_{(p-|O_j|)\times(p-|O_j|)} + \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}, z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}, z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]^\top. \tag{B.43}$$

Some terms involving missing and observed indices can also be combined:

$$\boldsymbol{W}^{(O_j)\top}\boldsymbol{W}^{(O_j)} + \boldsymbol{W}^{(M_j)\top}\boldsymbol{W}^{(M_j)} = \sum_{i=1}^{|O_j|}\widetilde{\boldsymbol{w}}_i^\top \widetilde{\boldsymbol{w}}_i + \sum_{i=|O_j|+1}^{p}\widetilde{\boldsymbol{w}}_i^\top \widetilde{\boldsymbol{w}}_i \tag{B.44}$$

$$= \boldsymbol{W}^\top \boldsymbol{W}, \tag{B.45}$$

where we have used the notation $\widetilde{\boldsymbol{w}}_i$ to represent the i-th row vector of $\boldsymbol{W}$ and we also have

$$\boldsymbol{W}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} + \boldsymbol{W}^{(M_j)\top}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right] = \begin{bmatrix}\sum_{i=1}^{|O_j|} w_{i1}\widetilde{x}_{ij}\\ \dots \\ \sum_{i=1}^{|O_j|} w_{iq}\widetilde{x}_{ij}\end{bmatrix} + \begin{bmatrix}\sum_{i=|O_j|+1}^{p} w_{i1}\widetilde{x}_{ij}\\ \dots \\ \sum_{i=|O_j|+1}^{p} w_{iq}\widetilde{x}_{ij}\end{bmatrix} \tag{B.46}$$

$$= \begin{bmatrix}\sum_{i=1}^{p} w_{i1}\widetilde{x}_{ij}\\ \dots \\ \sum_{i=1}^{p} w_{iq}\widetilde{x}_{ij}\end{bmatrix} \tag{B.47}$$

$$= \boldsymbol{W}\widetilde{\boldsymbol{x}}_j. \tag{B.48}$$

Using Equations (B.34), (B.37), (B.40), (B.43), as well as the identities from Equations (B.45) and (B.48), Equation (B.31) becomes

$$\begin{aligned}
&\sum_{j=1}^{n}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[\log \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)},\boldsymbol{x}_j^{(M_j)},z_j|\boldsymbol{W},\sigma^2\right)\right]\\
&= -\frac{np}{2}\log 2\pi\sigma^2 - \frac{nq}{2}\log 2\pi\\
&\quad - \sum_{j=1}^{n}\frac{1}{2}\mathrm{Tr}\left[\sigma^2\boldsymbol{M}^{-1} + \mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\right]\\
&\quad - \frac{1}{2\sigma^2}\left\{\boldsymbol{x}_j^{(O_j)\top}\boldsymbol{x}_j^{(O_j)}\right.\\
&\quad + \mathrm{Tr}\left[\sigma^2\boldsymbol{I}_{(p-|O_j|)\times(p-|O_j|)} + \mathbb{E}_{\boldsymbol{x}_j^{(M_j)},z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]\mathbb{E}_{\boldsymbol{x}_j^{(M_j)},z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]^\top\right]\\
&\quad + \mathrm{Tr}\left[\left(\sigma^2\boldsymbol{M}^{-1} + \mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\right)\boldsymbol{W}^\top\boldsymbol{W}\right]\\
&\quad \left. - 2\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\boldsymbol{W}^\top\widetilde{\boldsymbol{x}}_j\right\}.
\end{aligned} \tag{B.49}$$

Noting the following identities:

$$\begin{aligned}
&\mathrm{Tr}\left[\sigma^2\boldsymbol{M}^{-1} + \mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\right]\\
&= \mathrm{Tr}\left[\sigma^2\boldsymbol{M}^{-1}\right] + \mathrm{Tr}\left[\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\right] &\text{(B.50)}\\
&= \mathrm{Tr}\left[\sigma^2\boldsymbol{M}^{-1}\right] + \mathrm{Tr}\left[\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right] &\text{(B.51)}\\
&= \mathrm{Tr}\left[\sigma^2\boldsymbol{M}^{-1}\right] + \mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right] &\text{(B.52)}\\
&= \mathrm{Tr}\left[\sigma^2\boldsymbol{M}^{-1}\right] + \left\|\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right\|^2, &\text{(B.53)}
\end{aligned}$$

$$\begin{aligned}
&\operatorname{Tr}\left[\sigma^2 \boldsymbol{I}_{(p-|O_j|)\times(p-|O_j|)} + \mathbb{E}_{\boldsymbol{x}_j^{(M_j)},z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]\mathbb{E}_{\boldsymbol{x}_j^{(M_j)},z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]^\top\right] \\
&= \operatorname{Tr}\left[\sigma^2 \boldsymbol{I}_{(p-|O_j|)\times(p-|O_j|)}\right] + \operatorname{Tr}\left[\mathbb{E}_{\boldsymbol{x}_j^{(M_j)},z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]\mathbb{E}_{\boldsymbol{x}_j^{(M_j)},z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]^\top\right] && \text{(B.54)} \\
&= \sigma^2(p-|O_j|) + \operatorname{Tr}\left[\mathbb{E}_{\boldsymbol{x}_j^{(M_j)},z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]^\top \mathbb{E}_{\boldsymbol{x}_j^{(M_j)},z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]\right] && \text{(B.55)} \\
&= \sigma^2(p-|O_j|) + \mathbb{E}_{\boldsymbol{x}_j^{(M_j)},z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]^\top \mathbb{E}_{\boldsymbol{x}_j^{(M_j)},z_j}\left[\boldsymbol{x}_j^{(M_j)}\right] && \text{(B.56)} \\
&= \sigma^2(p-|O_j|) + \left\|\mathbb{E}_{\boldsymbol{x}_j^{(M_j)},z_j}\left[\boldsymbol{x}_j^{(M_j)}\right]\right\|^2, && \text{(B.57)}
\end{aligned}$$

and

$$\begin{aligned}
&\operatorname{Tr}\left[\left(\sigma^2 \boldsymbol{M}^{-1} + \mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\right)\boldsymbol{W}^\top\boldsymbol{W}\right] && \text{(B.58)} \\
&= \operatorname{Tr}\left[\boldsymbol{W}\left(\sigma^2 \boldsymbol{M}^{-1} + \mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\right)\boldsymbol{W}^\top\right] && \text{(B.59)} \\
&= \operatorname{Tr}\left[\boldsymbol{W}\sigma^2 \boldsymbol{M}^{-1}\boldsymbol{W}^\top + \boldsymbol{W}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\boldsymbol{W}^\top\right] && \text{(B.60)} \\
&= \operatorname{Tr}\left[\boldsymbol{W}\sigma^2 \boldsymbol{M}^{-1}\boldsymbol{W}^\top\right] + \operatorname{Tr}\left[\boldsymbol{W}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\boldsymbol{W}^\top\right] && \text{(B.61)} \\
&= \operatorname{Tr}\left[\boldsymbol{W}\sigma^2 \boldsymbol{M}^{-1}\boldsymbol{W}^\top\right] + \operatorname{Tr}\left[\left(\boldsymbol{W}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right)\left(\boldsymbol{W}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right)^\top\right] && \text{(B.62)} \\
&= \operatorname{Tr}\left[\boldsymbol{W}\sigma^2 \boldsymbol{M}^{-1}\boldsymbol{W}^\top\right] + \operatorname{Tr}\left[\left(\boldsymbol{W}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right)^\top\left(\boldsymbol{W}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right)\right] && \text{(B.63)} \\
&= \operatorname{Tr}\left[\boldsymbol{W}\sigma^2 \boldsymbol{M}^{-1}\boldsymbol{W}^\top\right] + \operatorname{Tr}\left[\left\|\boldsymbol{W}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right\|^2\right] && \text{(B.64)} \\
&= \operatorname{Tr}\left[\boldsymbol{W}\sigma^2 \boldsymbol{M}^{-1}\boldsymbol{W}^\top\right] + \left\|\boldsymbol{W}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right\|^2, && \text{(B.65)}
\end{aligned}$$

the joint likelihood in Equation (B.49) can be written as

$$\begin{aligned}
&\sum_{j=1}^{n}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[\log \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)},\boldsymbol{x}_j^{(M_j)},z_j|\boldsymbol{W},\sigma^2\right)\right] \\
&\quad=-\frac{np}{2}\log 2\pi\sigma^2-\frac{nq}{2}\log 2\pi-\sum_{j=1}^{n}\frac{1}{2}\mathrm{Tr}\left[\sigma^2\boldsymbol{M}^{-1}\right]-\frac{1}{2}\left\|\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right\|^2 \\
&\quad\quad-\frac{1}{2\sigma^2}\left(\left\|\boldsymbol{x}_j^{(O_j)}\right\|^2+\sigma_{\mathrm{old}}^2(p-|O_j|)+\left\|\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right]\right\|^2\right. \\
&\quad\quad\left.+\mathrm{Tr}\left[\boldsymbol{W}\sigma^2\boldsymbol{M}^{-1}\boldsymbol{W}^\top\right]+\left\|\boldsymbol{W}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right\|^2-2\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\boldsymbol{W}^\top\widetilde{\boldsymbol{x}}_j\right) \qquad \text{(B.66)} \\
&\quad=-\frac{np}{2}\log 2\pi\sigma^2-\frac{nq}{2}\log 2\pi-\frac{n}{2}\mathrm{Tr}\left[\sigma^2\boldsymbol{M}^{-1}\right]-\sum_{j=1}^{n}\frac{1}{2}\left\|\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right\|^2 \\
&\quad\quad-\frac{\sigma_{\mathrm{old}}^2}{2\sigma^2}\sum_{j=1}^{n}(p-|O_j|)-\frac{1}{2\sigma^2}\left(\sum_{j=1}^{n}\left\|\widetilde{\boldsymbol{x}}_j-\boldsymbol{W}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right\|^2\right. \\
&\quad\quad\left.-n\mathrm{Tr}\left[\boldsymbol{W}\sigma^2\boldsymbol{M}^{-1}\boldsymbol{W}^\top\right]\right). \qquad \text{(B.67)}
\end{aligned}$$

B.2.3 M step

Differentiating Equation (B.67) with respect to $\boldsymbol{W}$ first, we get

$$\begin{aligned}
\boldsymbol{W}_{\mathrm{new}}&=\left(\sum_{j=1}^{n}\widetilde{\boldsymbol{x}}_j\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\right)\left(\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_jz_j^\top\right]\right)^{-1} && \text{(B.68)} \\
&=\widetilde{\boldsymbol{X}}\overline{\boldsymbol{Z}}^\top\left(\sum_{j=1}^{n}\sigma^2\boldsymbol{M}^{-1}+\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]^\top\right)^{-1} && \text{(B.69)} \\
&=\widetilde{\boldsymbol{X}}\overline{\boldsymbol{Z}}^\top\left(n\sigma^2\boldsymbol{M}^{-1}+\overline{\boldsymbol{Z}}\overline{\boldsymbol{Z}}^\top\right)^{-1}. && \text{(B.70)}
\end{aligned}$$

Maximisation with respect to σ^2 gives

$$\begin{aligned}
\sigma_{\mathrm{new}}^2=&\frac{1}{np}\left(n\mathrm{Tr}\left[\boldsymbol{W}\sigma_{\mathrm{old}}^2\boldsymbol{M}^{-1}\boldsymbol{W}^\top\right]+\sum_{j=1}^{n}\left\|\widetilde{\boldsymbol{x}}_j-\boldsymbol{W}\mathbb{E}_{z_j,\boldsymbol{x}_j^{(M_j)}}\left[z_j\right]\right\|^2\right. \\
&\left.+\sigma_{\mathrm{old}}^2\sum_{j=1}^{n}(p-|O_j|)\right). \qquad \text{(B.71)}
\end{aligned}$$

The algorithm is executed by iterating through Equations (B.34), (B.37), (B.40), (B.43), (B.70), and (B.71).

B.3 EM algorithm 2 – derivation

Here we consider the EM algorithm of Ilin and Raiko [60]. This approach differs to that of Stacklies et al. [98] in that $\boldsymbol{\mu}$ is updated at each iteration and only the observed values are used in the expectation and parameter updates. We derive the updates for this situation as in Ilin and Raiko [60].

Since $\boldsymbol{x}_j|\boldsymbol{z}_j \sim \mathcal{N}(\boldsymbol{W}\boldsymbol{z}_j + \boldsymbol{\mu}, \sigma^2\boldsymbol{I}_{p\times p})$, we have

$$\boldsymbol{x}_j^{(O_j)}|\boldsymbol{z}_j \sim \mathcal{N}\left(\boldsymbol{W}^{(O_j)}\boldsymbol{z}_j + \boldsymbol{\mu}^{(O_j)}, \sigma^2\boldsymbol{I}_{p\times p}\right). \tag{B.72}$$

It follows that the distribution of $\boldsymbol{z}_j$ given $\boldsymbol{x}_j^{(O_j)}$ is

$$\boldsymbol{z}_j|\boldsymbol{x}_j^{(O_j)} \sim \mathcal{N}\left(\left(\boldsymbol{M}^{(O_j)}\right)^{-1}\left(\boldsymbol{W}^{(O_j)}\right)^\top\left(\boldsymbol{x}_j - \boldsymbol{\mu}^{(O_j)}\right), \sigma^2\left(\boldsymbol{M}^{(O_j)}\right)^{-1}\right), \tag{B.73}$$

where

$$\boldsymbol{M}^{(O_j)} = \sigma^2\boldsymbol{I}_{q\times q} + \left(\boldsymbol{W}^{(O_j)}\right)^\top\boldsymbol{W}^{(O_j)} = \sigma^2\boldsymbol{I}_{q\times q} + \sum_{i\in O_j}\boldsymbol{w}_i\boldsymbol{w}_i^\top \tag{B.74}$$

with $\boldsymbol{w}_i^\top$ defined to be the i-th row of $\boldsymbol{W}$, which has dimension q.

B.3.1 E step

Analogously to Equations (3.7) and (3.8), we then have the E-step updates

$$\mathbb{E}_{z_j}[\boldsymbol{z}_j] = \left(\boldsymbol{M}^{(O_j)}\right)^{-1}\sum_{i\in O_j}\boldsymbol{w}_i(x_{ij} - \mu_i) \tag{B.75}$$

$$\mathbb{E}_{z_j}\left[\boldsymbol{z}_j\boldsymbol{z}_j^\top\right] = \sigma^2\left(\boldsymbol{M}^{(O_j)}\right)^{-1} + \mathbb{E}_{z_j}[\boldsymbol{z}_j]\mathbb{E}_{z_j}[\boldsymbol{z}_j]^\top. \tag{B.76}$$

B.3.2 M step

Following steps similar to the case where there are no missing values in Section B.1, one can obtain the iterative updates for the parameters [60]. Maximisation with respect

to each parameter gives

$$(\mu_i)_{\text{new}} = \frac{1}{|O_i|} \sum_{j \in O_i} \left(x_{ij} - \boldsymbol{w}_i^\top \mathbb{E}_{z_j}[\boldsymbol{z}_j] \right) \tag{B.77}$$

$$(\boldsymbol{w}_i)_{\text{new}} = \left(\sum_{j=1}^{n} (x_{ij} - \mu_i) \mathbb{E}_{z_j}[\boldsymbol{z}_j]^\top \right) \left(\sum_{j \in O_i} \mathbb{E}_{z_j} \left[\boldsymbol{z}_j \boldsymbol{z}_j^\top \right] \right)^{-1} \tag{B.78}$$

$$\begin{aligned} \sigma^2_{\text{new}} = & \frac{1}{|O|} \sum_{ij \in O} (\boldsymbol{x}_{ij} - \mu_i)^2 + \text{Tr} \left[\mathbb{E}_{z_j} \left[\boldsymbol{z}_j \boldsymbol{z}_j^\top \right] (\boldsymbol{w}_i)_{\text{new}} (\boldsymbol{w}_i)_{\text{new}}^\top \right] \\ & - 2 (\boldsymbol{w}_i)_{\text{new}}^\top \mathbb{E}_{z_j}[\boldsymbol{z}_j]^\top (x_{ij} - \mu_i). \end{aligned} \tag{B.79}$$

We see that Equations (3.9), (3.10), and (3.11) can be derived as special cases of those above when there are no missing values. The algorithm is executed by iterating through Equations (B.75), (B.76), (B.77), (B.78), and (B.79).

B.4 Variational algorithm 1 - derivation

In this Section we attempt to derive the variational algorithm using the model from Oba et al. [82]. The full model is not provided in the paper and neither are the variational updates. We were able to contact the author in order to obtain clarity on the full model that was used. We were made aware of a further paper [81] in which details of a different (but related) algorithm are given in a situation without missing values. This paper does provide its model specification and variational approximation, but also emits the explicit variational update equations. Update equations are provided for a more simplified model but this is not the model assumed in the original paper [82] and again the presence of missing values is not considered. We also note that the updates derived in this Section do not completely agree with those presented in the code accompanying the paper [82]. After contacting the author about this discrepancy, we received no further reply. We also note that the mathematical details provided in Oba et al. [81] have been claimed to contain a mistake [1]. We followed the derivations provided in Agarwal and Bishop [1], which contain a more general model of that from Oba et al. [82]. Moreover, we extended the model presented in Agarwal and Bishop [1] to include the presence of missing values. The novelty of this Section is therefore an extension to a general model for Bayesian PCA in the presence of missing values.

B.4.1 Handling of missing data

We note that $\boldsymbol{x}_j$ may possess missing values and assume throughout that these are missing at random. When missing values are present, we adopt the partition (rearranging

the missing and observed values, and their respective rows of $\boldsymbol{W}$ and $\boldsymbol{\mu}$:

$$\boldsymbol{x}_j = \begin{bmatrix} \boldsymbol{x}_j^{(O_j)} \\ \boldsymbol{x}_j^{(M_j)} \end{bmatrix}, \tag{B.80}$$

where $\boldsymbol{x}_j^{(O_j)}$ and $\boldsymbol{x}_j^{(M_j)}$ correspond to the sets of observed and missing values, respectively. Denoting O_j as the set that contains the indices (after partitioning) of $\boldsymbol{x}_j$ that comprise $\boldsymbol{x}_j^{(O_j)}$, we can see that our partitioning induces $O_j = \{1,\ldots,|O_j|\}$, using the notation $|O_j|$ to denote the cardinality of the set O_j. Similarly using M_j for the missing indices we have that $M_j = \{|O_j|+1,\ldots,p\}$. Explicitly this can be written as $\boldsymbol{x}_j^{(O_j)} = (x_1,\ldots,x_{|O_j|})^\top$ and $\boldsymbol{x}_j^{(M_j)} = (x_{|O_j|+1},\ldots,x_p)^\top$, from which it is clear that the vectors are of dimension $|O_j|$ and $p-|O_j|$, respectively.

An implication of Equation (B.80) is that $\boldsymbol{\mu}$ and $\boldsymbol{W}$ may now be partitioned in similar fashion:

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}^{(O_j)} \\ \boldsymbol{\mu}^{(M_j)} \end{bmatrix}, \tag{B.81}$$

and

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}^{(O_j)} \\ \boldsymbol{W}^{(M_j)} \end{bmatrix}. \tag{B.82}$$

Equation (B.81) is analogous to Equation (B.80); for the matrices $\boldsymbol{W}^{(O_j)}$ and $\boldsymbol{W}^{(M_j)}$, we have retained only those rows of $\boldsymbol{W}$ which correspond to $\boldsymbol{x}_j^{(O_j)}$ and $\boldsymbol{x}_j^{(M_j)}$, respectively. Explicitly, the entries $w_{ij}^{(O_j)} = \{w_{ij} | i = 1,\ldots,|O_j|, j = 1,\ldots,q\}$ and $w_{ij}^{(M_j)} = \{w_{ij} | i = |O_j|+1,\ldots,p, j = 1,\ldots,q\}$ so that the matrices $\boldsymbol{W}^{(O_j)}$ and $\boldsymbol{W}^{(M_j)}$ are of dimension $|O_j| \times q$ and $(p-|O_j|)\times q$, respectively.

The conditional distribution of $\boldsymbol{x}_j$ given $\boldsymbol{z}_j$ can now be written in this partitioned form as

$$\begin{bmatrix} \boldsymbol{x}_j^{(O_j)} \\ \boldsymbol{x}_j^{(M_j)} \end{bmatrix} | \boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{W}^{(O_j)}\boldsymbol{z}_j + \boldsymbol{\mu}^{(O_j)} \\ \boldsymbol{W}^{(M_j)}\boldsymbol{z}_j + \boldsymbol{\mu}^{(M_j)} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}^{(O_j),(O_j)} & \boldsymbol{\Sigma}^{(O_j),(M_j)} \\ \boldsymbol{\Sigma}^{(M_j),(O_j)} & \boldsymbol{\Sigma}^{(M_j),(M_j)} \end{bmatrix} \right). \tag{B.83}$$

It can then be inferred that $\boldsymbol{\Sigma}^{(O_j),(M_j)} = \boldsymbol{0}_{|O_j|\times(p-|O_j|)}$, $\boldsymbol{\Sigma}^{(M_j),(O_j)} = \boldsymbol{0}_{(p-|O_j|)\times|O_j|}$ and $\boldsymbol{\Sigma}^{(O_j),(O_j)} = \tau^{-1}\boldsymbol{I}_{|O_j|\times|O_j|}$, and $\boldsymbol{\Sigma}^{(M_j),(M_j)} = \tau^{-1}\boldsymbol{I}_{(p-|O_j|)\times(p-|O_j|)}$. From Equation (B.83), it can be seen that

$$\boldsymbol{x}_j^{(O_j)} | \boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau \sim \mathcal{N}\left(\boldsymbol{W}^{(O_j)}\boldsymbol{z}_j + \boldsymbol{\mu}^{(O_j)}, \tau^{-1}\boldsymbol{I}_{|O_j|\times|O_j|} \right) \tag{B.84}$$

and

$$\boldsymbol{x}_j^{(M_j)}|z_j,\boldsymbol{W},\boldsymbol{\mu},\tau \sim \mathcal{N}\left(\boldsymbol{W}^{(M_j)}\boldsymbol{z}_j + \boldsymbol{\mu}^{(M_j)}, \tau^{-1}\boldsymbol{I}_{(p-|O_j|)\times(p-|O_j|)}\right). \tag{B.85}$$

It is worth noting that that $\boldsymbol{x}_j^{(O_j)}$ and $\boldsymbol{x}_j^{(M_j)}$ are assumed to be conditionally independent given the value of $\boldsymbol{z}_j$ so that $\mathrm{p}\left(\boldsymbol{x}_j^{(O_j)}|\boldsymbol{x}_j^{(M_j)},\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right) = \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)}|\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right)$. Using this, we can factor the likelihood of each $\boldsymbol{x}_j$ as follows:

$$\begin{aligned}
\mathrm{p}\left(\boldsymbol{x}_j|\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right) &= \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)},\boldsymbol{x}_j^{(M_j)}|\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right) && \text{(B.86)}\\
&= \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)}|\boldsymbol{x}_j^{(M_j)},\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right)\mathrm{p}\left(\boldsymbol{x}_j^{(M_j)}|\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right) && \text{(B.87)}\\
&= \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)}|\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right)\mathrm{p}\left(\boldsymbol{x}_j^{(M_j)}|\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right). && \text{(B.88)}
\end{aligned}$$

B.4.2 Priors

From Oba et al. [82], the prior specification is as follows:

$$\begin{aligned}
\mathrm{p}(\boldsymbol{\mu},\boldsymbol{W},\tau|\boldsymbol{\alpha}) &= \mathrm{p}(\boldsymbol{\mu}|\tau)\mathrm{p}(\boldsymbol{W}|\tau,\boldsymbol{\alpha})\mathrm{p}(\tau), && \text{(B.89)}\\
\mathrm{p}(\boldsymbol{\mu}|\tau) &= \mathcal{N}(\overline{\boldsymbol{\mu}}_0,(\gamma_{\boldsymbol{\mu}_0}\tau)^{-1}\boldsymbol{I}_{p\times p}), && \text{(B.90)}\\
\mathrm{p}(\boldsymbol{W}|\tau,\boldsymbol{\alpha}) &= \prod_{g=1}^{q}\mathrm{p}(\boldsymbol{w}_g|\tau,\alpha_g) = \prod_{g=1}^{q}\mathcal{N}(\boldsymbol{0},(\alpha_g\tau)^{-1}\boldsymbol{I}_{p\times p}), && \text{(B.91)}\\
\mathrm{p}(\tau) &= \Gamma(\overline{\tau}_0,\gamma_{\tau_0}), && \text{(B.92)}
\end{aligned}$$

where $\boldsymbol{w}_g$ denotes the g-th column of matrix $\boldsymbol{W}$. The function $\Gamma(\overline{\tau}_0,\gamma_{\tau_0})$ denotes the univariate gamma probability density, defined as:

$$\Gamma(\overline{\tau}_0,\gamma_{\tau_0}) \equiv \frac{\gamma_{\tau_0}\overline{\tau}_0^{-1}}{\Gamma(\gamma_{\tau_0})}\exp\left(-\gamma_{\tau_0}\overline{\tau}_0^{-1}\tau + (\gamma_{\tau_0}-1)\log\tau\right), \tag{B.93}$$

where $\Gamma(\gamma_{\tau_0})$ denotes the Gamma function and the natural base logarithm is taken. Using this definition yields $\mathbb{E}[\tau] = \overline{\tau}_0$. In Oba et al. [82], the hyperparameters are fixed as $\overline{\boldsymbol{\mu}}_0 = 0$, $\overline{\tau}_0 = 1$, and $\gamma_{\tau_0} = \gamma_{\boldsymbol{\mu}_0} = 10^{-10}$ (although $\gamma_{\boldsymbol{\mu}_0} = 10^{-3}$ in the code). Email correspondence with the author revealed that there is also a prior on $\boldsymbol{\alpha}$ that is used in the code:

$$\mathrm{p}(\boldsymbol{\alpha}) = \prod_{g=1}^{q}\mathrm{p}(\alpha_g) = \prod_{g=1}^{q}\Gamma(\overline{\alpha}_0,\gamma_{\alpha_0}), \tag{B.94}$$

whose hyperparameters are fixed as $\overline{\alpha}_0 = 1$, and $\gamma_{\alpha_0} = 10^{-10}$. In the following derivations, no hyperparameter values are fixed. This is done so that clarity may be preserved in seeing how the hyperparameters influence the variational updates. Recalling the prior

on $\boldsymbol{z}_j$ completes the prior specification:

$$\mathrm{p}(\boldsymbol{z}_j) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{q\times q}). \tag{B.95}$$

In the following sections, the missing values $\boldsymbol{x}^{(M_j)}$ are treated as latent variables akin to $\boldsymbol{z}_j$, with their prior distribution given by Equation (B.85).

B.4.3 Joint distribution

Using this prior and likelihood, the joint distribution for all variables factors as so:

$$\begin{aligned}\mathrm{p}(\boldsymbol{x}_j, \boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau, \boldsymbol{\alpha}) =& \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)} | \boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau\right) \mathrm{p}\left(\boldsymbol{x}_j^{(M_j)} | \boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau\right) \mathrm{p}(\boldsymbol{z}_j)\mathrm{p}(\boldsymbol{\mu}|\tau) \\ & \times \mathrm{p}(\boldsymbol{W}|\tau, \boldsymbol{\alpha})\mathrm{p}(\tau)\mathrm{p}(\boldsymbol{\alpha})\end{aligned} \tag{B.96}$$

The logarithm of Equation (B.96) is used to derive the variational updates, so expressions for the logarithm of each of the terms on the right-hand side of Equation (B.96) are required.

Using Equation (B.84), we have

$$\begin{aligned}\log \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)} | \boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau\right) =& -\frac{|O_j|}{2}\log 2\pi - \frac{|O_j|}{2}\log \tau^{-1} \\ & - \frac{\tau}{2}\left(\boldsymbol{x}_j^{(O_j)} - \boldsymbol{W}^{(O_j)}\boldsymbol{z}_j - \boldsymbol{\mu}^{(O_j)}\right)^\top \left(\boldsymbol{x}_j^{(O_j)} - \boldsymbol{W}^{(O_j)}\boldsymbol{z}_j - \boldsymbol{\mu}^{(O_j)}\right)\end{aligned} \tag{B.97}$$

$$\begin{aligned}=& -\frac{|O_j|}{2}\log 2\pi - \frac{|O_j|}{2}\log \tau^{-1} \\ & - \frac{\tau}{2}\left(\boldsymbol{x}_j^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} - \boldsymbol{x}_j^{(O_j)\top}\boldsymbol{W}^{(O_j)}\boldsymbol{z}_j - \boldsymbol{x}_j^{(O_j)\top}\boldsymbol{\mu}^{(O_j)}\right. \\ & - \boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} - \boldsymbol{\mu}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} + \boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top}\boldsymbol{W}^{(O_j)}\boldsymbol{z}_j \\ & \left. + \boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top}\boldsymbol{\mu}^{(O_j)} + \boldsymbol{\mu}^{(O_j)\top}\boldsymbol{W}^{(O_j)}\boldsymbol{z}_j + \boldsymbol{\mu}^{(O_j)\top}\boldsymbol{\mu}^{(O_j)}\right).\end{aligned} \tag{B.98}$$

This can be simplified by noting that for a- and b-dimensional column vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ and $a \times b$ dimensional matrix $\boldsymbol{C}$ we have $\boldsymbol{a}^\top \boldsymbol{C}\boldsymbol{b} = \mathrm{Tr}\left[\boldsymbol{a}^\top \boldsymbol{C}\boldsymbol{b}\right] = \mathrm{Tr}\left[(\boldsymbol{a}^\top \boldsymbol{C}\boldsymbol{b})^\top\right] = \mathrm{Tr}\left[\boldsymbol{b}^\top \boldsymbol{C}^\top \boldsymbol{a}\right] = \boldsymbol{b}^\top \boldsymbol{C}^\top \boldsymbol{a}$. Similarly, for a-dimensional column vectors $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$ we have $\boldsymbol{a}_1^\top \boldsymbol{a}_2 = \mathrm{Tr}\left[\boldsymbol{a}_1^\top \boldsymbol{a}_2\right] = \mathrm{Tr}\left[\left(\boldsymbol{a}_1^\top \boldsymbol{a}_2\right)^\top\right] = \mathrm{Tr}\left[\boldsymbol{a}_2^\top \boldsymbol{a}_1\right] = \boldsymbol{a}_2^\top \boldsymbol{a}_1$, where in both identities we have used the fact that a matrix and its transpose have the same trace. Using these

properties, Equation (B.98) becomes

$$\begin{aligned}\log \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)}|z_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right) = &-\frac{|O_j|}{2}\log 2\pi - \frac{|O_j|}{2}\log\tau^{-1} \\ &-\frac{\tau}{2}\left(\boldsymbol{x}_j^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} + 2\boldsymbol{\mu}^{(O_j)\top}\boldsymbol{W}^{(O_j)}z_j - 2z_j^\top\boldsymbol{W}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)}\right. \\ &\left.-2\boldsymbol{\mu}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} + z_j^\top\boldsymbol{W}^{(O_j)\top}\boldsymbol{W}^{(O_j)}z_j + \boldsymbol{\mu}^{(O_j)\top}\boldsymbol{\mu}^{(O_j)}\right). \quad \text{(B.99)}\end{aligned}$$

Using Equation (B.99), we can generalise this for n independent data vectors distributed as in Equation (B.84) as:

$$\log \mathrm{p}\left(\boldsymbol{x}_1^{(O_1)},\ldots,\boldsymbol{x}_n^{(O_n)}|\boldsymbol{Z},\boldsymbol{W},\boldsymbol{\mu},\tau\right) = \sum_{j=1}^{n}\log \mathrm{p}\left(\boldsymbol{x}_j^{(O_j)}|z_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right) \quad \text{(B.100)}$$

$$\begin{aligned}&= -\frac{n|O_j|}{2}\log 2\pi - \frac{n|O_j|}{2}\log\tau^{-1} \\ &\quad -\frac{\tau}{2}\sum_{j=1}^{n}\left(\boldsymbol{x}_j^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} + 2\boldsymbol{\mu}^{(O_j)\top}\boldsymbol{W}^{(O_j)}z_j\right. \\ &\quad -2z_j^\top\boldsymbol{W}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} - 2\boldsymbol{\mu}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} \\ &\quad \left.+z_j^\top\boldsymbol{W}^{(O_j)\top}\boldsymbol{W}^{(O_j)}z_j + \boldsymbol{\mu}^{(O_j)\top}\boldsymbol{\mu}^{(O_j)}\right). \quad \text{(B.101)}\end{aligned}$$

Since $\boldsymbol{x}_j^{(O_j)}$ and $\boldsymbol{x}_j^{(M_j)}$ have the same form of distribution, it is immediate from Equations (B.85) and (B.99) that

$$\begin{aligned}\log \mathrm{p}\left(\boldsymbol{x}_j^{(M_j)}|z_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right) = &-\frac{p-|O_j|}{2}\log 2\pi - \frac{p-|O_j|}{2}\log\tau^{-1} \\ &-\frac{\tau}{2}\left(\boldsymbol{x}_j^{(M_j)\top}\boldsymbol{x}_j^{(M_j)} + 2\boldsymbol{\mu}^{(M_j)\top}\boldsymbol{W}^{(M_j)}z_j - 2z_j^\top\boldsymbol{W}^{(M_j)\top}\boldsymbol{x}_j^{(M_j)}\right. \\ &\left.-2\boldsymbol{\mu}^{(M_j)\top}\boldsymbol{x}_j^{(M_j)} + z_j^\top\boldsymbol{W}^{(M_j)\top}\boldsymbol{W}^{(M_j)}z_j + \boldsymbol{\mu}^{(M_j)\top}\boldsymbol{\mu}^{(M_j)}\right). \\ &\quad \text{(B.102)}\end{aligned}$$

Generalising to n independent data vectors as in Equation (B.101) using the individual distributions in Equation (B.85) gives

$$\log \mathrm{p}\left(\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)}|\boldsymbol{Z},\boldsymbol{W},\boldsymbol{\mu},\tau\right) = \sum_{j=1}^{n} \log \mathrm{p}\left(\boldsymbol{x}_j^{(M_j)}|\boldsymbol{z}_j,\boldsymbol{W},\boldsymbol{\mu},\tau\right) \tag{B.103}$$

$$\begin{aligned} &= -\frac{n(p-|O_j|)}{2}\log 2\pi - \frac{n(p-|O_j|)}{2}\log \tau^{-1} \\ &\quad -\frac{\tau}{2}\sum_{j=1}^{n}\Big(\boldsymbol{x}_j^{(M_j)\top}\boldsymbol{x}_j^{(M_j)} + 2\boldsymbol{\mu}^{(M_j)\top}\boldsymbol{W}^{(M_j)}\boldsymbol{z}_j \\ &\quad -2\boldsymbol{z}_j^{\top}\boldsymbol{W}^{(M_j)\top}\boldsymbol{x}_j^{(M_j)} - 2\boldsymbol{\mu}^{(M_j)\top}\boldsymbol{x}_j^{(M_j)} \\ &\quad +\boldsymbol{z}_j^{\top}\boldsymbol{W}^{(M_j)\top}\boldsymbol{W}^{(M_j)}\boldsymbol{z}_j + \boldsymbol{\mu}^{(M_j)\top}\boldsymbol{\mu}^{(M_j)}\Big). \end{aligned} \tag{B.104}$$

For the latent variables $\boldsymbol{z}_j$, we use Equation (B.95) to obtain the simple expression

$$\log \mathrm{p}(\boldsymbol{z}_j) = -\frac{q}{2}\log 2\pi - \frac{1}{2}\boldsymbol{z}_j^{\top}\boldsymbol{z}_j, \tag{B.105}$$

from which we also obtain the result for n vectors

$$\log \mathrm{p}(\boldsymbol{Z}) = -\frac{nq}{2}\log 2\pi - \sum_{j=1}^{n}\frac{1}{2}\boldsymbol{z}_j^{\top}\boldsymbol{z}_j. \tag{B.106}$$

For the mean vector $\boldsymbol{\mu}$, we use Equation (B.90) to yield

$$\log \mathrm{p}(\boldsymbol{\mu}|\tau) = -\frac{p}{2}\log 2\pi - \frac{1}{2}\log\left|(\gamma_{\mu_0}\tau)^{-1}\boldsymbol{I}_{p\times p}\right| - \frac{\gamma_{\mu_0}\tau}{2}\left(\boldsymbol{\mu}-\overline{\boldsymbol{\mu}}_0\right)^{\top}\left(\boldsymbol{\mu}-\overline{\boldsymbol{\mu}}_0\right) \tag{B.107}$$

$$= -\frac{p}{2}\log 2\pi - \frac{p}{2}\log(\gamma_{\mu_0}\tau)^{-1} - \frac{\gamma_{\mu_0}\tau}{2}\left(\boldsymbol{\mu}^{\top}\boldsymbol{\mu} - 2\boldsymbol{\mu}^{\top}\overline{\boldsymbol{\mu}}_0 + \overline{\boldsymbol{\mu}}_0^{\top}\overline{\boldsymbol{\mu}}_0\right), \tag{B.108}$$

where we have evaluated the matrix determinant $\left|(\gamma_{\mu_0}\tau)^{-1}\boldsymbol{I}_{p\times p}\right| = (\gamma_{\mu_0}\tau)^{-p}$.

For the loadings matrix $\boldsymbol{W}$, we use Equation (B.91) to obtain

$$\log \mathrm{p}(\boldsymbol{W}|\tau,\boldsymbol{\alpha}) = \log\prod_{g=1}^{q}\mathrm{p}(\boldsymbol{w}_g|\tau,\alpha_g) \tag{B.109}$$

$$= \sum_{g=1}^{q}\left\{-\frac{p}{2}\log 2\pi - \frac{1}{2}\log|(\alpha_g\tau)^{-1}\boldsymbol{I}_{p\times p}| - \frac{\alpha_g\tau}{2}\boldsymbol{w}_g^{\top}\boldsymbol{w}_g\right\} \tag{B.110}$$

$$= \sum_{g=1}^{q}\left\{-\frac{p}{2}\log 2\pi - \frac{p}{2}\log(\alpha_g\tau)^{-1} - \frac{\alpha_g\tau}{2}\boldsymbol{w}_g^{\top}\boldsymbol{w}_g\right\}. \tag{B.111}$$

For the precision parameter τ, we use Equation (B.92) to get

$$\log \mathrm{p}(\tau) = \log \left\{ \frac{(\gamma_{\tau_0} \overline{\tau}_0^{-1})^{\gamma_{\tau_0}}}{\Gamma(\gamma_{\tau_0})} \exp\left(-\gamma_{\tau_0} \overline{\tau}_0^{-1} \tau + (\gamma_{\tau_0} - 1) \log \tau\right) \right\} \tag{B.112}$$

$$= \gamma_{\tau_0} \log \gamma_{\tau_0} \overline{\tau}_0^{-1} - \log \Gamma(\gamma_{\tau_0}) - \gamma_{\tau_0} \overline{\tau}_0^{-1} \tau + (\gamma_{\tau_0} - 1) \log \tau. \tag{B.113}$$

Finally, for the automatic relevance determination parameter $\boldsymbol{\alpha}$, we use Equation (B.94) to see that

$$\log \mathrm{p}(\boldsymbol{\alpha}) = \log \prod_{g=1}^{q} \left\{ \frac{(\gamma_{\alpha_0} \overline{\alpha}_0^{-1})^{\gamma_{\alpha_0}}}{\Gamma(\gamma_{\alpha_0})} \exp\left(-\gamma_{\alpha_0} \overline{\alpha}_0^{-1} \alpha_g + (\gamma_{\alpha_0} - 1) \log \alpha_g\right) \right\} \tag{B.114}$$

$$= \sum_{g=1}^{q} \left\{ \gamma_{\alpha_0} \log \gamma_{\alpha_0} \overline{\alpha}_0^{-1} - \log \Gamma(\gamma_{\alpha_0}) - \gamma_{\alpha_0} \overline{\alpha}_0^{-1} \alpha_g + (\gamma_{\alpha_0} - 1) \log \alpha_g \right\}, \tag{B.115}$$

which completes the terms required for Equation (B.96), the joint likelihood.

B.4.4 Variational updates

Variational approximation

From email correspondence with Shigeyuki Oba, author of Oba et al. [82], the variational approximation that is used takes the form

$$\mathrm{q}\left(\boldsymbol{x}_1^{(M_1)}, \ldots, \boldsymbol{x}_n^{(M_n)}, \boldsymbol{Z}, \boldsymbol{\mu}, \boldsymbol{W}, \tau, \boldsymbol{\alpha}\right) = \prod_{j=1}^{n} \mathrm{q}\left(\boldsymbol{x}_j^{(M_j)}\right) \mathrm{q}(\boldsymbol{z}_j) \mathrm{q}(\boldsymbol{\mu}, \boldsymbol{W}, \tau) \mathrm{q}(\boldsymbol{\alpha}). \tag{B.116}$$

Variational lower bound

The variational lower bound takes the form

$$\begin{aligned}
\mathcal{L}(\mathrm{q}) = &\int \mathrm{q}\left(\boldsymbol{x}_1^{(O_1)},\ldots,\boldsymbol{x}_n^{(O_n)},\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)},\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{W},\tau,\boldsymbol{\alpha}\right) \\
&\times\log\left(\frac{\mathrm{p}\left(\boldsymbol{x}_1^{(O_1)},\ldots,\boldsymbol{x}_n^{(O_n)},\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)},\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{W},\tau,\boldsymbol{\alpha}\right)}{\mathrm{q}\left(\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)},\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{W},\tau,\boldsymbol{\alpha}\right)}\right) \\
&\times\mathrm{d}\boldsymbol{x}_1^{(M_1)}\ldots\mathrm{d}\boldsymbol{x}_n^{(M_n)}\mathrm{d}\boldsymbol{Z}\,\mathrm{d}\boldsymbol{\mu}\mathrm{d}\boldsymbol{W}\,\mathrm{d}\tau\mathrm{d}\boldsymbol{\alpha} \qquad \text{(B.117)} \\
=&\mathbb{E}\left[\log\frac{\mathrm{p}\left(\boldsymbol{x}_1^{(O_1)},\ldots,\boldsymbol{x}_n^{(O_n)},\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)},\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{W},\tau,\boldsymbol{\alpha}\right)}{\mathrm{q}\left(\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)},\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{W},\tau,\boldsymbol{\alpha}\right)}\right] \qquad \text{(B.118)} \\
=&\mathbb{E}\left[\log\mathrm{p}\left(\boldsymbol{x}_1^{(O_1)},\ldots,\boldsymbol{x}_n^{(O_n)},\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)},\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{W},\tau,\boldsymbol{\alpha}\right)\right] \\
&-\mathbb{E}\left[\log\mathrm{q}\left(\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)},z,\boldsymbol{\mu},\boldsymbol{W},\tau,\boldsymbol{\alpha}\right)\right] \qquad \text{(B.119)} \\
=&\mathbb{E}\left[\log\mathrm{p}\left(\boldsymbol{x}_1^{(O_1)},\ldots,\boldsymbol{x}_n^{(O_n)}|\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{W},\tau\right)\right] \\
&+\mathbb{E}\left[\log\mathrm{p}\left(\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)}|\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{W},\tau\right)\right]+\mathbb{E}\left[\log\mathrm{p}(\boldsymbol{Z})\right] \\
&+\mathbb{E}\left[\log\mathrm{p}(\boldsymbol{\mu}|\tau)\right]+\mathbb{E}\left[\log\mathrm{p}(\boldsymbol{W}|\tau,\boldsymbol{\alpha})\right]+\mathbb{E}\left[\log\mathrm{p}(\tau)\right]+\mathbb{E}\left[\log\mathrm{p}(\boldsymbol{\alpha})\right] \\
&-\mathbb{E}\left[\log\mathrm{q}\left(\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)}\right)\right]-\mathbb{E}\left[\log\mathrm{q}(\boldsymbol{Z})\right]-\mathbb{E}\left[\log\mathrm{q}(\boldsymbol{\mu},\boldsymbol{W},\tau)\right] \\
&-\mathbb{E}\left[\log\mathrm{q}(\boldsymbol{\alpha})\right], \qquad \text{(B.120)}
\end{aligned}$$

where the expectations are taken over all variables with respect to the variational distribution q. When the integration is performed over only a subset of the variables, a subscript notation is used, e.g. $\mathbb{E}_z\left[\cdot\right]$.

Optimal distributions q^*

in the following sections we will use a general solution for the optimal distributions q^*. According to Equation (B.116), we may subset the variables as $\mathcal{V}_1 = \left\{\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)}\right\}$, $\mathcal{V}_2 = \{\boldsymbol{Z}\}$, $\mathcal{V}_3 = \{\boldsymbol{\mu},\boldsymbol{W},\tau\}$, and $\mathcal{V}_4 = \{\boldsymbol{\alpha}\}$ dictated by the assumed independence structure in the variational approximation. In order to obtain the form of the optimal distributions for each subset $\mathcal{V}_l$, $l = 1,\ldots,4$ one uses the following equation

$$\log\mathrm{q}^*(\mathcal{V}_l) = \mathbb{E}_{\mathcal{V}_{\neg l}}\left[\log\mathrm{p}\left(\boldsymbol{x}_1^{(O_1)},\ldots,\boldsymbol{x}_n^{(O_n)},\boldsymbol{x}_1^{(M_1)},\ldots,\boldsymbol{x}_n^{(M_n)},\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{W},\tau,\boldsymbol{\alpha}\right)\right]+\text{const}, \qquad \text{(B.121)}$$

where $\neg l$ indicates all indices not equal to l. This means that the expectation is computed over the variables which are not in the subset $\mathcal{V}_l$. In addition, terms from Equation (B.96) that do not depend on the variables in $\mathcal{V}_l$ can be absorbed into the constant term

of Equation (B.121). We will use this result when deriving the optimal distributions in each of the next sections.

Form of $\mathrm{q}^*(\boldsymbol{\mu}, \boldsymbol{W}, \tau)$

Only including the terms from Equation (B.96) that include $\mathcal{V}_3 = \{\boldsymbol{\mu}, \boldsymbol{W}, \tau\}$, and dropping subscripts from the expectation notation when the term inside the expectation does not depend upon on that particular variable, we have

$$\begin{aligned}\log \mathrm{q}^*(\boldsymbol{\mu}, \boldsymbol{W}, \tau) =& \mathbb{E}_{\boldsymbol{Z}}\left[\log \mathrm{p}\left(\boldsymbol{x}_1^{(O_1)}, \ldots, \boldsymbol{x}_n^{(O_n)} | \boldsymbol{Z}, \boldsymbol{W}, \boldsymbol{\mu}, \tau\right)\right] \\ &+ \mathbb{E}_{\boldsymbol{x}_1^{(M_1)}, \ldots, \boldsymbol{x}_n^{(M_n)}, \boldsymbol{Z}}\left[\log \mathrm{p}\left(\boldsymbol{x}_1^{(M_1)}, \ldots, \boldsymbol{x}_n^{(M_n)} | \boldsymbol{Z}, \boldsymbol{W}, \boldsymbol{\mu}, \tau\right)\right] \\ &+ \log \mathrm{p}(\boldsymbol{\mu}|\tau) + \mathbb{E}_{\boldsymbol{\alpha}}\left[\log \mathrm{p}(\boldsymbol{W}|\tau, \boldsymbol{\alpha})\right] + \log \mathrm{p}(\tau) + \text{const.} \end{aligned} \tag{B.122}$$

Substituting in the expressions from Equations (B.101), (B.104), (B.108), (B.111), and (B.113) and moving terms not dependent on $\boldsymbol{\mu}, \boldsymbol{W}, \tau$ into the constant, we get

$$\begin{aligned}\log \mathrm{q}^*(\boldsymbol{\mu}, \boldsymbol{W}, \tau) =& -\frac{\tau}{2}\sum_{j=1}^{n}\Big(\boldsymbol{x}_j^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} + 2\boldsymbol{\mu}^{(O_j)\top}\boldsymbol{W}^{(O_j)}\mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right] \\ &-2\mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j^\top\right]\boldsymbol{W}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} - 2\boldsymbol{\mu}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} \\ &+\mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j^\top\boldsymbol{W}^{(O_j)\top}\boldsymbol{W}^{(O_j)}\boldsymbol{z}_j\right] + \boldsymbol{\mu}^{(O_j)\top}\boldsymbol{\mu}^{(O_j)}\Big) \\ &-\frac{\tau}{2}\sum_{j=1}^{n}\Big(\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)\top}\boldsymbol{x}_j^{(M_j)}\right] + 2\boldsymbol{\mu}^{(M_j)\top}\boldsymbol{W}^{(M_j)}\mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right] \\ &-2\mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j^\top\right]\boldsymbol{W}^{(M_j)\top}\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right] \\ &-2\boldsymbol{\mu}^{(M_j)\top}\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right] + \mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j^\top\boldsymbol{W}^{(M_j)\top}\boldsymbol{W}^{(M_j)}\boldsymbol{z}_j\right] \\ &+\boldsymbol{\mu}^{(M_j)\top}\boldsymbol{\mu}^{(M_j)}\Big) - \frac{\gamma_{\mu_0}\tau}{2}\left(\boldsymbol{\mu}^\top\boldsymbol{\mu} - 2\boldsymbol{\mu}^\top\overline{\boldsymbol{\mu}}_0 + \overline{\boldsymbol{\mu}}_0^\top\overline{\boldsymbol{\mu}}_0\right) \\ &+\sum_{g=1}^{q}\left(\frac{p}{2}\log\tau - \frac{\mathbb{E}_{\alpha_g}\left[\alpha_g\right]\tau}{2}\boldsymbol{w}_g^\top\boldsymbol{w}_g\right) \\ &-\gamma_{\tau_0}\overline{\tau}_0^{-1}\tau + \left(\frac{p}{2}(n+1) + \gamma_{\tau_0} - 1\right)\log\tau + \text{const.} \end{aligned} \tag{B.123}$$

Rearranging gives

$$
\begin{aligned}
\log \mathrm{q}^*(\boldsymbol{\mu}, \boldsymbol{W}, \tau) = & \left(\frac{p}{2}(n+q+1) + \gamma_{\tau_0} - 1\right) \log \tau \\
& - \Bigg(\gamma_{\tau_0} \overline{\tau}_0^{-1} + \frac{1}{2} \sum_{j=1}^{n} \boldsymbol{x}_j^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} + \frac{1}{2} \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}} \left[\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{x}_j^{(M_j)} \right] \\
& + \frac{\gamma_{\boldsymbol{\mu}_0}}{2} \overline{\boldsymbol{\mu}}_0^{\top} \overline{\boldsymbol{\mu}}_0 \Bigg) \tau - \frac{\tau}{2} \Bigg(\sum_{j=1}^{n} \mathbb{E}_{z_j} \left[\boldsymbol{z}_j^{\top} \boldsymbol{W}^{(O_j)\top} \boldsymbol{W}^{(O_j)} \boldsymbol{z}_j \right] \\
& + \sum_{j=1}^{n} \mathbb{E}_{z_j} \left[\boldsymbol{z}_j^{\top} \boldsymbol{W}^{(M_j)\top} \boldsymbol{W}^{(M_j)} \boldsymbol{z}_j \right] + \sum_{g=1}^{q} \mathbb{E}_{\alpha_g} \left[\alpha_g \right] \boldsymbol{w}_g^{\top} \boldsymbol{w}_g \Bigg) \\
& + \tau \Bigg(\sum_{j=1}^{n} \mathbb{E}_{z_j} \left[\boldsymbol{z}_j^{\top} \right] \boldsymbol{W}^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} \\
& + \sum_{j=1}^{n} \mathbb{E}_{z_j} \left[\boldsymbol{z}_j^{\top} \right] \boldsymbol{W}^{(M_j)\top} \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}} \left[\boldsymbol{x}_j^{(M_j)} \right] \Bigg) \\
& - \frac{\tau}{2} \Bigg(\sum_{j=1}^{n} \boldsymbol{\mu}^{(O_j)\top} \boldsymbol{\mu}^{(O_j)} + \sum_{j=1}^{n} \boldsymbol{\mu}^{(M_j)\top} \boldsymbol{\mu}^{(M_j)} + \gamma_{\mu_0} \boldsymbol{\mu}^{\top} \boldsymbol{\mu} \Bigg) \\
& + \tau \Bigg(\sum_{j=1}^{n} \boldsymbol{\mu}^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} - \sum_{j=1}^{n} \boldsymbol{\mu}^{(O_j)\top} \boldsymbol{W}^{(O_j)} \mathbb{E}_{z_j} \left[\boldsymbol{z}_j \right] \\
& + \sum_{j=1}^{n} \boldsymbol{\mu}^{(M_j)\top} \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}} \left[\boldsymbol{x}_j^{(M_j)} \right] - \sum_{j=1}^{n} \boldsymbol{\mu}^{(M_j)\top} \boldsymbol{W}^{(M_j)} \mathbb{E}_{z_j} \left[\boldsymbol{z}_j \right] \\
& + \gamma_{\mu_0} \boldsymbol{\mu}^{\top} \overline{\boldsymbol{\mu}}_0 \Bigg) + \text{const.}
\end{aligned}
\tag{B.124}
$$

The following identities are then useful to simplify this expression:

$$\boldsymbol{\mu}^{(O_j)\top}\boldsymbol{\mu}^{(O_j)} + \boldsymbol{\mu}^{(M_j)\top}\boldsymbol{\mu}^{(M_j)} = \sum_{i=1}^{|O_j|}\mu_i^2 + \sum_{i=|O_j|+1}^{p}\mu_i^2 = \sum_{i=1}^{p}\mu_i^2 = \boldsymbol{\mu}^\top\boldsymbol{\mu} \tag{B.125}$$

$$\begin{aligned}\boldsymbol{W}^{(O_j)\top}\boldsymbol{W}^{(O_j)} + \boldsymbol{W}^{(M_j)\top}\boldsymbol{W}^{(M_j)} &= \sum_{i=1}^{|O_j|}\widetilde{\boldsymbol{w}}_i^\top\widetilde{\boldsymbol{w}}_i + \sum_{i=|O_j|+1}^{p}\widetilde{\boldsymbol{w}}_i^\top\widetilde{\boldsymbol{w}}_i \\ &= \sum_{i=1}^{p}\widetilde{\boldsymbol{w}}_i^\top\widetilde{\boldsymbol{w}}_i = \boldsymbol{W}^\top\boldsymbol{W}\end{aligned} \tag{B.126}$$

$$\boldsymbol{\mu}^{(O_j)\top}\boldsymbol{W}^{(O_j)} + \boldsymbol{\mu}^{(M_j)\top}\boldsymbol{W}^{(M_j)} = \begin{bmatrix}\sum_{i=1}^{|O_j|}\mu_i w_{i1} \\ \dots \\ \sum_{i=1}^{|O_j|}\mu_i w_{iq}\end{bmatrix}^\top + \begin{bmatrix}\sum_{i=|O_j|+1}^{p}\mu_i w_{i1} \\ \dots \\ \sum_{i=|O_j|+1}^{p}\mu_i w_{iq}\end{bmatrix}^\top \tag{B.127}$$

$$= \begin{bmatrix}\sum_{i=1}^{p}\mu_i w_{i1} \\ \dots \\ \sum_{i=1}^{p}\mu_i w_{iq}\end{bmatrix}^\top \tag{B.128}$$

$$= \boldsymbol{\mu}^\top\boldsymbol{W}. \tag{B.129}$$

Also setting

$$\widetilde{\boldsymbol{x}}_j = \begin{bmatrix}\boldsymbol{x}_j^{(O_j)} \\ \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right]\end{bmatrix}, \tag{B.130}$$

gives the identities:

$$\boldsymbol{\mu}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} + \boldsymbol{\mu}^{(M_j)\top}\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right] = \sum_{i=1}^{|O_j|}\mu_i\widetilde{x}_{ij} + \sum_{i=|O_j|+1}^{p}\mu_i\widetilde{x}_{ij} = \boldsymbol{\mu}^\top\widetilde{\boldsymbol{x}}_j \tag{B.131}$$

$$\boldsymbol{x}_j^{(O_j)\top}\boldsymbol{W}^{(O_j)} + \mathbb{E}_{\boldsymbol{x}^{(M_j)}}\left[\boldsymbol{x}^{(M_j)}\right]^\top\boldsymbol{W}^{(M_j)} = \begin{bmatrix}\sum_{i=1}^{|O_j|}\widetilde{x}_{ij}w_{i1} \\ \dots \\ \sum_{i=1}^{|O_j|}\widetilde{x}_{ij}w_{iq}\end{bmatrix}^\top + \begin{bmatrix}\sum_{i=|O_j|+1}^{p}\widetilde{x}_{ij}w_{i1} \\ \dots \\ \sum_{i=|O_j|+1}^{p}\widetilde{x}_{ij}w_{iq}\end{bmatrix}^\top \tag{B.132}$$

$$= \begin{bmatrix}\sum_{i=1}^{p}\widetilde{x}_{ij}w_{i1} \\ \dots \\ \sum_{i=1}^{p}\widetilde{x}_{ij}w_{iq}\end{bmatrix}^\top \tag{B.133}$$

$$= \widetilde{\boldsymbol{x}}_j^\top\boldsymbol{W}. \tag{B.134}$$

Using these identities, Equation (B.124) becomes

$$\begin{aligned}
\log \mathrm{q}^*(\boldsymbol{\mu}, \boldsymbol{W}, \tau) = & \left(\frac{p}{2}(n+q+1)+\gamma_{\tau_0}-1\right)\log\tau \\
& -\Bigg(\gamma_{\tau_0}\overline{\tau}_0^{-1}+\frac{1}{2}\sum_{j=1}^{n}\boldsymbol{x}_j^{(O_j)\top}\boldsymbol{x}_j^{(O_j)}+\frac{1}{2}\sum_{j=1}^{n}\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)\top}\boldsymbol{x}_j^{(M_j)}\right] \\
& +\frac{\gamma_{\mu_0}}{2}\overline{\boldsymbol{\mu}}_0^{\top}\overline{\boldsymbol{\mu}}_0\Bigg)\tau-\frac{\tau}{2}\Bigg(\sum_{j=1}^{n}\mathbb{E}_{z_j}\left[\boldsymbol{z}_j^{\top}\boldsymbol{W}^{\top}\boldsymbol{W}\boldsymbol{z}_j\right] \\
& +\sum_{g=1}^{q}\mathbb{E}_{\alpha_g}\left[\alpha_g\right]\boldsymbol{w}_g^{\top}\boldsymbol{w}_g\Bigg)+\tau\sum_{j=1}^{n}\mathbb{E}_{z_j}\left[\boldsymbol{z}_j\right]^{\top}\boldsymbol{W}^{\top}\widetilde{\boldsymbol{x}}_j \\
& -\frac{\tau}{2}(\gamma_{\mu_0}+n)\boldsymbol{\mu}^{\top}\boldsymbol{\mu}+\tau\Bigg(\sum_{j=1}^{n}\boldsymbol{\mu}^{\top}\widetilde{\boldsymbol{x}}_j-\boldsymbol{\mu}^{\top}\boldsymbol{W}\mathbb{E}_{z_j}\left[\boldsymbol{z}_j\right] \\
& +\gamma_{\mu_0}\boldsymbol{\mu}^{\top}\overline{\boldsymbol{\mu}}_0\Bigg)+\text{const} \qquad \text{(B.135)} \\
= & \left(\frac{p}{2}(n+q+1)+\gamma_{\tau_0}-1\right)\log\tau \\
& -\Bigg(\gamma_{\tau_0}\overline{\tau}_0^{-1}+\frac{1}{2}\sum_{j=1}^{n}\boldsymbol{x}_j^{(O_j)\top}\boldsymbol{x}_j^{(O_j)}+\frac{1}{2}\sum_{j=1}^{n}\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)\top}\boldsymbol{x}_j^{(M_j)}\right] \\
& +\frac{\gamma_{\mu_0}}{2}\overline{\boldsymbol{\mu}}_0^{\top}\overline{\boldsymbol{\mu}}_0\Bigg)\tau-\frac{\tau}{2}\Bigg(\sum_{j=1}^{n}\mathbb{E}_{z_j}\left[\boldsymbol{z}_j^{\top}\boldsymbol{W}^{\top}\boldsymbol{W}\boldsymbol{z}_j\right] \\
& +\sum_{g=1}^{q}\mathbb{E}_{\alpha_g}\left[\alpha_g\right]\boldsymbol{w}_g^{\top}\boldsymbol{w}_g\Bigg)+\tau\sum_{j=1}^{n}\mathbb{E}_{z_j}\left[\boldsymbol{z}_j\right]^{\top}\boldsymbol{W}^{\top}\widetilde{\boldsymbol{x}}_j \\
& -\frac{\tau}{2}(\gamma_{\mu_0}+n)\boldsymbol{\mu}^{\top}\boldsymbol{\mu}+\tau\boldsymbol{\mu}^{\top}\Bigg(\sum_{j=1}^{n}\left(\widetilde{\boldsymbol{x}}_j-\boldsymbol{W}\mathbb{E}_{z_j}\left[\boldsymbol{z}_j\right]\right)+\gamma_{\mu_0}\overline{\boldsymbol{\mu}}_0\Bigg) \\
& +\text{const.} \qquad \text{(B.136)}
\end{aligned}$$

Noting that the logarithm of the density $\mathcal{N}(\boldsymbol{W}\boldsymbol{s}_{\boldsymbol{\mu}}+\boldsymbol{m}_{\boldsymbol{\mu}},(\gamma_{\boldsymbol{\mu}}\tau)^{-1}\boldsymbol{I}_{p\times p})$ can be expressed as

$$\frac{p}{2}\log\tau-\frac{\gamma_{\mu}\tau}{2}(\boldsymbol{\mu}^{\top}\boldsymbol{\mu}-2\boldsymbol{\mu}^{\top}(\boldsymbol{W}\boldsymbol{s}_{\boldsymbol{\mu}}+\boldsymbol{m}_{\boldsymbol{\mu}})+(\boldsymbol{W}\boldsymbol{s}_{\boldsymbol{\mu}}+\boldsymbol{m}_{\boldsymbol{\mu}})^{\top}(\boldsymbol{W}\boldsymbol{s}_{\boldsymbol{\mu}}+\boldsymbol{m}_{\boldsymbol{\mu}}))+\text{const}, \qquad \text{(B.137)}$$

comparing the $\boldsymbol{\mu}^\top\boldsymbol{\mu}$ and $\boldsymbol{\mu}^\top$ terms in Equations (B.136) and (B.137) gives

$$\begin{aligned}-\frac{\tau}{2}(\gamma_{\mu_0}+n)\boldsymbol{\mu}^\top\boldsymbol{\mu}+\tau\boldsymbol{\mu}^\top\left(\sum_{j=1}^{n}\left(\widetilde{\boldsymbol{x}}_j-\boldsymbol{W}\mathbb{E}_{z_j}\left[z_j\right]\right)+\gamma_{\mu_0}\overline{\boldsymbol{\mu}}_0\right)\\ =-\frac{\gamma_\mu\tau}{2}\boldsymbol{\mu}^\top\boldsymbol{\mu}+\gamma_\mu\tau\boldsymbol{\mu}^\top(\boldsymbol{W}\boldsymbol{s}_\mu+\boldsymbol{m}_\mu)\end{aligned} \tag{B.138}$$

so that

$$\gamma_\mu=\gamma_{\mu_0}+n, \tag{B.139}$$

$$\boldsymbol{s}_\mu=-\frac{1}{\gamma_\mu}\sum_{j=1}^{n}\mathbb{E}_{z_j}\left[z_j\right], \tag{B.140}$$

$$\boldsymbol{m}_\mu=\frac{1}{\gamma_\mu}\left(\gamma_{\mu_0}\overline{\boldsymbol{\mu}}_0+\sum_{j=1}^{n}\widetilde{\boldsymbol{x}}_j\right). \tag{B.141}$$

It can then be seen that $\mathrm{q}^*(\boldsymbol{\mu}|\boldsymbol{W},\tau)$ factors from the distribution $\mathrm{q}^*(\boldsymbol{\mu},\boldsymbol{W},\tau)$ where it takes the form

$$\mathrm{q}^*(\boldsymbol{\mu}|\boldsymbol{W},\tau)=\mathcal{N}(\boldsymbol{W}\boldsymbol{s}_\mu+\boldsymbol{m}_\mu,(\gamma_\mu\tau)^{-1}\boldsymbol{I}_{p\times p}). \tag{B.142}$$

Subtracting this distribution from Equation (B.136) gives

$$
\begin{aligned}
&\log \mathrm{q}^*(\boldsymbol{\mu}, \boldsymbol{W}, \tau) \\
&-\mathrm{q}^*(\boldsymbol{\mu}|\boldsymbol{W}, \tau) = \left(\frac{p}{2}(n+q+1) + \gamma_{\tau_0} - 1\right) \log \tau \\
&\qquad - \Bigg(\gamma_{\tau_0} \overline{\tau}_0^{-1} + \frac{1}{2} \sum_{j=1}^{n} \boldsymbol{x}_j^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} + \frac{1}{2} \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}} \left[\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{x}_j^{(M_j)} \right] \\
&\qquad + \frac{\gamma_{\boldsymbol{\mu}_0}}{2} \overline{\boldsymbol{\mu}}_0^{\top} \overline{\boldsymbol{\mu}}_0 \Bigg) \tau - \frac{\tau}{2} \Bigg(\sum_{j=1}^{n} \mathbb{E}_{z_j} \left[z_j^{\top} \boldsymbol{W}^{\top} \boldsymbol{W} z_j \right] \\
&\qquad + \sum_{g=1}^{q} \mathbb{E}_{\alpha_g} \left[\alpha_g \right] \boldsymbol{w}_g^{\top} \boldsymbol{w}_g \Bigg) + \tau \sum_{j=1}^{n} \mathbb{E}_{z_j} \left[z_j \right]^{\top} \boldsymbol{W}^{\top} \widetilde{\boldsymbol{x}}_j \\
&\qquad - \frac{p}{2} \log \tau + \frac{\gamma_{\mu} \tau}{2} (\boldsymbol{W} \boldsymbol{s}_{\boldsymbol{\mu}} + \boldsymbol{m}_{\boldsymbol{\mu}})^{\top} (\boldsymbol{W} \boldsymbol{s}_{\boldsymbol{\mu}} + \boldsymbol{m}_{\boldsymbol{\mu}}) + \text{const} \qquad \text{(B.143)} \\
&= \left(\frac{p}{2}(n+q) + \gamma_{\tau_0} - 1\right) \log \tau - \Bigg(\gamma_{\tau_0} \overline{\tau}_0^{-1} + \frac{1}{2} \sum_{j=1}^{n} \boldsymbol{x}_j^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} \\
&\qquad + \frac{1}{2} \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}} \left[\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{x}_j^{(M_j)} \right] + \frac{\gamma_{\boldsymbol{\mu}_0}}{2} \overline{\boldsymbol{\mu}}_0^{\top} \overline{\boldsymbol{\mu}}_0 - \frac{\gamma_{\boldsymbol{\mu}}}{2} \boldsymbol{m}_{\boldsymbol{\mu}}^{\top} \boldsymbol{m}_{\boldsymbol{\mu}} \Bigg) \tau \\
&\qquad - \frac{\tau}{2} \Bigg(\sum_{j=1}^{n} \mathbb{E}_{z_j} \left[z_j^{\top} \boldsymbol{W}^{\top} \boldsymbol{W} z_j \right] - \gamma_{\boldsymbol{\mu}} \boldsymbol{s}_{\boldsymbol{\mu}}^{\top} \boldsymbol{W}^{\top} \boldsymbol{W} \boldsymbol{s}_{\boldsymbol{\mu}} \\
&\qquad + \sum_{g=1}^{q} \mathbb{E}_{\alpha_g} \left[\alpha_g \right] \boldsymbol{w}_g^{\top} \boldsymbol{w}_g \Bigg) + \tau \Bigg(\sum_{j=1}^{n} \mathbb{E}_{z_j} \left[z_j \right]^{\top} \boldsymbol{W}^{\top} \widetilde{\boldsymbol{x}}_j \\
&\qquad + \gamma_{\boldsymbol{\mu}} \boldsymbol{m}_{\boldsymbol{\mu}}^{\top} \boldsymbol{W} \boldsymbol{s}_{\boldsymbol{\mu}} \Bigg) + \text{const.} \qquad \text{(B.144)}
\end{aligned}
$$

Noting the identities

$$\sum_{g=1}^{q} \mathbb{E}_{\alpha_g}\left[\alpha_g\right] \boldsymbol{w}_g^\top \boldsymbol{w}_g = \sum_{g=1}^{q} \mathbb{E}_{\alpha_g}\left[\alpha_g\right] \sum_{i=1}^{p} w_{ig}^2 \tag{B.145}$$

$$= \sum_{i=1}^{p} \sum_{g=1}^{q} \mathbb{E}_{\alpha_g}\left[\alpha_g\right] w_{ig}^2 \tag{B.146}$$

$$= \sum_{i=1}^{p} \widetilde{\boldsymbol{w}}_i \mathrm{diag}\left(\mathbb{E}_{\boldsymbol{\alpha}}\left[\boldsymbol{\alpha}\right]\right) \widetilde{\boldsymbol{w}}_i^\top, \tag{B.147}$$

$$\boldsymbol{b}^\top \boldsymbol{W}^\top \boldsymbol{W} \boldsymbol{b} = \boldsymbol{b}^\top \left(\sum_{i=1}^{p} \widetilde{\boldsymbol{w}}_i^\top \widetilde{\boldsymbol{w}}_i \right) \boldsymbol{b} \tag{B.148}$$

$$= \sum_{i=1}^{p} \mathrm{Tr}\left[\boldsymbol{b}^\top \widetilde{\boldsymbol{w}}_i^\top \widetilde{\boldsymbol{w}}_i \boldsymbol{b}\right] \tag{B.149}$$

$$= \sum_{i=1}^{p} \mathrm{Tr}\left[\widetilde{\boldsymbol{w}}_i \boldsymbol{b}\boldsymbol{b}^\top \widetilde{\boldsymbol{w}}_i^\top\right] \tag{B.150}$$

$$= \sum_{i=1}^{p} \widetilde{\boldsymbol{w}}_i \boldsymbol{b}\boldsymbol{b}^\top \widetilde{\boldsymbol{w}}_i^\top, \tag{B.151}$$

$$\boldsymbol{a}^\top \boldsymbol{W} \boldsymbol{b} = \sum_{i=1}^{p} a_i \widetilde{\boldsymbol{w}}_i \boldsymbol{b}, \tag{B.152}$$

where again $\widetilde{\boldsymbol{w}}_i$ denotes the i-th row-vector of $\boldsymbol{W}$ and $\mathrm{diag}\left(\mathbb{E}_{\boldsymbol{\alpha}}\left[\boldsymbol{\alpha}\right]\right)$ is a diagonal matrix whose non-zero elements are equal to $\mathbb{E}_{\alpha_j}\left[\alpha_j\right]$. Using these identities in Equation (B.144) gives

$$\log \mathrm{q}^*(\boldsymbol{\mu}, \boldsymbol{W}, \tau) \tag{B.153}$$

$$\begin{aligned}
-\mathrm{q}^*(\boldsymbol{\mu}|\boldsymbol{W}, \tau) = &\left(\frac{p}{2}(n+q) + \gamma_{\tau_0} - 1\right) \log \tau \\
&- \Bigg(\gamma_{\tau_0} \overline{\tau}_0^{-1} + \frac{1}{2} \sum_{j=1}^{n} \boldsymbol{x}_j^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} + \frac{1}{2} \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}} \left[\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{x}_j^{(M_j)} \right] \\
&+ \frac{\gamma_{\mu_0}}{2} \overline{\boldsymbol{\mu}}_0^\top \overline{\boldsymbol{\mu}}_0 - \frac{\gamma_{\boldsymbol{\mu}}}{2} \boldsymbol{m}_{\boldsymbol{\mu}}^\top \boldsymbol{m}_{\boldsymbol{\mu}} \Bigg) \tau - \frac{\tau}{2} \sum_{i=1}^{p} \widetilde{\boldsymbol{w}}_i \Bigg(\frac{1}{2} \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{z}_j} \left[\boldsymbol{z}_j \boldsymbol{z}_j^\top \right] \\
&- \gamma_{\boldsymbol{\mu}} \boldsymbol{s}_{\boldsymbol{\mu}} \boldsymbol{s}_{\boldsymbol{\mu}}^\top + \mathrm{diag}\left(\mathbb{E}_{\boldsymbol{\alpha}}\left[\boldsymbol{\alpha}\right]\right) \Bigg) \widetilde{\boldsymbol{w}}_i^\top \\
&+ \tau \sum_{i=1}^{p} \widetilde{\boldsymbol{w}}_i \left(\sum_{j=1}^{n} \widetilde{x}_{ij} \mathbb{E}_{\boldsymbol{z}_j} \left[\boldsymbol{z}_j \right] + \gamma_{\boldsymbol{\mu}} m_{\mu i} \boldsymbol{s}_{\boldsymbol{\mu}} \right) + \mathrm{const.}
\end{aligned} \tag{B.154}$$

Note that the logarithm of the density $\prod_{i=1}^{p} \mathcal{N}(\boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i}, (\tau \boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}})^{-1})$ can be expressed as

$$\sum_{i=1}^{p} \left[-\frac{\tau}{2} \left(\widetilde{\boldsymbol{w}}_i^\top - \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i} \right)^\top \boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}} \left(\widetilde{\boldsymbol{w}}_i^\top - \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i} \right) \right] - \sum_{i=1}^{p} \frac{1}{2} \log |(\tau \boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}})^{-1}| + \text{const} \tag{B.155}$$

$$= \sum_{i=1}^{p} \left[-\frac{\tau}{2} \left(\widetilde{\boldsymbol{w}}_i \boldsymbol{\Lambda}_{\boldsymbol{w}} \widetilde{\boldsymbol{w}}_i^\top - 2 \widetilde{\boldsymbol{w}}_i \boldsymbol{\Lambda}_{\boldsymbol{w}} \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i}^\top + \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i}^\top \boldsymbol{\Lambda}_{\boldsymbol{w}} \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i} \right) \right] + \frac{pq}{2} \log \tau + \text{const}. \tag{B.156}$$

Comparing the $\widetilde{\boldsymbol{w}}_i \widetilde{\boldsymbol{w}}_i^\top$ and $\widetilde{\boldsymbol{w}}_i$ terms from Equations (B.154) and (B.156) gives

$$\boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}} = \sum_{j=1}^{n} \mathbb{E}_{z_j} \left[\boldsymbol{z}_j \boldsymbol{z}_j^\top \right] - \gamma_{\boldsymbol{\mu}} \boldsymbol{s}_{\boldsymbol{\mu}} \boldsymbol{s}_{\boldsymbol{\mu}}^\top + \text{diag}\left(\mathbb{E}_{\boldsymbol{\alpha}} [\boldsymbol{\alpha}] \right) \tag{B.157}$$

$$\boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i} = \boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}}^{-1} \left(\sum_{j=1}^{n} \widetilde{x}_{ij} \mathbb{E}_{z_j} \left[\boldsymbol{z}_j \right] + \gamma_{\boldsymbol{\mu}} m_{\boldsymbol{\mu} i} \boldsymbol{s}_{\boldsymbol{\mu}} \right). \tag{B.158}$$

It can then be seen that $\mathrm{q}^*(\boldsymbol{W}|\tau)$ factors from the distribution $\mathrm{q}^*(\boldsymbol{\mu}, \boldsymbol{W}, \tau)$ where it takes the form

$$\mathrm{q}^*(\boldsymbol{W}|\tau) = \prod_{i=1}^{p} \mathcal{N}(\boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i}, (\tau \boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}})^{-1}). \tag{B.159}$$

Subtracting this distribution from Equation (B.154) gives

$$\log \mathrm{q}^*(\boldsymbol{\mu}, \boldsymbol{W}, \tau) - \mathrm{q}^*(\boldsymbol{\mu}|\boldsymbol{W}, \tau) - \mathrm{q}^*(\boldsymbol{W}|\tau) \tag{B.160}$$

$$= \left(\frac{p}{2}(n+q) + \gamma_{\tau_0} - 1 \right) \log \tau$$

$$- \left(\gamma_{\tau_0} \overline{\tau}_0^{-1} + \frac{1}{2} \sum_{j=1}^{n} \boldsymbol{x}_j^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} + \frac{1}{2} \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}} \left[\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{x}_j^{(M_j)} \right] \right. \tag{B.161}$$

$$\left. + \frac{\gamma_{\mu_0}}{2} \overline{\boldsymbol{\mu}}_0^\top \overline{\boldsymbol{\mu}}_0 - \frac{\gamma_{\boldsymbol{\mu}}}{2} \boldsymbol{m}_{\boldsymbol{\mu}}^\top \boldsymbol{m}_{\boldsymbol{\mu}} \right) \tau$$

$$+ \frac{\tau}{2} \sum_{i=1}^{p} \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i}^\top \boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}} \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i} - \frac{pq}{2} \log \tau + \text{const} \tag{B.162}$$

$$= \left(\frac{np}{2} + \gamma_{\tau_0} - 1 \right) \log \tau - \left(\gamma_{\tau_0} \overline{\tau}_0^{-1} + \frac{1}{2} \sum_{j=1}^{n} \boldsymbol{x}_j^{(O_j)\top} \boldsymbol{x}_j^{(O_j)} \right.$$

$$+ \frac{1}{2} \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{x}^{(M_j)}} \left[\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{x}_j^{(M_j)} \right] + \frac{\gamma_{\mu_0}}{2} \overline{\boldsymbol{\mu}}_0^\top \overline{\boldsymbol{\mu}}_0 - \frac{\gamma_{\boldsymbol{\mu}}}{2} \boldsymbol{m}_{\boldsymbol{\mu}}^\top \boldsymbol{m}_{\boldsymbol{\mu}}$$

$$\left. - \frac{1}{2} \sum_{i=1}^{p} \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i}^\top \boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}} \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i} \right) \tau + \text{const}. \tag{B.163}$$

Note that the logarithm of the density $\Gamma(\overline{\tau}, \gamma_\tau)$ can be expressed as

$$(\gamma_\tau - 1)\log\tau - \gamma_\tau\overline{\tau}\tau + \text{const.} \tag{B.164}$$

Comparing the $\log\tau$ and τ terms from Equations (B.163) and (B.164) gives

$$\gamma_\tau = \frac{np}{2} + \gamma_{\tau_0} \tag{B.165}$$

$$\begin{aligned}\overline{\tau} = \gamma_\tau \Bigg(& \gamma_{\tau_0}\overline{\tau}_0^{-1} + \frac{1}{2}\sum_{j=1}^{n} \boldsymbol{x}_j^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} + \frac{1}{2}\sum_{j=1}^{n}\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)\top}\boldsymbol{x}_j^{(M_j)}\right] \\ & + \frac{\gamma_{\mu_0}}{2}\overline{\boldsymbol{\mu}}_0^\top\overline{\boldsymbol{\mu}}_0 - \frac{\gamma_{\boldsymbol{\mu}}}{2}\boldsymbol{m}_{\boldsymbol{\mu}}^\top\boldsymbol{m}_{\boldsymbol{\mu}} - \frac{1}{2}\sum_{i=1}^{p}\boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i}^\top\boldsymbol{\Lambda}_{\widetilde{\boldsymbol{w}}}\boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i}\Bigg)^{-1}.\end{aligned} \tag{B.166}$$

Finally, it can be seen that $\mathrm{q}^*(\tau)$ factors from the distribution $\mathrm{q}^*(\boldsymbol{\mu}, \boldsymbol{W}, \tau)$ where it takes the form

$$\mathrm{q}^*(\tau) = \Gamma(\overline{\tau}, \gamma_\tau). \tag{B.167}$$

Subtracting this from Equation (B.163) leaves only the constant term, indicating that the variational approximation takes the factored form

$$\mathrm{q}^*(\boldsymbol{\mu}, \boldsymbol{W}, \tau) = \mathrm{q}^*(\boldsymbol{\mu}|\boldsymbol{W}, \tau)\mathrm{q}^*(\boldsymbol{W}|\tau)\mathrm{q}^*(\tau). \tag{B.168}$$

Form of $\mathrm{q}^*(\boldsymbol{\alpha})$

Only including the terms from Equation (B.96) that include $\mathcal{V}_4 = \{\boldsymbol{\alpha}\}$, and dropping subscripts from the integration notation when the integrand does not depend upon on that particular variable, we have

$$\log\mathrm{q}^*(\boldsymbol{\alpha}) = \mathbb{E}_{\boldsymbol{W},\tau}\left[\log\mathrm{p}(\boldsymbol{W}|\tau, \boldsymbol{\alpha})\right] + \log\mathrm{p}(\boldsymbol{\alpha}) + \text{const.} \tag{B.169}$$

Substituting in the expressions from Equations (B.111) and (B.115) and moving terms not dependent on $\boldsymbol{\alpha}$ into the constant term, we get

$$\log\mathrm{q}^*(\boldsymbol{\alpha}) = \sum_{j=1}^{q}\frac{p}{2}\log\alpha_j - \frac{\alpha_j}{2}\mathbb{E}_{\boldsymbol{w}_j,\tau}\left[\tau\boldsymbol{w}_j^\top\boldsymbol{w}_j\right] - \gamma_{\alpha_0}\overline{\alpha}_0^{-1}\alpha_j + (\gamma_{\alpha_0} - 1)\log\alpha_j + \text{const} \tag{B.170}$$

$$= \sum_{j=1}^{q}\left(\gamma_{\alpha_0} + \frac{p}{2} - 1\right)\log\alpha_j - \left(\gamma_{\alpha_0}\overline{\alpha}_0^{-1} + \frac{1}{2}\mathbb{E}_{\boldsymbol{w}_j,\tau}\left[\tau\boldsymbol{w}_j^\top\boldsymbol{w}_j\right]\right)\alpha_j. \tag{B.171}$$

Note that the logarithm of the density $\prod_{j=1}^{q} \Gamma(\overline{\alpha}_j, \gamma_\alpha)$ can be expressed as

$$\sum_{j=1}^{q} (\gamma_\alpha - 1) \log \alpha_j - \gamma_\alpha \overline{\alpha}^{-1} \alpha_j + \text{const.} \tag{B.172}$$

Comparing the terms for α_j and $\log \alpha_j$ from Equations (B.171) and (B.172) gives

$$\gamma_\alpha = \gamma_{\alpha_0} + \frac{p}{2} \tag{B.173}$$

$$\overline{\alpha}_j = \gamma_\alpha \left(\gamma_{\alpha_0} \overline{\alpha}_0^{-1} + \frac{1}{2} \mathbb{E}_{\boldsymbol{w}_j, \tau} \left[\tau \boldsymbol{w}_j^\top \boldsymbol{w}_j \right] \right)^{-1}, \tag{B.174}$$

and so

$$\mathrm{q}^*(\boldsymbol{\alpha}) = \prod_{j=1}^{q} \Gamma(\overline{\alpha}_j, \gamma_\alpha). \tag{B.175}$$

Form of $\mathrm{q}^*(\boldsymbol{Z})$

Only including the terms from Equation (B.96) that include $\mathcal{V}_2 = \{\boldsymbol{Z}\}$, and dropping subscripts from the integration notation when the integrand does not depend upon on that particular variable, we have

$$\begin{aligned} \log \mathrm{q}^*(\boldsymbol{Z}) = \sum_{j=1}^{n} \log \mathrm{q}^*(\boldsymbol{z}_j) = \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{W}, \boldsymbol{\mu}, \tau} \left[\log \mathrm{p} \left(\boldsymbol{x}_j^{(O_j)} | \boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau \right) \right] \\ + \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{x}_j^{(M_j)}, \boldsymbol{W}, \boldsymbol{\mu}, \tau} \left[\log \mathrm{p} \left(\boldsymbol{x}_j^{(M_j)} | \boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau \right) \right] + \sum_{j=1}^{n} \log \mathrm{p}(\boldsymbol{z}_j) \\ + \text{const.} \end{aligned} \tag{B.176}$$

Substituting in the expressions from Equations (B.99), (B.102), and (B.105) and moving terms not dependent on $\boldsymbol{Z}$ into the constant term, we get

$$\begin{aligned}
\log \mathrm{q}^*(\boldsymbol{Z}) = &\sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{W},\boldsymbol{\mu},\tau}\left[-\frac{\tau}{2}\left(-2\boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} + 2\boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top}\boldsymbol{\mu}^{(O_j)}\right.\right.\\
&\left.\left.+\boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top}\boldsymbol{W}^{(O_j)}\boldsymbol{z}_j\right)\right]\\
&+\sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{x}_j^{(M_j)},\boldsymbol{W},\boldsymbol{\mu},\tau}\left[-\frac{\tau}{2}\left(-2\boldsymbol{z}_j^\top \boldsymbol{W}^{(M_j)\top}\boldsymbol{x}_j^{(M_j)} + 2\boldsymbol{z}_j^\top \boldsymbol{W}^{(M_j)\top}\boldsymbol{\mu}^{(M_j)}\right.\right.\\
&\left.\left.+\boldsymbol{z}_j^\top \boldsymbol{W}^{(M_j)\top}\boldsymbol{W}^{(M_j)}\boldsymbol{z}_j\right)\right] - \frac{1}{2}\sum_{j=1}^{n}\boldsymbol{z}_j^\top \boldsymbol{z}_j + \text{const} \qquad \text{(B.177)}\\
= &\sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{W},\boldsymbol{\mu},\tau}\left[-\frac{\tau}{2}\left(-2\boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top}\boldsymbol{x}_j^{(O_j)} + 2\boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top}\boldsymbol{\mu}^{(O_j)}\right.\right.\\
&\left.\left.+\boldsymbol{z}_j^\top \boldsymbol{W}^{(O_j)\top}\boldsymbol{W}^{(O_j)}\boldsymbol{z}_j\right)\right]\\
&+\sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{W},\boldsymbol{\mu},\tau}\left[-\frac{\tau}{2}\left(-2\boldsymbol{z}^\top \boldsymbol{W}^{(M_j)\top}\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)}\right] + 2\boldsymbol{z}_j^\top \boldsymbol{W}^{(M_j)\top}\boldsymbol{\mu}^{(M_j)}\right.\right.\\
&\left.\left.+\ \boldsymbol{z}_j^\top \boldsymbol{W}^{(M_j)\top}\boldsymbol{W}^{(M_j)}\boldsymbol{z}_j\right)\right] - \frac{1}{2}\sum_{j=1}^{n}\boldsymbol{z}_j^\top \boldsymbol{z}_j + \text{const} \qquad \text{(B.178)}\\
= &\sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{W},\boldsymbol{\mu},\tau}\left[-\frac{\tau}{2}\left(-2\boldsymbol{z}_j^\top \boldsymbol{W}^\top \widetilde{\boldsymbol{x}}_j + 2\boldsymbol{z}_j^\top \boldsymbol{W}^\top \boldsymbol{\mu} + \boldsymbol{z}_j^\top \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{z}_j\right)\right]\\
&-\frac{1}{2}\sum_{j=1}^{n}\boldsymbol{z}_j^\top \boldsymbol{z}_j + \text{const} \qquad \text{(B.179)}\\
= &-\frac{1}{2}\sum_{j=1}^{n}\left(-2\boldsymbol{z}_j^\top \mathbb{E}_{\boldsymbol{W},\boldsymbol{\mu},\tau}\left[\tau \boldsymbol{W}^\top\left(\widetilde{\boldsymbol{x}}_j - \boldsymbol{\mu}\right)\right]\right.\\
&\left.+\boldsymbol{z}_j^\top\left(\boldsymbol{I}_{q\times q} + \mathbb{E}_{\boldsymbol{W},\tau}\left[\tau \boldsymbol{W}^\top \boldsymbol{W}\right]\right)\boldsymbol{z}_j\right) + \text{const.} \qquad \text{(B.180)}
\end{aligned}$$

Note that the logarithm of the density $\mathcal{N}(\overline{\boldsymbol{z}}_j, \boldsymbol{\Sigma}_z)$ can be expressed as

$$-\frac{1}{2}\left[\boldsymbol{z}_j^\top \boldsymbol{\Sigma}_z^{-1}\boldsymbol{z}_j - 2\boldsymbol{z}_j^\top \boldsymbol{\Sigma}_z^{-1}\overline{\boldsymbol{z}}_j\right] + \text{const.} \qquad \text{(B.181)}$$

Comparing the terms for $\boldsymbol{z}_j^\top \boldsymbol{z}_j$ and $\boldsymbol{z}_j^\top$ in Equations (B.180) and (B.181) gives

$$\begin{aligned}
\boldsymbol{\Sigma}_z &= \left(\boldsymbol{I}_{q\times q} + \mathbb{E}_{\boldsymbol{W},\tau}\left[\tau \boldsymbol{W}^\top \boldsymbol{W}\right]\right)^{-1}, \qquad \text{(B.182)}\\
\overline{\boldsymbol{z}}_j &= \boldsymbol{\Sigma}_z\left(\mathbb{E}_{\boldsymbol{W},\tau}[\tau \boldsymbol{W}]^\top - \mathbb{E}_{\boldsymbol{W},\boldsymbol{\mu},\tau}\left[\tau \boldsymbol{W}^\top \boldsymbol{\mu}\right]\right). \qquad \text{(B.183)}
\end{aligned}$$

So we can see that

$$\mathrm{q}^*(\boldsymbol{Z}) = \prod_{j=1}^{n} \mathrm{q}^*(\boldsymbol{z}_j) = \prod_{j=1}^{n} \mathcal{N}(\overline{\boldsymbol{z}}_j, \boldsymbol{\Sigma}_z). \tag{B.184}$$

Form of $\mathbf{q}^*\left(\boldsymbol{x}_1^{(M_1)}, \ldots, \boldsymbol{x}_n^{(M_n)}\right)$

Only including the terms from Equation (B.96) that include $\mathcal{V}_1 = \left\{\boldsymbol{x}_1^{(M_1)}, \ldots, \boldsymbol{x}_n^{(M_n)}\right\}$, and dropping subscripts from the integration notation when the integrand does not depend upon on that particular variable, we have

$$\log \mathrm{q}^*\left(\boldsymbol{x}_1^{(M_1)}, \ldots, \boldsymbol{x}_n^{(M_n)}\right) = \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau}\left[\log \mathrm{p}\left(\boldsymbol{x}_j^{(M_j)} | \boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau\right)\right] + \text{const.} \tag{B.185}$$

Substituting in the expressions from Equation (B.102) and moving terms not dependent on $\boldsymbol{x}_j^{(M_j)}$ into the constant term, we get

$$\begin{aligned}
\log \mathrm{q}^*(\boldsymbol{x}_1^{(M_1)}, \ldots, \boldsymbol{x}_n^{(M_n)}) = \sum_{j=1}^{n} \mathbb{E}_{\boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau}\Big[& -\frac{\tau}{2}\left(\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{x}_j^{(M_j)} - 2\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{W} \boldsymbol{z}_j \right. \\
& \left. -2\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{\mu}^{(M_j)}\right)\Big] + \text{const} \qquad \text{(B.186)} \\
= & -\frac{1}{2} \sum_{j=1}^{n} \Big(\boldsymbol{x}_j^{(M_j)\top} \mathbb{E}_{\tau}[\tau] \boldsymbol{x}_j^{(M_j)} \\
& -2\boldsymbol{x}_j^{(M_j)\top} \mathbb{E}_{\boldsymbol{z}_j, \boldsymbol{W}, \boldsymbol{\mu}, \tau}\left[\tau\left(\boldsymbol{W}_j^{(M_j)} \boldsymbol{z}_j + \boldsymbol{\mu}^{(M_j)}\right)\right]\Big) + \text{const.} \qquad \text{(B.187)}
\end{aligned}$$

Note that the logarithm of the density $\mathcal{N}\left(\overline{\boldsymbol{x}}_j^{(M_j)}, \boldsymbol{\Sigma}_{\boldsymbol{x}_j^{(M_j)}}\right)$ can be expressed as

$$-\frac{1}{2}\left[\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{\Sigma}_{\boldsymbol{x}_j^{(M_j)}}^{-1} \boldsymbol{x}_j^{(M_j)} - 2\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{\Sigma}_{\boldsymbol{x}_j^{(M_j)}}^{-1} \overline{\boldsymbol{x}}_j^{(M_j)}\right] + \text{const.} \tag{B.188}$$

Comparing the terms for $\boldsymbol{x}_j^{(M_j)\top} \boldsymbol{x}_j^{(M_j)}$ and $\boldsymbol{x}_j^{(M_j)\top}$ in Equations (B.187) and (B.188) gives

$$\boldsymbol{\Sigma}_{\boldsymbol{x}_j^{(M_j)}} = \mathbb{E}_{\tau}[\tau]^{-1} \boldsymbol{I}_{(p-|O_j|)\times(p-|O_j|)} \tag{B.189}$$

$$\overline{\boldsymbol{x}}_j^{(M_j)} = \boldsymbol{\Sigma}_{\boldsymbol{x}_j^{(M_j)}}\left(\mathbb{E}_{\boldsymbol{W}, \tau, \boldsymbol{z}_j}\left[\tau \boldsymbol{W}^{(M_j)} \boldsymbol{z}_j\right] + \mathbb{E}_{\boldsymbol{\mu}, \tau}\left[\tau \boldsymbol{\mu}^{(M_j)}\right]\right). \tag{B.190}$$

So we can see that

$$\mathrm{q}^*\left(\boldsymbol{x}_1^{(M_1)}, \ldots, \boldsymbol{x}_n^{(M_n)}\right) = \prod_{j=1}^{n} \mathrm{q}^*\left(\boldsymbol{x}_j^{(M_j)}\right) = \prod_{j=1}^{n} \mathcal{N}\left(\overline{\boldsymbol{x}}_j^{(M_j)}, \boldsymbol{\Sigma}_{\boldsymbol{x}_j^{(M_j)}}\right). \tag{B.191}$$

B.4.5 Computation of moments

The non-trivial moments (i.e. those not immediate from their variational distributions) are as follows

$$\mathbb{E}_{z_j}\left[z_j z_j^\top\right] = \boldsymbol{\Sigma}_z + \overline{z}_j \overline{z}_j^\top, \tag{B.192}$$

$$\mathbb{E}_{\boldsymbol{x}_j^{(M_j)}}\left[\boldsymbol{x}_j^{(M_j)\top}\boldsymbol{x}_j^{(M_j)}\right] = \mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{x}_j^{(M_j)}}\right] + \overline{\boldsymbol{x}}_j^{(M_j)\top}\overline{\boldsymbol{x}}_j^{(M_j)}, \tag{B.193}$$

$$\mathbb{E}_{\boldsymbol{W},\tau}[\tau \boldsymbol{W}] = \int\int \tau \boldsymbol{W} \mathrm{q}^*(\boldsymbol{W}|\tau)\mathrm{q}^*(\tau)\mathrm{d}\boldsymbol{W}\boldsymbol{\tau} \tag{B.194}$$

$$= \int \tau \left[\int \boldsymbol{W} \mathrm{q}^*(\boldsymbol{W}|\tau)\mathrm{d}\boldsymbol{W}\right] \mathrm{q}^*(\tau)\mathrm{d}\tau \tag{B.195}$$

$$= \int \tau \boldsymbol{M}_{\widetilde{\boldsymbol{W}}}^\top \mathrm{q}^*(\tau)\mathrm{d}\tau \tag{B.196}$$

$$= \overline{\tau}\boldsymbol{M}_{\widetilde{\boldsymbol{W}}}^\top, \tag{B.197}$$

where $\boldsymbol{M}_{\widetilde{\boldsymbol{W}}} = (\boldsymbol{m}_{\widetilde{\boldsymbol{w}}_1}, \ldots, \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_p})$. In addition we have

$$\mathbb{E}_{\boldsymbol{W},\tau}\left[\tau \boldsymbol{W}^\top \boldsymbol{W}\right] = \int\int \tau \boldsymbol{W}^\top \boldsymbol{W} \mathrm{q}^*(\boldsymbol{W}|\tau)\mathrm{q}^*(\tau)\mathrm{d}\boldsymbol{W}\,\mathrm{d}\tau \tag{B.198}$$

$$= \int \tau \left[\int \boldsymbol{W}^\top \boldsymbol{W} \mathrm{q}^*(\boldsymbol{W}|\tau)\mathrm{d}\boldsymbol{W}\right] \mathrm{q}^*(\tau)\mathrm{d}\tau \tag{B.199}$$

$$= \int \tau \left[p(\tau\boldsymbol{\Lambda}_{\boldsymbol{w}})^{-1} + \boldsymbol{M}_{\widetilde{\boldsymbol{W}}}\boldsymbol{M}_{\widetilde{\boldsymbol{W}}}^\top\right] \mathrm{q}^*(\tau)\mathrm{d}\tau \tag{B.200}$$

$$= p\boldsymbol{\Lambda}_{\boldsymbol{w}}^{-1} + \overline{\tau}\boldsymbol{M}_{\widetilde{\boldsymbol{W}}}\boldsymbol{M}_{\widetilde{\boldsymbol{W}}}^\top, \tag{B.201}$$

$$\mathbb{E}_{\boldsymbol{w}_j,\tau}\left[\tau \boldsymbol{w}_j^\top \boldsymbol{w}_j\right] = p(\boldsymbol{\Lambda}_{\boldsymbol{w}}^{-1})_{jj} + \overline{\tau}\boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i}^\top \boldsymbol{m}_{\widetilde{\boldsymbol{w}}_i}, \tag{B.202}$$

$$\mathbb{E}_{\boldsymbol{\mu},\boldsymbol{W},\tau}\left[\tau \boldsymbol{W}^\top \boldsymbol{\mu}\right] = \int\int\int \tau \boldsymbol{W}^\top \boldsymbol{\mu} \mathrm{q}^*(\boldsymbol{\mu}|\boldsymbol{W},\tau)\mathrm{q}^*(\boldsymbol{W}|\tau)\mathrm{q}^*(\tau)\mathrm{d}\boldsymbol{\mu}\mathrm{d}\boldsymbol{W}\mathrm{d}\tau \tag{B.203}$$

$$= \int\int \tau \boldsymbol{W}^\top \left[\int \boldsymbol{\mu} \mathrm{q}^*(\boldsymbol{\mu}|\boldsymbol{W},\tau)\mathrm{d}\boldsymbol{\mu}\right] \mathrm{q}^*(\boldsymbol{W}|\tau)\mathrm{q}^*(\tau)\mathrm{d}\boldsymbol{W}\mathrm{d}\tau \tag{B.204}$$

$$= \int\int \tau \boldsymbol{W}^\top \left[\boldsymbol{W}\boldsymbol{s}_{\boldsymbol{\mu}} + \boldsymbol{m}_{\boldsymbol{\mu}}\right] \mathrm{q}^*(\boldsymbol{W}|\tau)\mathrm{q}^*(\tau)\mathrm{d}\boldsymbol{W}\mathrm{d}\tau \tag{B.205}$$

$$= \mathbb{E}_{\boldsymbol{W},\tau}\left[\tau \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{s}_{\boldsymbol{\mu}}\right] + \mathbb{E}_{\boldsymbol{W},\tau}\left[\tau \boldsymbol{W}\right]^\top \tag{B.206}$$

$$= p\boldsymbol{\Lambda}^{-1}\boldsymbol{s}_{\boldsymbol{\mu}} + \overline{\tau}\boldsymbol{M}_{\boldsymbol{W}}\boldsymbol{M}_{\boldsymbol{W}}^\top \boldsymbol{s}_{\boldsymbol{\mu}} + \overline{\tau}\boldsymbol{M}_{\boldsymbol{W}}\boldsymbol{m}_{\boldsymbol{\mu}} \tag{B.207}$$

$$= p\boldsymbol{\Lambda}^{-1}\boldsymbol{s}_{\boldsymbol{\mu}} + \overline{\tau}\boldsymbol{M}_{\boldsymbol{W}}\left(\boldsymbol{M}_{\boldsymbol{W}}^\top \boldsymbol{s}_{\boldsymbol{\mu}} + \boldsymbol{m}_{\boldsymbol{\mu}}\right), \tag{B.208}$$

$$\mathbb{E}_{\boldsymbol{W},\boldsymbol{z}_j,\tau}\left[\tau \boldsymbol{W}^{(M_j)}\boldsymbol{z}_j\right] = \int\int\int \tau \boldsymbol{W}^{(M_j)}\boldsymbol{z}_j \mathrm{q}^*(\boldsymbol{W}|\tau)\mathrm{q}^*(\tau)\mathrm{q}^*\left(\boldsymbol{z}_j\right)\mathrm{d}\boldsymbol{W}\mathrm{d}\tau\mathrm{d}\boldsymbol{z}_j \tag{B.209}$$

$$= \int\int \tau \boldsymbol{W}^{(M_j)} \mathrm{q}^*(\boldsymbol{W})\mathrm{q}^*(\tau)\mathrm{d}\boldsymbol{W}\mathrm{d}\tau \int \boldsymbol{z}_j \mathrm{q}^*\left(\boldsymbol{z}_j\right)\mathrm{d}\boldsymbol{z}_j \tag{B.210}$$

$$= \mathbb{E}_{\boldsymbol{W},\tau}\left[\tau \boldsymbol{W}^{(M_j)}\right] \mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right] \tag{B.211}$$

$$= \overline{\tau}\boldsymbol{M}_{\boldsymbol{W}^{(M_j)}}^\top \overline{\boldsymbol{z}}_j \tag{B.212}$$

$$\mathbb{E}_{\boldsymbol{\mu},\tau}\left[\tau \boldsymbol{\mu}^{(M_j)}\right] = \int\int \tau \boldsymbol{\mu}^{(M_j)} \mathrm{q}^*(\boldsymbol{\mu}|\boldsymbol{W},\tau)\mathrm{q}^*(\tau)\mathrm{d}\boldsymbol{\mu}\mathrm{d}\tau \tag{B.213}$$

$$= \int \tau \left[\int \boldsymbol{\mu}^{(M_j)} \mathrm{q}^*(\boldsymbol{\mu}|\boldsymbol{W},\tau)\mathrm{d}\boldsymbol{\mu}\right] \mathrm{q}^*(\tau)\mathrm{d}\tau \tag{B.214}$$

$$= \int \tau \left[\boldsymbol{W}^{(M_j)}\boldsymbol{s}_{\boldsymbol{\mu}} + \boldsymbol{m}_{\boldsymbol{\mu}}^{(M_j)}\right] \mathrm{q}^*(\tau)\mathrm{d}\tau \tag{B.215}$$

$$= \overline{\tau}\left(\boldsymbol{W}^{(M_j)}\boldsymbol{s}_{\boldsymbol{\mu}} + \boldsymbol{m}_{\boldsymbol{\mu}}^{(M_j)}\right). \tag{B.216}$$

The algorithm is then executed by cycling through the updates in Equations (B.139), (B.140), (B.141), (B.157), (B.158), (B.165), (B.166), (B.173), (B.174), (B.182), (B.183), (B.189), and (B.190) using the moments above.

B.5 Variational algorithm 2 - derivation

In this Section we derive the algorithm from Ilin and Raiko [60]. The paper provides the full model specification and variational updates, and so the contribution of this Section is the explicit presentation of the derivation.

B.5.1 Handling missing values

Missing values are dealt with using the same notation as Section B.4.1. In this algorithm, the missing values are not modelled probabilistically so that only the observed values are used in the computation of the variational updates.

B.5.2 Priors

The prior specification is as follows:

$$\mathrm{p}\left(\boldsymbol{z}_j\right) = \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{q\times q}), \tag{B.217}$$

$$\mathrm{p}\left(\boldsymbol{w}_g\right) = \mathcal{N}(\mathbf{0}, \nu_{w,g}\boldsymbol{I}_{p\times p}), \tag{B.218}$$

$$\mathrm{p}\left(\boldsymbol{\mu}\right) = \mathcal{N}(\mathbf{0}, \nu_{\mu}\boldsymbol{I}_{p\times p}) \tag{B.219}$$

$$\mathrm{p}\left(\boldsymbol{\epsilon}\right) = \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I}_{p\times p}). \tag{B.220}$$

In this algorithm, $(\sigma^2, \nu_{\mu}, \nu_{w,k})$ are not formally given priors. They are instead treated as hyperparameters whose values are set by analytically maximising the variational lower bound. Since the lower bound aims to approximate the marginal likelihood, this procedure approximates empirical Bayes estimation having assumed uniform priors on the parameters.

B.5.3 Joint distribution

Writing $\boldsymbol{X}$ for the $p\times n$ matrix whose columns are the $\boldsymbol{x}_j$, and $\boldsymbol{Z}$ for the $q\times n$ matrix whose columns are the $\boldsymbol{z}_j$, we can factorise the full joint model as follows:

$$\mathrm{p}(\boldsymbol{X},\boldsymbol{W},\boldsymbol{Z},\boldsymbol{\mu}) = \mathrm{p}(\boldsymbol{X}|\boldsymbol{W},\boldsymbol{Z},\boldsymbol{\mu})\mathrm{p}(\boldsymbol{W})\mathrm{p}(\boldsymbol{Z})\mathrm{p}(\boldsymbol{\mu}) \tag{B.221}$$

$$= \prod_{i=1}^{p}\prod_{j=1}^{n}\mathrm{p}(x_{ij}|\boldsymbol{w}_i,\boldsymbol{z}_j,\mu_i)\prod_{i=1}^{p}\mathrm{p}(\boldsymbol{w}_i)\prod_{j=1}^{n}\mathrm{p}(\boldsymbol{z}_j)\prod_{i=1}^{p}\mathrm{p}(\mu_i). \tag{B.222}$$

Using the notation for observed values introduced in Section 3.3.2, the joint model can be straightforwardly modified to allow for the possibility of missing values, as follows:

$$\mathrm{p}(\boldsymbol{X},\boldsymbol{W},\boldsymbol{Z},\boldsymbol{\mu}) = \prod_{ij\in O}\mathrm{p}(x_{ij}|\boldsymbol{w}_i,\boldsymbol{z}_j,\mu_i)\prod_{i=1}^{p}\mathrm{p}(\boldsymbol{w}_i)\prod_{j=1}^{n}\mathrm{p}(\boldsymbol{z}_j)\prod_{i=1}^{p}\mathrm{p}(\mu_i). \tag{B.223}$$

For the VB derivations, it is useful to write down the log densities for each expression in Equation (B.223). For the first term we have

$$\log \mathrm{p}(x_{ij}|\boldsymbol{w}_i, \boldsymbol{z}_j, \mu_i) = \log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{x_{ij}-(\boldsymbol{w}_i^\top \boldsymbol{z}_j+\mu_i)^2}{2\sigma^2}\right)\right) \tag{B.224}$$

$$= -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\left(x_{ij}-(\boldsymbol{w}_i^\top \boldsymbol{x}_j+\mu_i)\right)^2 \tag{B.225}$$

$$= -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\left(x_{ij}^2+(\boldsymbol{w}_i^\top \boldsymbol{z}_j)^2+\mu_i^2+2\mu_i\boldsymbol{w}_i^\top \boldsymbol{z}_j\right. \tag{B.226}$$
$$\left.-2\boldsymbol{w}_i^\top \boldsymbol{z}_j x_{ij} - 2\mu_i x_{ij}\right)$$

It is also useful to note that we can alternatively write $(x_{ij}-(\boldsymbol{w}_i^\top \boldsymbol{z}_j+\mu_i))^2$ as follows:

$$(x_{ij}-(\boldsymbol{w}_i^\top \boldsymbol{z}_j+\mu_i))^2 = (x_{ij}-(\boldsymbol{w}_i^\top \boldsymbol{z}_j+\mu_i))^\top (x_{ij}-(\boldsymbol{w}_i^\top \boldsymbol{z}_j+\mu_i)) \tag{B.227}$$

$$= x_{ij}^\top x_{ij} - \boldsymbol{x}_j^\top \boldsymbol{w}_i x_{ij} + \boldsymbol{z}_j^\top \boldsymbol{w}_i\mu_i - x_{ij}\boldsymbol{w}_i^\top \boldsymbol{z}_j + \mu_i^\top \boldsymbol{w}_i^\top \boldsymbol{z}_j \tag{B.228}$$
$$+ \boldsymbol{z}_j^\top(\boldsymbol{w}_i\boldsymbol{w}_i^\top)\boldsymbol{z}_j + \mu_i^\top\mu_i.$$

Substituting this into Equation (B.225) gives the alternative (but equivalent) expression

$$\log \mathrm{p}(x_{ij}|\boldsymbol{w}_i, \boldsymbol{z}_j, \mu_i) = -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\left(x_{ij}^\top x_{ij} - \boldsymbol{z}_j^\top \boldsymbol{w}_i x_{ij} + \boldsymbol{z}_j^\top \boldsymbol{w}_i\mu_i\right. \tag{B.229}$$
$$\left.+\mu_i^\top\mu_i - x_{ij}\boldsymbol{w}_i^\top \boldsymbol{z}_j + \mu_i^\top \boldsymbol{w}_i^\top \boldsymbol{z}_j + \boldsymbol{z}_j^\top(\boldsymbol{w}_i\boldsymbol{w}_i^\top)\boldsymbol{z}_j\right).$$

For $\log \mathrm{p}(\boldsymbol{w}_i)$ we have

$$\log \mathrm{p}(\boldsymbol{w}_i) = \log\left(\prod_{g=1}^{q}\frac{1}{\sqrt{2\pi\nu_{w,g}}}\exp -\frac{w_{ig}^2}{2\nu_{w,g}}\right) \tag{B.230}$$

$$= \sum_{g=1}^{q}\left(-\frac{1}{2}\log(2\pi\nu_{w,g}) - \frac{1}{2\nu_{w,g}}w_{ig}^2\right). \tag{B.231}$$

For $\log \mathrm{p}(\boldsymbol{z}_j)$ we have

$$\log \mathrm{p}(\boldsymbol{z}_j) = \log\left(\prod_{g=1}^{q}\frac{1}{\sqrt{2\pi}}\exp -\frac{z_{gj}^2}{2}\right) \tag{B.232}$$

$$= \sum_{g=1}^{q}\left(-\frac{1}{2}\log(2\pi) - \frac{1}{2}z_{gj}^2\right). \tag{B.233}$$

For $\log \mathrm{p}(\mu_i)$ we have

$$\log \mathrm{p}(\mu_i) = \log\left(\frac{1}{\sqrt{2\pi\nu_\mu}}\exp -\frac{\mu_i^2}{2\nu_\mu}\right) \tag{B.234}$$

$$= -\frac{1}{2}\log(2\pi\nu_\mu) - \frac{1}{2\nu_\mu}\mu_i^2. \tag{B.235}$$

B.5.4 Variational updates

Variational approximation

We seek to approximate the posterior distribution, $\mathrm{p}(\boldsymbol{W},\boldsymbol{Z},\boldsymbol{\mu}|\boldsymbol{X})$, by the mean field approximation

$$\mathrm{q}(\boldsymbol{W},\boldsymbol{Z},\boldsymbol{\mu}) = \mathrm{q}(\boldsymbol{W})\mathrm{q}(\boldsymbol{Z})\mathrm{q}(\boldsymbol{\mu}) \tag{B.236}$$

$$= \prod_{i=1}^{p}\mathrm{q}(\boldsymbol{w}_i)\prod_{j=1}^{n}\mathrm{q}(\boldsymbol{z}_j)\prod_{i=1}^{p}\mathrm{q}(\mu_i). \tag{B.237}$$

Variational lower bound

Using Equations (B.222) and (B.237), we can decompose the variational lower bound as follows:

$$\begin{aligned}\mathcal{L}(\mathrm{q}) = &\sum_{ij\in O}\mathbb{E}[\log \mathrm{p}(x_{ij}|\boldsymbol{w}_i,\boldsymbol{z}_j,\mu_i)] + \sum_{i=1}^{p}\mathbb{E}[\log \mathrm{p}(\boldsymbol{w}_i) - \log \mathrm{q}(\boldsymbol{w}_i)] \\ &+ \sum_{j=1}^{n}\mathbb{E}[\log \mathrm{p}(\boldsymbol{z}_j) - \log \mathrm{q}(\boldsymbol{z}_j)] + \sum_{i=1}^{p}\mathbb{E}[\log \mathrm{p}(\mu_i) - \log \mathrm{q}(\mu_i)],\end{aligned} \tag{B.238}$$

where the expectation is computed with respect to the joint variational density of all variables.

Optimal distributions q^*

According to Equation (B.237), we may subset the variables as $\mathcal{V}_1 = \{\boldsymbol{W}\}$, $\mathcal{V}_2 = \{\boldsymbol{Z}\}$, and $\mathcal{V}_3 = \{\boldsymbol{\mu}\}$. In order to obtain the form of the optimal distributions for each subset $\mathcal{V}_l$, $l = 1,2,3$ we use

$$\log \mathrm{q}^*(\mathcal{V}_l) = \sum_{ij\in O}\mathbb{E}_{\mathcal{V}_{\neg l}}\left[\log \mathrm{p}\left(x_{ij},\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{W}\right)\right] + \text{const}, \tag{B.239}$$

where $\neg l$ indicates all indices not equal to l. As before, terms from Equation (B.222) that do not depend on the variables in $\mathcal{V}_l$ can be absorbed into the constant term of

Equation (B.239). We will use this result when deriving the optimal distributions in each of the next sections.

Form of $\mathrm{q}^*(\mu_i)$

First, we isolate the terms in Equation (B.222) that involve μ_i, since all other terms are constant as a function of μ_i and hence will be absorbed into a constant term, and then consider the expectation of the log

$$\log \mathrm{q}^*(\mu_i) = \mathbb{E}_{\boldsymbol{w}_i, \boldsymbol{z}_j}\left[\log\left(\mathrm{p}(\mu_i)\prod_{j\in O_i}\mathrm{p}(x_{ij}|\boldsymbol{w}_i,\boldsymbol{z}_j,\mu_i)\right)\right] + \text{ const} \tag{B.240}$$

$$= \log \mathrm{p}(\mu_i) + \sum_{j\in O_i}\mathbb{E}_{\boldsymbol{w}_i,\boldsymbol{z}_j}\left[\log \mathrm{p}(x_{ij}|\boldsymbol{w}_i,\boldsymbol{z}_j,\mu_i)\right] + \text{ const} \tag{B.241}$$

$$= -\frac{1}{2\nu_\mu}\mu_i^2 + \sum_{j\in O_i}\mathbb{E}_{\boldsymbol{w}_i,\boldsymbol{z}_j}\left[-\frac{1}{2\sigma^2}\left(\mu_i^2 + 2\mu_i \boldsymbol{w}_i^\top \boldsymbol{z}_j - 2\mu_i x_{ij}\right)\right] + \text{ const} \tag{B.242}$$

$$= -\frac{1}{2\nu_\mu}\mu_i^2 - \frac{|O_i|}{2\sigma^2}\mu_i^2 - \frac{1}{2\sigma^2}2\mu_i\sum_{j\in O_i}\left(\mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]^\top \mathbb{E}_{\boldsymbol{z}_j}[\boldsymbol{z}_j] - x_{ij}\right) + \text{ const} \tag{B.243}$$

$$= -\frac{\sigma^2 + \nu_\mu|O_i|}{2\nu_\mu\sigma^2}\mu_i^2 + \frac{1}{\sigma^2}\mu_i\sum_{j\in O_i}\left(x_{ij} - \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]^\top \mathbb{E}_{\boldsymbol{z}_j}[\boldsymbol{z}_j]\right) + \text{ const.} \tag{B.244}$$

Comparing the above to the form of the log of a univariate normal density with mean $\overline{\mu}_i$ and variance $\widetilde{\mu}_i$

$$\log \mathcal{N}(\mu_i|\overline{\mu}_i,\widetilde{\mu}_i) = -\frac{1}{2\widetilde{\mu}_i}(\mu_i^2 - 2\mu_i\overline{\mu}_i) + \text{ constant,} \tag{B.245}$$

we deduce that $\mathrm{q}^*(\mu_i)$ is a univariate normal. Moreover, matching the coefficients of μ_i^2, we see that

$$\frac{1}{2\widetilde{\mu}_i} = \frac{\sigma^2 + \nu_\mu|O_i|}{2\nu_\mu\sigma^2}, \tag{B.246}$$

which, after rearranging, gives,

$$\widetilde{\mu}_i = \frac{\nu_\mu\sigma^2}{\sigma^2 + \nu_\mu|O_i|} \tag{B.247}$$

$$= \frac{\nu_\mu\sigma^2}{|O_i|(\nu_\mu + \sigma^2/|O_i|)}. \tag{B.248}$$

Similarly, matching the coefficients of μ_i gives

$$\frac{\overline{\mu}_i}{\widetilde{\mu}_i} = \frac{1}{\sigma^2} \sum_{j \in O_i} \left(x_{ij} - \overline{\boldsymbol{w}}_i^\top \overline{\boldsymbol{z}}_j \right). \tag{B.249}$$

Rearranging and substituting in our expression for $\widetilde{\mu}_i$, we obtain.

$$\overline{\mu}_i = \frac{\nu_\mu}{|O_i|(\nu_\mu + \sigma^2/|O_i|)} \sum_{j \in O_i} \left(x_{ij} - \mathbb{E}_{\boldsymbol{w}_i} [\boldsymbol{w}_i]^\top \mathbb{E}_{\boldsymbol{z}_j} \left[\boldsymbol{z}_j \right] \right). \tag{B.250}$$

Form of $\mathbf{q}^*(\boldsymbol{z}_j)$

We proceed similarly for $\boldsymbol{z}_j$:

$$\log \mathrm{q}^*(\boldsymbol{z}_j) = \mathbb{E}_{\boldsymbol{w}_i, \mu_i} \left[\log \left(\mathrm{p}(\boldsymbol{z}_j) \prod_{j \in O_i} \mathrm{p}(x_{ij} | \boldsymbol{w}_i, \boldsymbol{z}_j, \mu_i) \right) \right] + \text{ const} \tag{B.251}$$

$$= \log \mathrm{p}(\boldsymbol{z}_j) + \sum_{j \in O_i} \mathbb{E}_{\boldsymbol{w}_i, \mu_i} \left[\log \mathrm{p}(x_{ij} | \boldsymbol{w}_i, \boldsymbol{z}_j, \mu_i) \right] + \text{ const} \tag{B.252}$$

$$\begin{aligned} = -\frac{1}{2} \sum_{g=1}^{q} z_{gj}^2 - \frac{1}{2\sigma^2} \sum_{i \in O_j} \mathbb{E}_{\boldsymbol{w}_i, \mu_i} \Big[& -\boldsymbol{z}_j^\top \boldsymbol{w}_i x_{ij} + \boldsymbol{z}_j^\top \boldsymbol{w}_i \mu_i - x_{ij} \boldsymbol{w}_i^\top \boldsymbol{z}_j \\ & + \mu_i^\top \mu_i + \mu_i^\top \boldsymbol{w}_i^\top \boldsymbol{z}_j + \boldsymbol{z}_j^\top (\boldsymbol{w}_i \boldsymbol{w}_i^\top) \boldsymbol{z}_j \Big] + \text{ const} \end{aligned} \tag{B.253}$$

$$\begin{aligned} = -\frac{1}{2} \boldsymbol{z}_j^\top \boldsymbol{z}_j - \frac{1}{2\sigma^2} \sum_{i \in O_j} & \mathbb{E}_{\boldsymbol{w}_i} \left[\boldsymbol{z}_j^\top (\boldsymbol{w}_i \boldsymbol{w}_i^\top) \boldsymbol{z}_j \right] \\ & - \frac{1}{2\sigma^2} \sum_{i \in O_j} \mathbb{E}_{\boldsymbol{w}_i, \mu_i} \left[\mu_i^\top \boldsymbol{w}_i^\top - x_{ij} \boldsymbol{w}_i^\top \right] \boldsymbol{z}_j \\ & - \frac{1}{2\sigma^2} \sum_{i \in O_j} \boldsymbol{z}_j^\top \mathbb{E}_{\boldsymbol{w}_i, \mu_i} \left[\boldsymbol{w}_i \mu_i - \boldsymbol{w}_i x_{ij} \right] + \text{ const.} \end{aligned} \tag{B.254}$$

Note that $\mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i \boldsymbol{w}_i^\top] = \Sigma_{\boldsymbol{w}_i} + \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]\mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]^\top$, by the definition of covariance. Evaluating the expectations, we obtain

$$\begin{aligned}
\log \mathrm{q}^*(z_j) = &-\frac{1}{2} z_j^\top z_j - \frac{1}{2\sigma^2} z_j^\top \left(\sum_{i \in O_j} \left(\boldsymbol{\Sigma}_{\boldsymbol{w}_i} + \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]\mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]^\top \right) \right) z_j \\
&- \frac{1}{2\sigma^2} \left(\sum_{i \in O_j} \left(\mathbb{E}_{\mu_i}[\mu_i]^\top \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]^\top - x_{ij} \mathbb{E}[\boldsymbol{w}_i]^\top \right) \right) z_j \\
&- \frac{1}{2\sigma^2} z_j^\top \left(\sum_{i \in O_j} \left(\mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]\mathbb{E}_{\mu_i}[\mu_i] - \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i] x_{ij} \right) \right) + \text{ const} \qquad \text{(B.255)} \\
= &-\frac{1}{2\sigma^2} z_j^\top \left(\sigma^2 \boldsymbol{I}_{q \times q} + \sum_{i \in O_j} \left(\boldsymbol{\Sigma}_{\boldsymbol{w}_i} + \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]\mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]^\top \right) \right) z_j \\
&- \frac{1}{2\sigma^2} \left(\sum_{i \in O_j} \left(\mathbb{E}_{\mu_i}[\mu_i]^\top - x_{ij} \right) \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]^\top \right) z_j \\
&- \frac{1}{2\sigma^2} z_j^\top \left(\sum_{i \in O_j} \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i] \left(\mathbb{E}_{\mu_i}[\mu_i] - x_{ij} \right) \right) + \text{ const} \qquad \text{(B.256)}
\end{aligned}$$

Recall the form of the log of a multivariate normal density with mean $\overline{z}_j$ and covariance $\boldsymbol{\Sigma}_{z_j}$ as

$$\log \mathcal{N}(z_j | \overline{z}_j, \boldsymbol{\Sigma}_{z_j}) = -\frac{1}{2} \left(z_j^\top \boldsymbol{\Sigma}_{z_j}^{-1} z_j - \overline{z}_j^\top \boldsymbol{\Sigma}_{z_j}^{-1} z_j - z_j^\top \boldsymbol{\Sigma}_{z_j}^{-1} \overline{z}_j \right) + \text{ const.} \qquad \text{(B.257)}$$

Comparing Equations (B.256) and (B.257) and matching terms in $z_j^\top z_j$, we have

$$\begin{aligned}
z_j^\top \left(\sigma^2 \boldsymbol{I}_{q \times q} + \sum_{i \in O_j} \left(\boldsymbol{\Sigma}_{\boldsymbol{w}_i} + \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]\mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]^\top \right) \right) z_j &= \sigma^2 z_j^\top \boldsymbol{\Sigma}_{z_j}^{-1} z_j \qquad \text{(B.258)} \\
&= z_j^\top \left(\frac{1}{\sigma^2} \boldsymbol{\Sigma}_{z_j} \right)^{-1} z_j. \qquad \text{(B.259)}
\end{aligned}$$

Hence,

$$\left(\sigma^2 \boldsymbol{I}_{q \times q} + \sum_{i \in O_j} \left(\boldsymbol{\Sigma}_{\boldsymbol{w}_i} + \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]\mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]^\top \right) \right) = \left(\frac{1}{\sigma^2} \boldsymbol{\Sigma}_{z_j} \right)^{-1}. \qquad \text{(B.260)}$$

Inverting both sides, and multiplying through by σ^2, we obtain

$$\boldsymbol{\Sigma}_{z_j} = \sigma^2 \left(\sigma^2 \boldsymbol{I}_{q\times q} + \sum_{i \in O_j} \left(\boldsymbol{\Sigma}_{\boldsymbol{w}_i} + \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i] \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i]^\top \right) \right)^{-1}. \tag{B.261}$$

Comparing Equations (B.256) and (B.257) and matching terms in $\boldsymbol{z}_j^\top$, we have

$$\boldsymbol{\Sigma}_{z_j}^{-1} \overline{\boldsymbol{z}}_j = \frac{1}{\sigma^2} \left(\sum_{i \in O_j} \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i] \left(x_{ij} - \mathbb{E}_{\mu_i}[\mu_i] \right) \right), \tag{B.262}$$

which, after rearranging, gives

$$\overline{\boldsymbol{z}}_j = \frac{1}{\sigma^2} \boldsymbol{\Sigma}_{z_j} \left(\sum_{i \in O_j} \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i] \left(x_{ij} - \mathbb{E}_{\mu_i}[\mu_i] \right) \right). \tag{B.263}$$

Form of $\mathbf{q}^*(\boldsymbol{w}_i)$

Now we consider $\boldsymbol{w}_i$, in which we make use of the identity $\boldsymbol{x}_j^\top \boldsymbol{w}_i \boldsymbol{w}_i^\top \boldsymbol{x}_j = \boldsymbol{w}_i^\top \boldsymbol{x}_j \boldsymbol{x}_j^\top \boldsymbol{w}_i$. We have

$$\log \mathrm{q}^*(\boldsymbol{w}_i) = \mathbb{E}\left[\log\left(\mathrm{p}(\boldsymbol{w}_i)\prod_{j\in O_i}\mathrm{p}(x_{ij}|\boldsymbol{w}_i,\boldsymbol{z}_j,\mu_i)\right)\right] + \text{ const} \tag{B.264}$$

$$= \log \mathrm{p}(\boldsymbol{w}_i) + \sum_{j\in O_i}\mathbb{E}_{\boldsymbol{z}_j,\mu_i}\left[\log \mathrm{p}(x_{ij}|\boldsymbol{w}_i,\boldsymbol{z}_j,\mu_i)\right] + \text{ const} \tag{B.265}$$

$$= -\frac{1}{2}\sum_{g=1}^{q}\frac{w_{ig}^2}{\nu_{w,g}} + \sum_{j\in O_i}\mathbb{E}_{\boldsymbol{z}_j,\mu_i}\left[-\frac{1}{2\sigma^2}\left(-\boldsymbol{z}_j^\top \boldsymbol{w}_i x_{ij} + \boldsymbol{z}_j^\top \boldsymbol{w}_i \mu_i - x_{ij}\boldsymbol{w}_i^\top \boldsymbol{z}_j\right.\right. \tag{B.266}$$

$$\left.\left. + \mu_i^\top \boldsymbol{w}_i^\top \boldsymbol{z}_j + \boldsymbol{z}_j^\top\left(\boldsymbol{w}_i\boldsymbol{w}_i^\top\right)\boldsymbol{z}_j\right)\right] + \text{ const} \tag{B.267}$$

$$= -\frac{1}{2}\boldsymbol{w}_i^\top \mathrm{diag}\left(\nu_{w,g}^{-1}\right)\boldsymbol{w}_i - \frac{1}{2\sigma^2}\sum_{j\in O_i}\mathbb{E}_{\boldsymbol{z}_j,\mu_i}\left[\boldsymbol{z}_j^\top\left(\boldsymbol{w}_i\boldsymbol{w}_i^\top\right)\boldsymbol{z}_j\right] \tag{B.268}$$

$$-\frac{1}{2\sigma^2}\sum_{j\in O_i}\mathbb{E}_{\boldsymbol{z}_j,\mu_i}\left[\boldsymbol{z}_j^\top \boldsymbol{w}_i\mu_i - \boldsymbol{z}_j^\top \boldsymbol{w}_i x_{ij}\right] \tag{B.269}$$

$$-\frac{1}{2\sigma^2}\sum_{j\in O_i}\mathbb{E}_{\boldsymbol{z}_j,\mu_i}\left[\mu_i^\top \boldsymbol{w}_i^\top \boldsymbol{z}_j - x_{ij}\boldsymbol{w}_i^\top \boldsymbol{z}_j\right] + \text{ const} \tag{B.270}$$

$$= -\frac{1}{2}\boldsymbol{w}_i^\top \mathrm{diag}\left(\nu_{w,g}^{-1}\right)\boldsymbol{w}_i - \frac{1}{2\sigma^2}\sum_{j\in O_i}\mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{w}_i^\top\left(\boldsymbol{z}_j\boldsymbol{z}_j^\top\right)\boldsymbol{w}_i\right] \tag{B.271}$$

$$-\frac{1}{2\sigma^2}\left(\sum_{j\in O_i}\mathbb{E}_{\boldsymbol{z}_j,\mu_i}\left[\boldsymbol{z}_j^\top\mu_i - \boldsymbol{z}_j^\top x_{ij}\right]\right)\boldsymbol{w}_i$$

$$-\frac{1}{2\sigma^2}\boldsymbol{w}_i^\top\left(\sum_{j\in O_i}\mathbb{E}_{\boldsymbol{z}_j,\mu_i}\left[\mu_i^\top \boldsymbol{z}_j - x_{ij}\boldsymbol{z}_j\right]\right) + \text{ const} \tag{B.272}$$

$$= -\frac{1}{2\sigma^2}\boldsymbol{w}_i^\top\left(\sigma^2\mathrm{diag}\left(\nu_{w,g}^{-1}\right) + \sum_{j\in O_j}\left(\boldsymbol{\Sigma}_{\boldsymbol{z}_j} + \mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right]\mathbb{E}\left[\boldsymbol{z}_j\right]^\top\right)\right)\boldsymbol{w}_i$$

$$+\frac{1}{2\sigma^2}\left(\sum_{j\in O_i}\mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right]^\top\left(x_{ij} - \mathbb{E}_{\mu_i}\left[\mu_i\right]\right)\right)\boldsymbol{w}_i$$

$$+\frac{1}{2\sigma^2}\boldsymbol{w}_i^\top\left(\sum_{j\in O_i}\left(x_{ij} - \mathbb{E}_{\mu_i}\left[\mu_i\right]^\top\right)\mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right]\right) + \text{ const.} \tag{B.273}$$

Note again the form of the log of a multivariate normal density, but now with mean $\overline{\boldsymbol{w}}_i$ and covariance $\boldsymbol{\Sigma}_{\boldsymbol{w}_i}$

$$\log \mathcal{N}\left(\boldsymbol{w}_i|\overline{\boldsymbol{w}}_i,\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right) = -\frac{1}{2}\left(\boldsymbol{w}_i^\top\boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}\boldsymbol{w}_i - \overline{\boldsymbol{w}}_i^\top\boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}\boldsymbol{w}_i - \boldsymbol{w}_i^\top\boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}\overline{\boldsymbol{w}}_i\right) + \text{ constant.} \tag{B.274}$$

Comparing the $\boldsymbol{w}_i^\top \boldsymbol{w}_i$ terms in Equations (B.273) and (B.274), we have

$$-\frac{1}{2}\boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1} = -\frac{1}{2\sigma^2}\left(\sigma^2 \text{diag}\left(\nu_{w,g}^{-1}\right) + \sum_{j\in O_j}\left(\boldsymbol{\Sigma}_{\boldsymbol{z}_j} + \mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right]\mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right]^\top\right)\right) \tag{B.275}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{w}_i} = \sigma^2\left(\sigma^2 \text{diag}\left(\nu_{w,g}^{-1}\right) + \sum_{j\in O_j}\left(\boldsymbol{\Sigma}_{\boldsymbol{z}_j} + \mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right]\mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right]^\top\right)\right)^{-1}. \tag{B.276}$$

Now comparing the $\boldsymbol{w}_i^\top$ coefficients from Equations (B.273) and (B.274) we get

$$\frac{1}{2}\boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}\overline{\boldsymbol{w}}_i = \frac{1}{2\sigma^2}\left(\sum_{j\in O_i}\left(x_{ij} - \overline{\mu}_i^\top\right)\overline{\boldsymbol{z}}_j\right) \tag{B.277}$$

$$\overline{\boldsymbol{w}}_i = \frac{1}{\sigma^2}\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\left(\sum_{j\in O_i}\left(x_{ij} - \overline{\mu}_i^\top\right)\overline{\boldsymbol{z}}_j\right). \tag{B.278}$$

It is now clear that the expectations in each of the q* distributions can be directly computed as $\mathbb{E}_{\mu_i}\left[\mu_i\right] = \overline{\mu}_i$, $\mathbb{E}_{\boldsymbol{z}_j}\left[\boldsymbol{z}_j\right] = \overline{\boldsymbol{z}}_j$, and $\mathbb{E}_{\boldsymbol{w}_i}\left[\boldsymbol{w}_i\right] = \overline{\boldsymbol{w}}_i$.

Hyperparameter updates

For convenience, can decompose the variational lower bound from Equation (B.238) as follows:

$$-\mathcal{L}(\text{q}) = \sum_{ij\in O} C_{x_{ij}} + \sum_{i=1}^{p} C_{\boldsymbol{w}_i} + \sum_{j=1}^{n} C_{\boldsymbol{z}_j} + \sum_{i=1}^{p} C_{\mu_i}, \tag{B.279}$$

where

$$C_{x_{ij}} = \mathbb{E}[-\log \text{p}(x_{ij}|\boldsymbol{w}_i, \boldsymbol{z}_j, \mu_i)] \tag{B.280}$$

$$C_{\boldsymbol{w}_i} = \mathbb{E}[\log \text{q}(\boldsymbol{w}_i) - \log \text{p}(\boldsymbol{w}_i)] \tag{B.281}$$

$$C_{\boldsymbol{z}_j} = \mathbb{E}[\log \text{q}(\boldsymbol{z}_j) - \log \text{p}(\boldsymbol{z}_j)] \tag{B.282}$$

$$C_{\mu_i} = \mathbb{E}[\log \text{q}(\mu_i) - \log \text{p}(\mu_i)] \tag{B.283}$$

and the expectations are taken with respect to the variational density of all the random variables (although in all but one term this can be reduced to just one variable due to the independence assumption). We now derive expressions for each of these terms. For

C_{μ_i} we note the following:

$$\log \mathrm{q}(\mu_i) - \log \mathrm{p}(\mu_i) = -\frac{1}{2}\log(2\pi\widetilde{\mu}_i) - \frac{1}{2\widetilde{\mu}_i}(\mu_i - \overline{\mu}_i)^2 + \frac{1}{2}\log(2\pi\nu_\mu) + \frac{1}{2\nu_\mu}\mu_i^2 \quad \text{(B.284)}$$

$$= -\frac{1}{2}\log\widetilde{\mu}_i + \frac{1}{2}\log\nu_\mu - \frac{1}{2\widetilde{\mu}_i}(\mu_i^2 + \overline{\mu}_i^2 - 2\mu_i\overline{\mu}_i) + \frac{1}{2\nu_\mu}\mu_i^2. \quad \text{(B.285)}$$

Evaluating the expectation, and noting that $\mathbb{E}_{\mu_i}\left[\mu_i^2\right] = \widetilde{\mu}_i + \overline{\mu}_i^2$, we obtain

$$\mathbb{E}_{\mu_i}[\log \mathrm{q}(\mu_i) - \log \mathrm{p}(\mu_i)] = -\frac{1}{2}\log\frac{\widetilde{\mu}_i}{\nu_\mu} - \frac{1}{2\widetilde{\mu}_i}(\widetilde{\mu}_i + \overline{\mu}_i^2 + \overline{\mu}_i^2 - 2\overline{\mu}_i^2) + \frac{1}{2\nu_\mu}(\widetilde{\mu}_i + \overline{\mu}_i^2) \quad \text{(B.286)}$$

$$= -\frac{1}{2}\log\frac{\widetilde{\mu}_i}{\nu_\mu} - \frac{1}{2} + \frac{1}{2\nu_\mu}(\widetilde{\mu}_i + \overline{\mu}_i^2). \quad \text{(B.287)}$$

For C_{z_j} we note the following:

$$\log \mathrm{q}(z_j) - \log \mathrm{p}(z_j) = -\frac{1}{2}\left(q\log 2\pi + \log|\boldsymbol{\Sigma}_{z_j}|\right) - \frac{1}{2}(z_j - \overline{z}_j)^\top \boldsymbol{\Sigma}_{z_j}^{-1}(z_j - \overline{z}_j) - \left(-\frac{q}{2}\log(2\pi) - \frac{1}{2}z_j^\top z_j\right) \quad \text{(B.288)}$$

$$= -\frac{1}{2}\log|\boldsymbol{\Sigma}_{z_j}| - \frac{1}{2}\left(z_j^\top \boldsymbol{\Sigma}_{z_j}^{-1} z_j - \overline{z}_j^\top \boldsymbol{\Sigma}_{z_j}^{-1} z_j - z_j^\top \boldsymbol{\Sigma}_{z_j}^{-1}\overline{z}_j + \overline{z}_j^\top \boldsymbol{\Sigma}_{z_j}^{-1}\overline{z}_j - z_j^\top z_j\right) \quad \text{(B.289)}$$

To evaluate the expectation, it is useful to first note the following identity for a symmetric matrix $\boldsymbol{A}$ and random variable $\boldsymbol{y}$ with mean $\overline{\boldsymbol{y}}$ and variance $\boldsymbol{\Sigma}_{\boldsymbol{y}}$

$$\mathbb{E}_{\boldsymbol{y}}[\boldsymbol{y}^\top \boldsymbol{A}\boldsymbol{y}] = \mathrm{Tr}(\boldsymbol{A}\boldsymbol{\Sigma}_{\boldsymbol{y}}) + \overline{\boldsymbol{y}}^\top \boldsymbol{A}\overline{\boldsymbol{y}}, \quad \text{(B.290)}$$

where $\mathrm{Tr}(\boldsymbol{A})$ denotes the trace of matrix $\boldsymbol{A}$. Thus, $\mathbb{E}[z_j^\top \boldsymbol{\Sigma}_{z_j}^{-1} z_j] = \mathrm{Tr}(\boldsymbol{\Sigma}_{z_j}^{-1}\boldsymbol{\Sigma}_{z_j}) + \overline{z}_j^\top \boldsymbol{\Sigma}_{z_j}^{-1}\overline{z}_j = q + \overline{z}_j^\top \boldsymbol{\Sigma}_{z_j}^{-1}\overline{z}_j$. Moreover, $\mathbb{E}_{z_j}[z_j^\top z_j] = \mathrm{Tr}(\boldsymbol{\Sigma}_{z_j}) + \overline{z}_j^\top \overline{z}_j$. Using this, we obtain

$$\mathbb{E}_{z_j}[\log \mathrm{q}(z_j) - \log \mathrm{p}(z_j)] = -\frac{1}{2}\log|\boldsymbol{\Sigma}_{z_j}| - \frac{1}{2}\left(q - \mathrm{Tr}(\boldsymbol{\Sigma}_{z_j}) - \overline{z}_j^\top \overline{z}_j)\right). \quad \text{(B.291)}$$

For $C_{x_{ij}}$ we note that:

$$\log \mathrm{p}(x_{ij}|\boldsymbol{w}_i, z_j, \mu_i) = -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\left(x_{ij}^\top x_{ij} - z_j^\top \boldsymbol{w}_i x_{ij} + z_j^\top \boldsymbol{w}_i\mu_i + \mu_i^\top \mu_i - x_{ij}\boldsymbol{w}_i^\top z_j + \mu_i^\top \boldsymbol{w}_i^\top z_j + z_j^\top(\boldsymbol{w}_i\boldsymbol{w}_i^\top)z_j\right). \quad \text{(B.292)}$$

Hence, taking the expectation, we have

$$\begin{aligned}\mathbb{E}_{\boldsymbol{w}_i,\boldsymbol{z}_j,\mu_i}[\log \mathrm{p}(x_{ij}|\boldsymbol{w}_i,\boldsymbol{z}_j,\mu_i)] = &-\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\Big(x_{ij}^\top x_{ij} - \overline{\boldsymbol{z}}_j^\top \overline{\boldsymbol{w}}_i x_{ij} \\ &+ \overline{\boldsymbol{z}}_j^\top \overline{\boldsymbol{w}}_i \overline{\mu}_i + \mathbb{E}_{\mu_i}\left[\mu_i^\top \mu_i\right] - x_{ij}\overline{\boldsymbol{w}}_i^\top \overline{\boldsymbol{z}}_j \\ &+ \overline{\mu}_i^\top \overline{\boldsymbol{w}}_i^\top \overline{\boldsymbol{z}}_j + \mathbb{E}_{\boldsymbol{w}_i,\boldsymbol{z}_j}[\boldsymbol{z}_j^\top(\boldsymbol{w}_i\boldsymbol{w}_i^\top)\boldsymbol{z}_j]\Big) \qquad \text{(B.293)} \\ = &-\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\Big(\left(x_{ij} - (\overline{\boldsymbol{w}}_i^\top \overline{\boldsymbol{z}}_j + \overline{\mu}_i)\right)^2 \\ &- \overline{\mu}_i^\top \overline{\mu}_i - \overline{\boldsymbol{z}}_j^\top(\overline{\boldsymbol{w}}_i\overline{\boldsymbol{w}}_i^\top)\overline{\boldsymbol{z}}_j + \mathbb{E}_{\mu_i}[\mu_i^\top \mu_i] \\ &+ \mathbb{E}_{\boldsymbol{w}_i,\boldsymbol{z}_j}[\boldsymbol{z}_j^\top(\boldsymbol{w}_i\boldsymbol{w}_i^\top)\boldsymbol{z}_j]\Big). \qquad \text{(B.294)}\end{aligned}$$

It remains to evaluate the final two expectations in the above (and then to simplify). We have

$$\mathbb{E}_{\mu_i}[\mu_i^\top \mu_i] = \widetilde{\mu}_i + \overline{\mu}_i\overline{\mu}_i^\top, \qquad \text{(B.295)}$$

by the definition of variance. To evaluate the final expectation, we first evaluate the expectation with respect to $\boldsymbol{w}_i$, and then with respect to $\boldsymbol{z}_j$, so that we have

$$\begin{aligned}\mathbb{E}_{\boldsymbol{w}_i,\boldsymbol{z}_j}[\boldsymbol{z}_j^\top(\boldsymbol{w}_i\boldsymbol{w}_i^\top)\boldsymbol{z}_j] =&\mathbb{E}_{\boldsymbol{z}_j}[\boldsymbol{z}_j^\top \mathbb{E}_{\boldsymbol{w}_i}[\boldsymbol{w}_i\boldsymbol{w}_i^\top]\boldsymbol{z}_j] &\text{(B.296)}\\ =&\mathbb{E}_{\boldsymbol{z}_j}[\boldsymbol{z}_j^\top(\boldsymbol{\Sigma}_{\boldsymbol{w}_i} + \overline{\boldsymbol{w}}_i\overline{\boldsymbol{w}}_i^\top)\boldsymbol{z}_j] &\text{(B.297)}\\ =&\mathbb{E}_{\boldsymbol{z}_j}[\boldsymbol{z}_j^\top\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\boldsymbol{z}_j] + \mathbb{E}_{\boldsymbol{z}_j}[\boldsymbol{z}_j^\top(\overline{\boldsymbol{w}}_i\overline{\boldsymbol{w}}_i^\top)\boldsymbol{z}_j] &\text{(B.298)}\\ =&\mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\right] + \overline{\boldsymbol{z}}_j^\top\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\overline{\boldsymbol{z}}_j + \mathrm{Tr}\left[(\overline{\boldsymbol{w}}_i\overline{\boldsymbol{w}}_i^\top)\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\right] \\ &+ \overline{\boldsymbol{z}}_j^\top(\overline{\boldsymbol{w}}_i\overline{\boldsymbol{w}}_i^\top)\overline{\boldsymbol{z}}_j. &\text{(B.299)}\end{aligned}$$

Note also that, using the property that the trace of a product is invariant under cyclic permutations, we have

$$\mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\right] = \mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right] \qquad \text{(B.300)}$$

and also

$$\begin{aligned}\mathrm{Tr}\left[\overline{\boldsymbol{w}}_i\overline{\boldsymbol{w}}_i^\top\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\right] &= \mathrm{Tr}\left[\overline{\boldsymbol{w}}_i^\top\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\overline{\boldsymbol{w}}_i\right] &\text{(B.301)}\\ &= \overline{\boldsymbol{w}}_i^\top\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\overline{\boldsymbol{w}}_i, &\text{(B.302)}\end{aligned}$$

with the final equality following from the fact that $\overline{\boldsymbol{w}}_i^\top\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\overline{\boldsymbol{w}}_i$ is a scalar, and hence equal to its trace. Thus,

$$\mathbb{E}_{\boldsymbol{w}_i,\boldsymbol{z}_j}\left[\boldsymbol{z}_j^\top\left(\boldsymbol{w}_i\boldsymbol{w}_i^\top\right)\boldsymbol{z}_j\right] = \mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right] + \overline{\boldsymbol{z}}_j^\top\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\overline{\boldsymbol{z}}_j + \overline{\boldsymbol{w}}_i^\top\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\overline{\boldsymbol{w}}_i + \overline{\boldsymbol{z}}_j^\top\left(\overline{\boldsymbol{w}}_i\overline{\boldsymbol{w}}_i^\top\right)\overline{\boldsymbol{z}}_j. \qquad \text{(B.303)}$$

Substituting Equations (B.295) and (B.303) into Equation (B.294), we obtain:

$$\mathbb{E}_{\boldsymbol{w}_i,z_j,\mu_i}\left[\log \mathrm{p}\left(x_{ij}|\boldsymbol{w}_i,z_j,\mu_i\right)\right] = -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\left(\left(x_{ij} - (\overline{\boldsymbol{w}}_i^\top \overline{z}_j + \overline{\mu}_i)\right)^2 + \widetilde{\mu}_i + \mathrm{Tr}\left[\boldsymbol{\Sigma}_{z_j}\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right] + \overline{z}_j^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i}\overline{z}_j + \overline{\boldsymbol{w}}_i^\top \boldsymbol{\Sigma}_{z_j}\overline{\boldsymbol{w}}_i\right). \quad \text{(B.304)}$$

Thus,

$$C_{x_{ij}} = \frac{1}{2}\log(2\pi\sigma^2) + \frac{1}{2\sigma^2}\left(\left(x_{ij} - (\overline{\boldsymbol{w}}_i^\top \overline{z}_j + \overline{\mu}_i)\right)^2 + \widetilde{\mu}_i + \mathrm{Tr}\left[\boldsymbol{\Sigma}_{z_j}\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right] + \overline{z}_j^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i}\overline{z}_j + \overline{\boldsymbol{w}}_i^\top \boldsymbol{\Sigma}_{z_j}\overline{\boldsymbol{w}}_i\right) \quad \text{(B.305)}$$

For $C_{\boldsymbol{w}_i}$ we have

$$\begin{aligned}\log \mathrm{q}(\boldsymbol{w}_i) - \log \mathrm{p}(\boldsymbol{w}_i) &= -\frac{1}{2}\left(q\log(2\pi) + \log|\boldsymbol{\Sigma}_{\boldsymbol{w}_i}|\right) \\ &\quad - \frac{1}{2}(\boldsymbol{w}_i - \overline{\boldsymbol{w}}_i)^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}(\boldsymbol{w}_i - \overline{\boldsymbol{w}}_i) \\ &\quad + \frac{1}{2}q\log(2\pi) + \frac{1}{2}\log \nu_{w,g} + \frac{1}{2\nu_{w,g}}\boldsymbol{w}_i^\top \boldsymbol{w}_i \quad \text{(B.306)} \\ &= \frac{1}{2}\log \nu_{w,g} - \frac{1}{2}\log|\boldsymbol{\Sigma}_{\boldsymbol{w}_i}| \\ &\quad - \frac{1}{2}(\boldsymbol{w}_i - \overline{\boldsymbol{w}}_i)^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}(\boldsymbol{w}_i - \overline{\boldsymbol{w}}_i) \\ &\quad + \frac{1}{2\nu_{w,g}}\boldsymbol{w}_i^\top \boldsymbol{w}_i \quad \text{(B.307)} \\ &= -\frac{1}{2}\log\frac{|\boldsymbol{\Sigma}_{\boldsymbol{w}_i}|}{\nu_{w,g}} - \frac{1}{2}\left(\boldsymbol{w}_i^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}\boldsymbol{w}_i - \boldsymbol{w}_i^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}\overline{\boldsymbol{w}}_i\right. \\ &\quad \left. - \overline{\boldsymbol{w}}_i^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}\boldsymbol{w}_i + \overline{\boldsymbol{w}}_i^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}\overline{\boldsymbol{w}}_i - \frac{1}{\nu_{w,g}}\boldsymbol{w}_i^\top \boldsymbol{w}_i\right). \quad \text{(B.308)}\end{aligned}$$

Taking the expectation of this term gives

$$\begin{aligned}\mathbb{E}_{\boldsymbol{w}_i}[\log \mathrm{q}(\boldsymbol{w}_i) - \log \mathrm{p}(\boldsymbol{w}_i)] &= -\frac{1}{2}\log\frac{|\boldsymbol{\Sigma}_{\boldsymbol{w}_i}|}{\nu_{w,g}} - \frac{1}{2}\left(q + \overline{\boldsymbol{w}}_i^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}\overline{\boldsymbol{w}}_i - \overline{\boldsymbol{w}}_i^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}\overline{\boldsymbol{w}}_i\right. \\ &\quad - \overline{\boldsymbol{w}}_i^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}\overline{\boldsymbol{w}}_i + \overline{\boldsymbol{w}}_i^\top \boldsymbol{\Sigma}_{\boldsymbol{w}_i}^{-1}\overline{\boldsymbol{w}}_i \\ &\quad \left. - \frac{1}{\nu_{w,g}}\left(\mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right] + \overline{\boldsymbol{w}}_i^\top \overline{\boldsymbol{w}}_i\right)\right) \quad \text{(B.309)} \\ &= -\frac{1}{2}\log\frac{|\boldsymbol{\Sigma}_{\boldsymbol{w}_i}|}{\nu_{w,g}} - \frac{1}{2}\left(q - \frac{1}{\nu_{w,g}}\mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right]\right. \\ &\quad \left. - \frac{1}{\nu_{w,g}}\overline{\boldsymbol{w}}_i^\top \overline{\boldsymbol{w}}_i\right). \quad \text{(B.310)}\end{aligned}$$

The updates for the variance parameters $(\sigma^2, \nu_\mu, \nu_{w,g})$ are derived by finding where the variational cost function has a minimum (the maximum of the variational lower bound).

Update for ν_μ

The only term in the variational cost function that includes ν_μ is the sum $\sum_{i=1}^{p} C_{\mu_i}$, so

$$\frac{\partial}{\partial \nu_\mu}(-\mathcal{L}(\mathrm{q})) = \frac{\partial}{\partial \nu_\mu}\left(\sum_{i=1}^{p} C_{\mu_i}\right) \tag{B.311}$$

$$= \sum_{i=1}^{p} \frac{\partial}{\partial \nu_\mu} C_{\mu_i} \tag{B.312}$$

$$= \sum_{i=1}^{p} \frac{\partial}{\partial \nu_\mu}\left(-\frac{1}{2}\log\frac{\widetilde{\mu}_i}{\nu_\mu} - \frac{1}{2} + \frac{1}{2\nu_\mu}(\widetilde{\mu}_i + \overline{\mu}_i^2)\right) \tag{B.313}$$

$$= \sum_{i=1}^{p} \frac{\partial}{\partial \nu_\mu}\left(\frac{1}{2}\log\nu_\mu + \frac{1}{\nu_\mu}\frac{1}{2}(\widetilde{\mu}_i + \overline{\mu}_i^2)\right) \tag{B.314}$$

$$= \sum_{i=1}^{p}\left(\frac{1}{2}\frac{1}{\nu_\mu} + \frac{-1}{\nu_\mu^2}\frac{1}{2}(\widetilde{\mu}_i + \overline{\mu}_i^2)\right). \tag{B.315}$$

Setting equal to zero, and multiplying through by $2\nu_\mu^2$, we obtain

$$0 = \sum_{i=1}^{p}\left(\nu_\mu - (\widetilde{\mu}_i + \overline{\mu}_i^2)\right) \tag{B.316}$$

$$= p\nu_\mu - \sum_{i=1}^{p}(\widetilde{\mu}_i + \overline{\mu}_i^2). \tag{B.317}$$

Hence,

$$\nu_\mu = \frac{1}{p}\sum_{i=1}^{p} \widetilde{\mu}_i + \overline{\mu}_i^2. \tag{B.318}$$

Update for σ^2

The only term in the variational cost function that includes σ^2 is the sum $\sum_{ij\in O} C_{x_{ij}}$, so

$$\frac{\partial}{\partial\sigma^2}(-\mathcal{L}(\mathrm{q})) = \frac{\partial}{\partial\sigma^2}\left(\sum_{ij\in O} C_{x_{ij}}\right) \tag{B.319}$$

$$= \sum_{ij\in O}\frac{\partial}{\partial\sigma^2}\left(\frac{1}{2}\log 2\pi\sigma^2 + \frac{1}{2\sigma^2}\left[\left(x_{ij} - \left(\overline{\boldsymbol{w}}_i^\top\overline{\boldsymbol{z}}_j + \overline{\mu}_i\right)\right)^2 + \widetilde{\mu}_i\right.\right. \tag{B.320}$$

$$\left.\left. + \mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right] + \overline{\boldsymbol{z}}_j^\top\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\boldsymbol{z}_j + \overline{\boldsymbol{w}}_i^\top\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\boldsymbol{w}_i\right]\right) \tag{B.321}$$

$$= \sum_{ij\in O}\left(\frac{1}{2\sigma^2} - \frac{1}{2\sigma^4}\left[\left(x_{ij} - \left(\overline{\boldsymbol{w}}_i^\top\overline{\boldsymbol{z}}_j + \overline{\mu}_i\right)\right)^2 + \widetilde{\mu}_i + \mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right]\right.\right. \tag{B.322}$$

$$\left.\left. + \overline{\boldsymbol{z}}_j^\top\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\boldsymbol{z}_j + \overline{\boldsymbol{w}}_i^\top\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\boldsymbol{w}_i\right]\right) \tag{B.323}$$

Setting this equal to zero, multiplying through by $2\sigma^4$, and rearranging gives

$$\sigma^2 = \frac{1}{np}\sum_{ij\in O}\left(x_{ij} - \overline{\boldsymbol{w}}_i^\top\overline{\boldsymbol{z}}_j - \overline{\mu}_i\right)^2 + \widetilde{\mu}_i + \overline{\boldsymbol{z}}_j^\top\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\boldsymbol{z}_j + \overline{\boldsymbol{w}}_i^\top\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\boldsymbol{w}_i + \mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{z}_j}\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right]. \tag{B.324}$$

Update for $\nu_{w,g}$

The only term in the variational cost function that includes $\nu_{w,g}$ is $\sum_{i=1}^{p} C_{\boldsymbol{w}_i}$, so

$$\frac{\partial}{\partial\nu_{w,g}}(-\mathcal{L}(\mathrm{q})) = \frac{\partial}{\partial\nu_{w,g}}\left(\sum_{ij\in O}^{p} C_{\boldsymbol{w}_i}\right) \tag{B.325}$$

$$= \sum_{i=1}^{p}\frac{\partial}{\partial\nu_{w,g}}\left(-\frac{1}{2}\log\frac{|\boldsymbol{\Sigma}_{\boldsymbol{w}_i}|}{\nu_{w,g}} + \frac{1}{2\nu_{w,g}}\left(\mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right] + \overline{\boldsymbol{w}}_i^\top\overline{\boldsymbol{w}}_i\right)\right) \tag{B.326}$$

$$= \sum_{i=1}^{p}\left(\frac{1}{2\nu_{w,g}} - \frac{1}{2\nu_{w,g}^2}\left(\mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right] + \overline{\boldsymbol{w}}_i^\top\overline{\boldsymbol{w}}_i\right)\right). \tag{B.327}$$

Equating this equal to zero and multiplying through by $2\nu_{w,g}^2$ gives:

$$0 = \sum_{i=1}^{p}\left(\nu_{w,g} - \mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right] - \overline{\boldsymbol{w}}_i^\top\overline{\boldsymbol{w}}_i\right) \tag{B.328}$$

$$\nu_{w,g} = \frac{1}{p}\sum_{i=1}^{p}\mathrm{Tr}\left[\boldsymbol{\Sigma}_{\boldsymbol{w}_i}\right] + \overline{\boldsymbol{w}}_i^\top\overline{\boldsymbol{w}}_i. \tag{B.329}$$

The algorithm is then executed by cycling through the update Equations (B.248), (B.250), (B.261), (B.263), (B.276), (B.278), (B.318), (B.324), and (B.329) until convergence.

B.6 pcaNet vs. pcaMethods timing simulation

In this section we compare the timing of algorithms EM 1 from pcaMethods [98] against their implementation in pcaNet. We present a small simulation study to demonstrate this.

We generate $M = 100$ datasets of size $n \in \{p/5, p/2, p\}$ from $\boldsymbol{X} = \boldsymbol{W}\boldsymbol{Z} + \boldsymbol{\mu} + \boldsymbol{\epsilon}$ using $p \in \{100, 250, 500\}$, $q = 3$, $\boldsymbol{\mu} = \boldsymbol{0}$, and $\log \sigma^2 \sim \mathcal{N}(0,1)$. Recall that by construction, this means that $\boldsymbol{z}_j \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{3\times 3})$ for $j = 1,\ldots,n$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma \boldsymbol{I}_{3\times 3})$.

To generate $\boldsymbol{W}$, we first simulate a binary matrix of dimension $(p \times q)$ whose entries are generated using $\mathcal{B}(0.5)$, the Bernoulli distribution with probability of success equal to 0.5. We use this binary matrix to create $\boldsymbol{W}$ by simulating $W_{ik} \sim \mathcal{N}(0,1)$ for the (i,k)-th binary entry equal to 1 $i = 1,\ldots,p$, $k = 1,\ldots,q$, and $W_{ik} = 0$ when the Bernoulli trial resulted in 0. There is one binary matrix for each specification of p and n, from which the M loading matrices $\boldsymbol{W}$ are generated.

We measure the time taken for each function to run, replicating the actual time that a user would experience when running the code. Since pcaNet only changes the implementation of the iterative updates (using Rcpp instead) for algorithm EM 1, the timing comparison is not biased by 'housekeeping' code such as argument checking or other auxilliary functionality. We report the log-fold change in time taken for each algorithm with the pcaNet version as the numerator. This is a relative measure of timing such that positive values indicate an improvement of pcaNet over the pcaMethods implementation, negative values vice versa, and values of 0 indicating no change. The log scale was chosen as some changes were very high/low in real-time. We use box plots to show the spread across the M datasets for each data dimension. All parameter estimates do not practically differ (within $\exp(-15)$) and so are not presented.

The results for EM algorithm 1 are shown in Figure B.1. We can see that the lower quartile change for all values of n and p are above 0, showing the frequent decrease in time taken by the pcaNet implementation relative to that of pcaMethods. Recall that values of 0.5 and 1 on this log scale refer to improvements of approximately 1.6 and 2.7 times (respectively) in real-time. Few values lie in the negative range of the log-fold change, with almost all of them regarded as outliers as defined by the 1.5 times interquartile range whiskers.

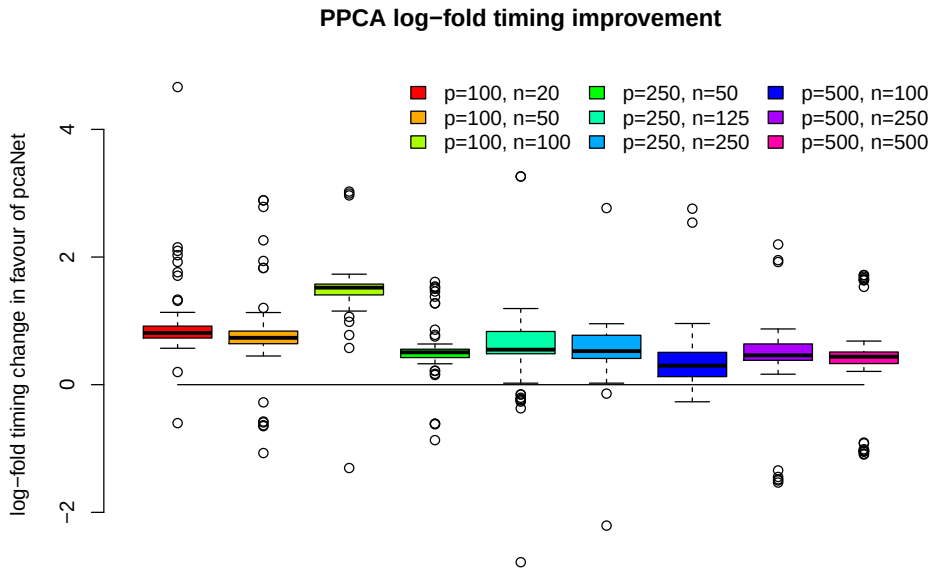


Fig. B.1 Log-fold improvement of `pcaNet` compared to `pcaMethods` for EM algorithm 1. the black horizontal line at $y = 0$ indicates the point of no difference between the timings.

This simulation demonstrates the claim that `pcaNet` provides accelerated versions of the algorithms contained within `pcaMethods`, at least for the data dimensions specified here. The acceleration that is shown could most likely be improved with further optimisation of the code, but that is beyond the scope of this work.

B.7 Model-based simulation: additional results

Figures B.2, B.3, B.4, B.5, B.6, and complement Figures 3.1 and 3.2 of Section 3.7 by providing results for $n \in (60, 80)$.

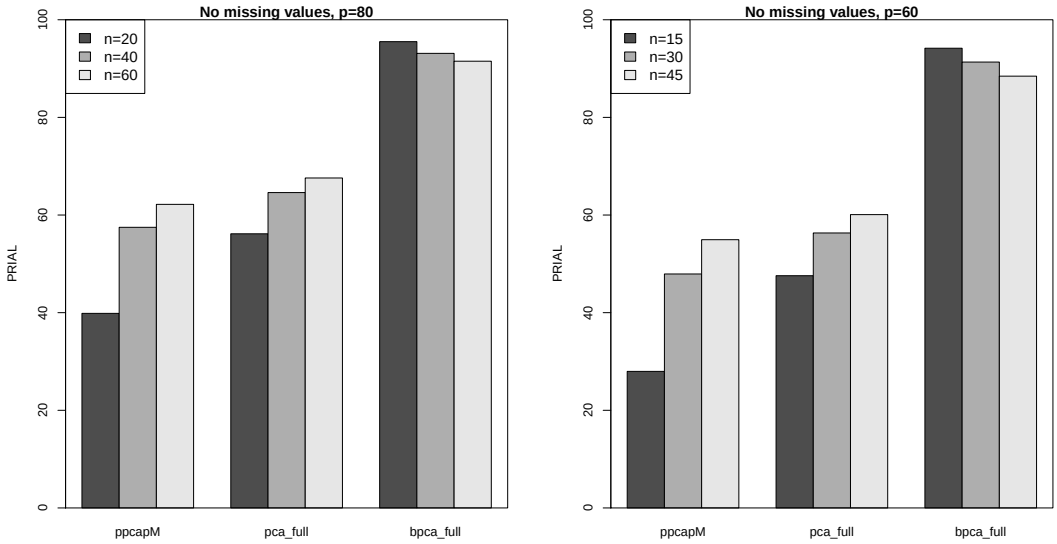


Fig. B.2 PRIAL for $p = 80,60$ with no missing values.

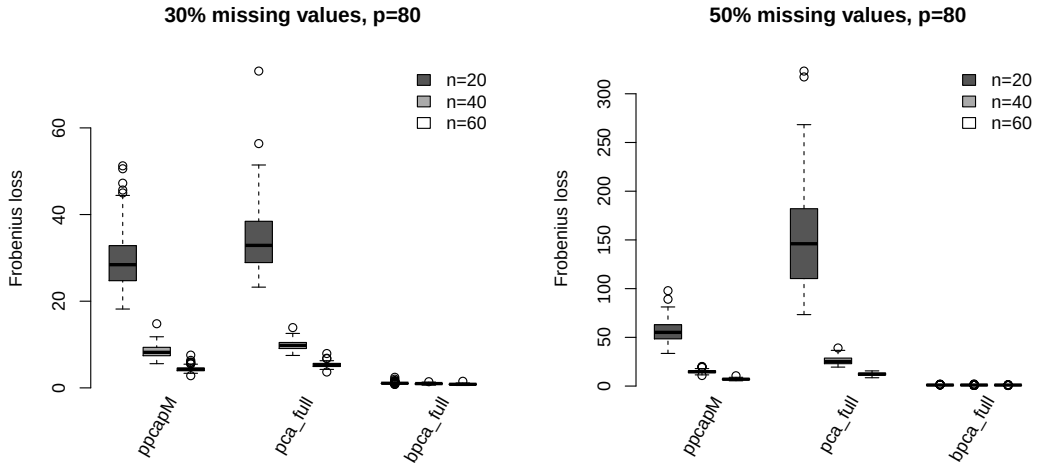


Fig. B.3 PRIAL for $p = 80$ with 30% and 50% missing values.

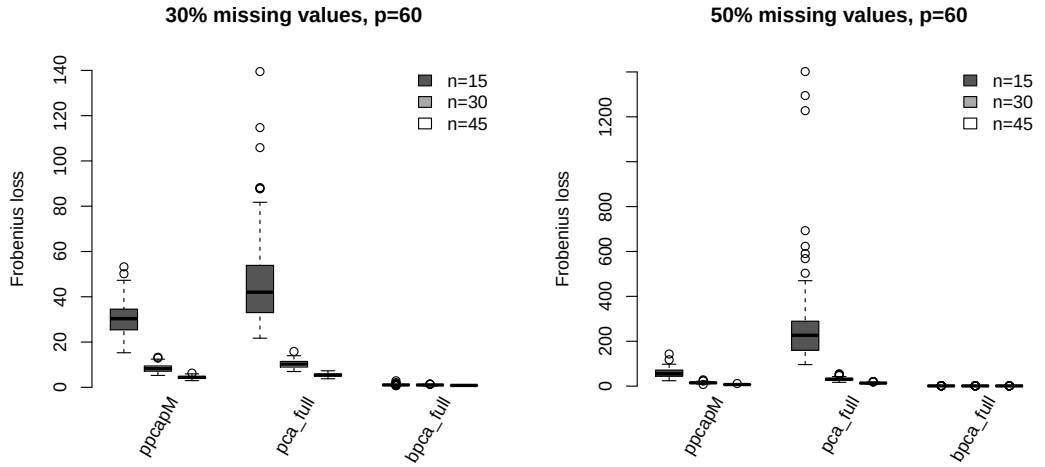


Fig. B.4 PRIAL for $p = 60$ with 30% and 50% missing values.

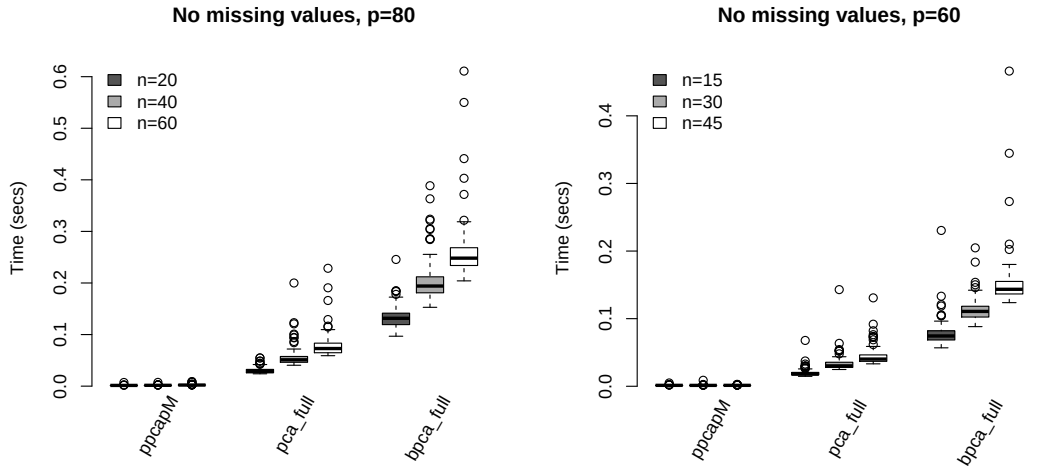


Fig. B.5 Run-time in seconds of each PPCA algorithm for $p = 80{,}60$ with no missing values.

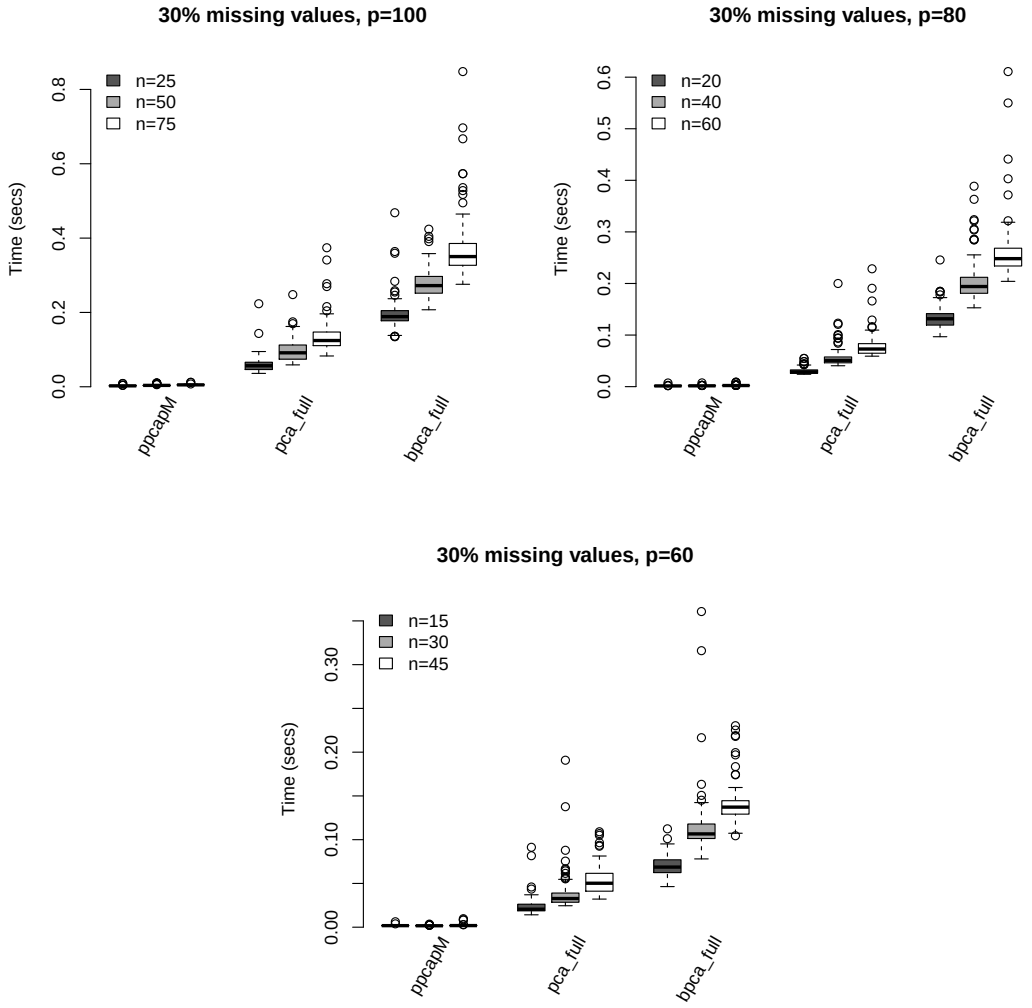


Fig. B.6 Run-time in seconds of each PPCA algorithm for $p = 100, 80, 60$ with 30% missing values.

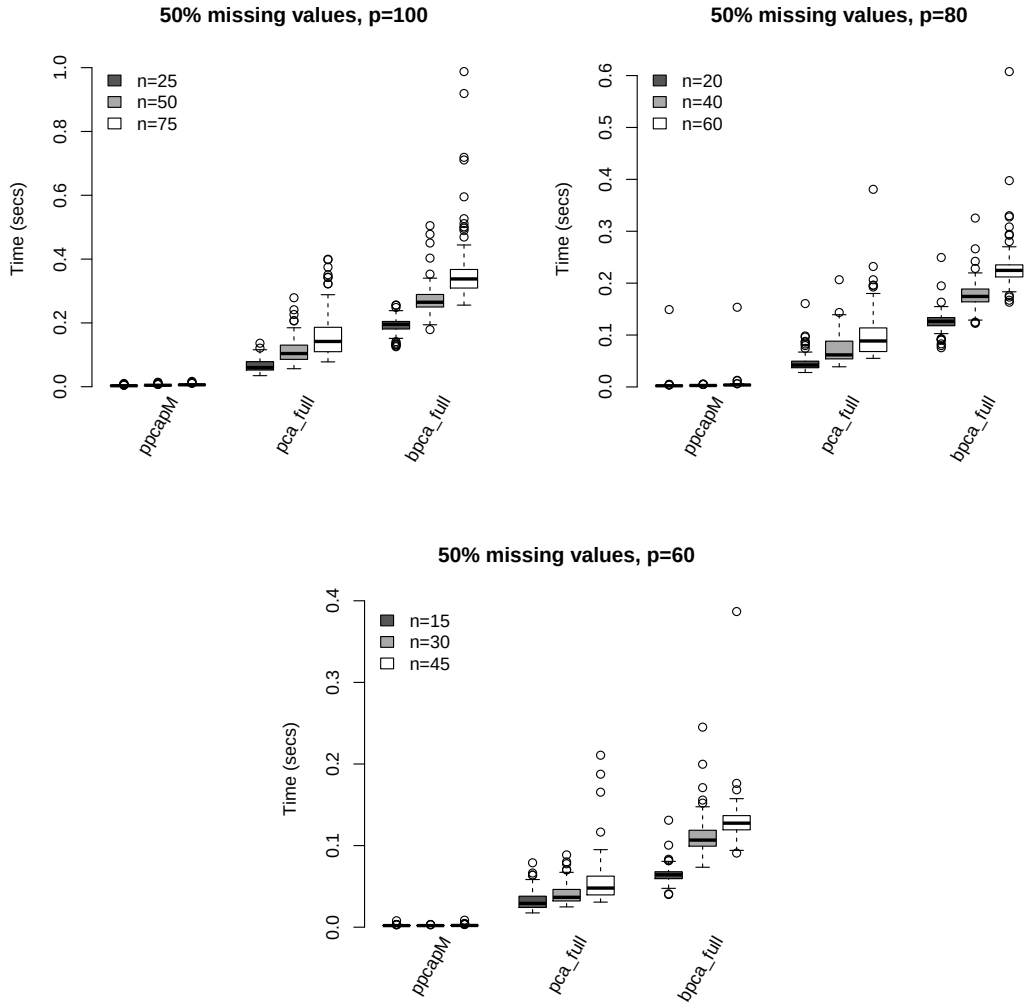
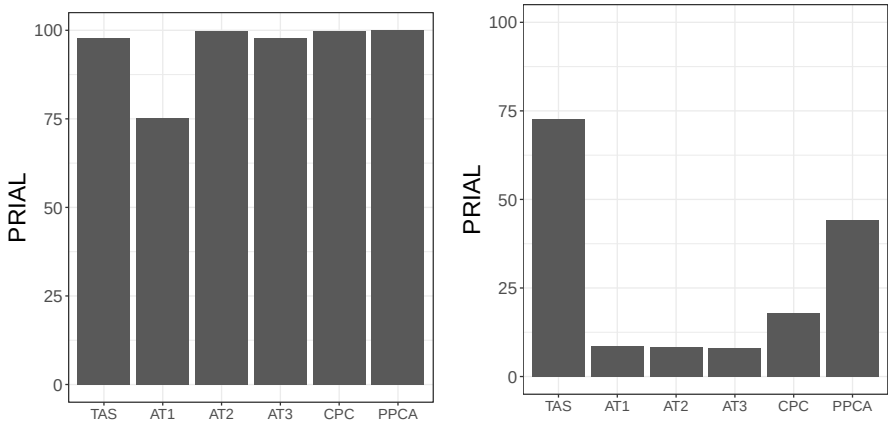


Fig. B.7 Run-time in seconds of each PPCA algorithm for $p = 100, 80, 60$ with 50% missing values.

B.8 Comparison with TAS: additional results

Figures B.8 and B.9 complements Figure 3.3 from Section 3.8 for $n \in (50, 75)$.



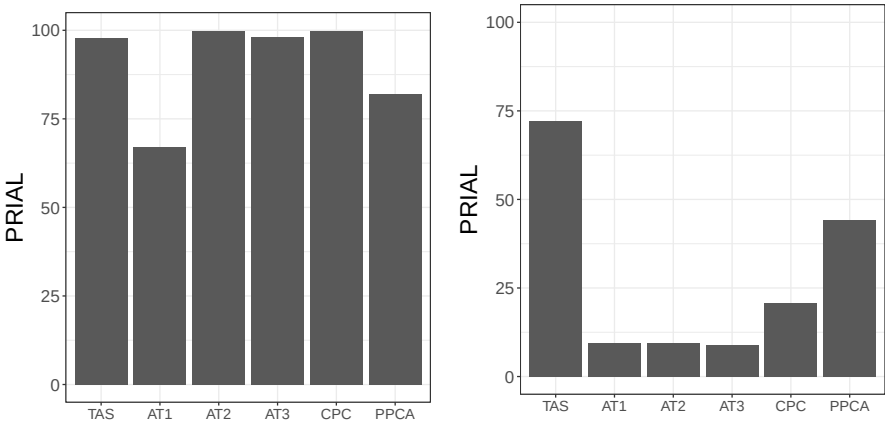
(a) Scenario 1: PRIAL

(b) Scenario 2: PRIAL

(c) Scenario 3: PRIAL

(d) Scenario 4: PRIAL

Fig. B.8 PRIAL for $p = 100$ and $n = 50$ for scenarios 1, 2, 3, and 4..



(a) Scenario 1: PRIAL

(b) Scenario 2: PRIAL

(c) Scenario 3: PRIAL

(d) Scenario 4: PRIAL

Fig. B.9 PRIAL for $p = 100$ and $n = 75$ for scenarios 1, 2, 3, and 4..

Appendix C

Case study: cytokine expression in the context of traumatic brain injury

C.1 Cytokines under study

Table contains the cytokines that were measured in the study and the corresponding abbreviations.

Cytokine	Abbreviation
Epidermal Growth Factor	EGF
Eotaxin	Eotaxin
Basic Fibroblast Growth Factor	FGF2
Fms-related tyrosine kinase 3 ligand	Flt3 lig
Fractalkine	Fractalkine
Granulocyte Colony Stimulating Factor	G-CSF
Granulocyte-Monocyte Colony Stimulating Factor	GM-CSF
GRO	GRO
Interferon alpha-2	IFNa2
Interferon gamma	IFNg
Interleukin-1 alpha	IL1a
Interleukin-1 beta	IL1b
Interleukin-1 receptor antagonist	IL1ra
Interleukin-2	IL2
Interleukin-3	IL3
Interleukin-4	IL4
Interleukin-5	IL5
Interleukin-6	IL6
Interleukin-7	IL7
Interleukin-8	IL8
Interleukin-9	IL9
Interleukin-10	IL10
Interleukin-12 subunit beta	IL12p40
Interleukin-12	IL12p70
x Interleukin-13	IL13
Interleukin-15	IL15
Interleukin-17	IL17
Chemokine (C-X-C motif) ligand 10	IP10
Monocyte Chemotactic Protein 1	MCP1
Monocyte Chemotactic Protein 3	MCP3
Macrophage Derived Chemoattractant	MDC
Macrophage Inflammatory Protein-1 alpha	MIP1a
Macrophage Inflammatory Protein-1 beta	MIP1b
Platelet Derived Growth Factor AA	PDGF AA
Platelet Derived Growth Factor AB/BB	PDGF AB/BB
RANTES	RANTES
Soluble CD40 Ligand	sCD40L
Soluble Interleuking-2 Receptor	sIL2R
Transforming Growth Factor alpha	TGFa
Transforming Necrosis Factor alpha	TNFa
Transforming Necrosis Factor beta	TNFb
Vascula Endothilial Growth Factor	VEGF

Table C.1 Cytokines measured in the study and their abbreviations.

C.2 Alternative imputation values

Here we display the results from Section 4.4 after having run kNN with a different number of neighbours.

C.2.1 $k = 5$

Figures C.1, C.2, C.3, C.4,C.5, and C.6 correspond to the Figures 4.10, 4.11, 4.12, 4.13, 4.15, and 4.16 having run the imputation algorithm with $k = 5$.

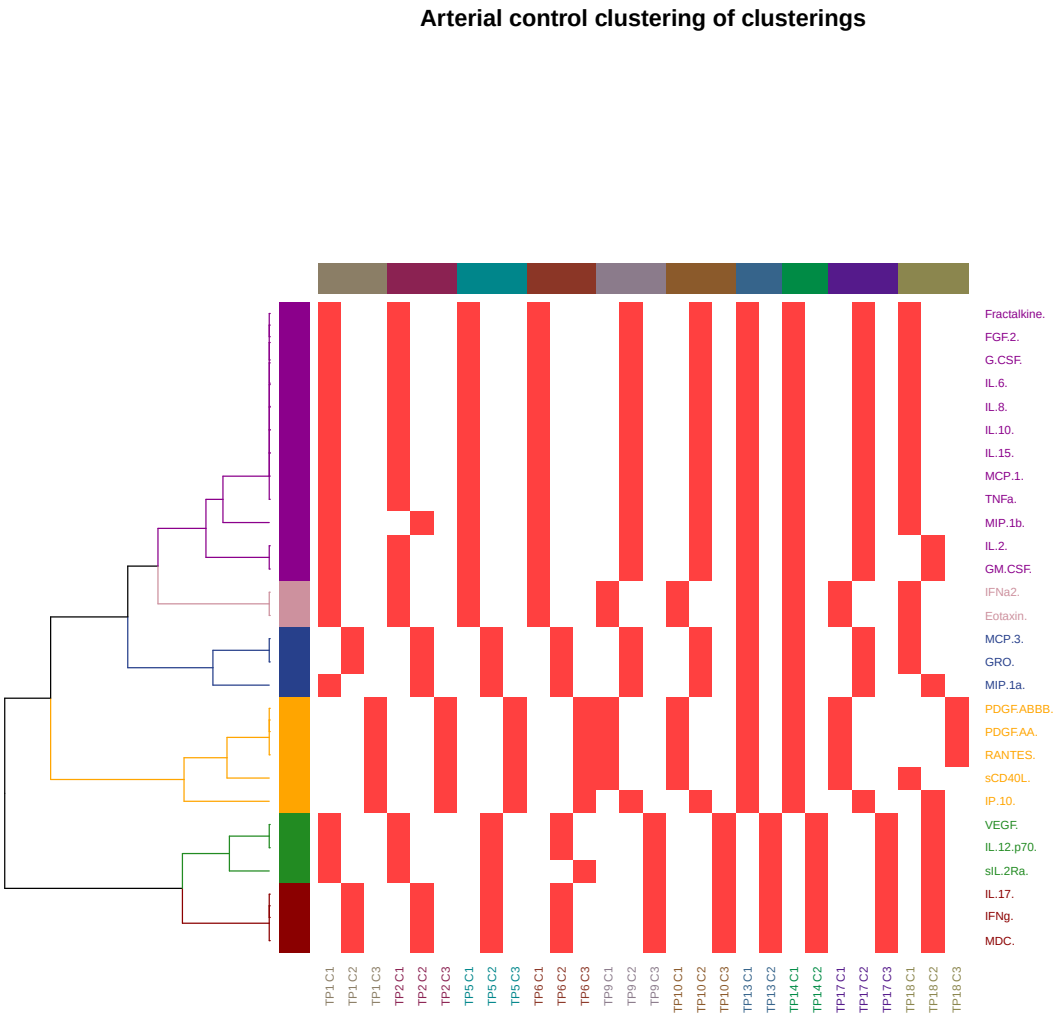


Fig. C.1 COCA for the arterial samples of the control group using TAS having imputed with $k = 5$. The labels of the columns indicate a cluster allocation for a given time point, e.g. time point 10 cluster 2 is denoted as TP10 C2. Each time point has been given its own colour for clarity.

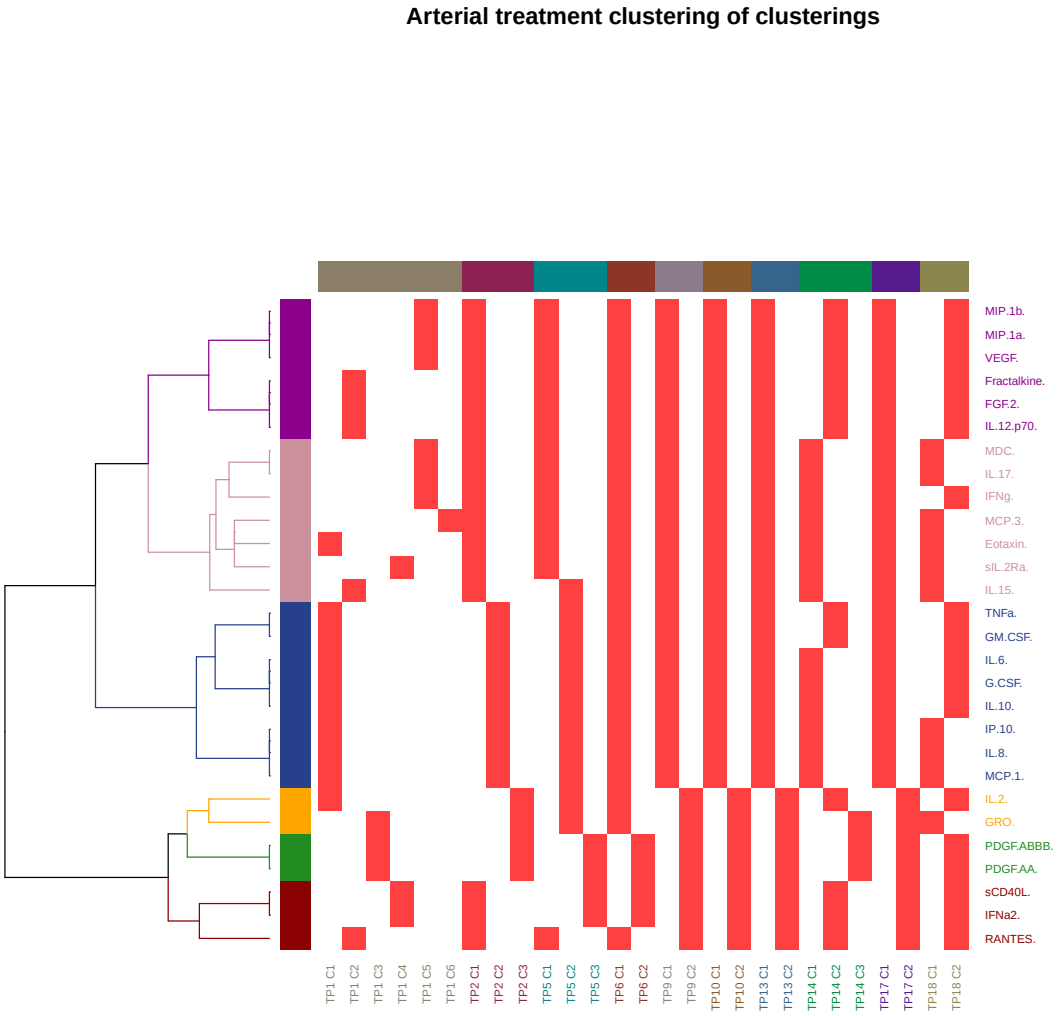


Fig. C.2 COCA for the arterial samples of the treatment group using TAS having imputed with $k = 5$. The labels of the columns indicate a cluster allocation for a given time point, e.g. time point 10 cluster 2 is denoted as TP10 C2. Each time point has been given its own colour for clarity.

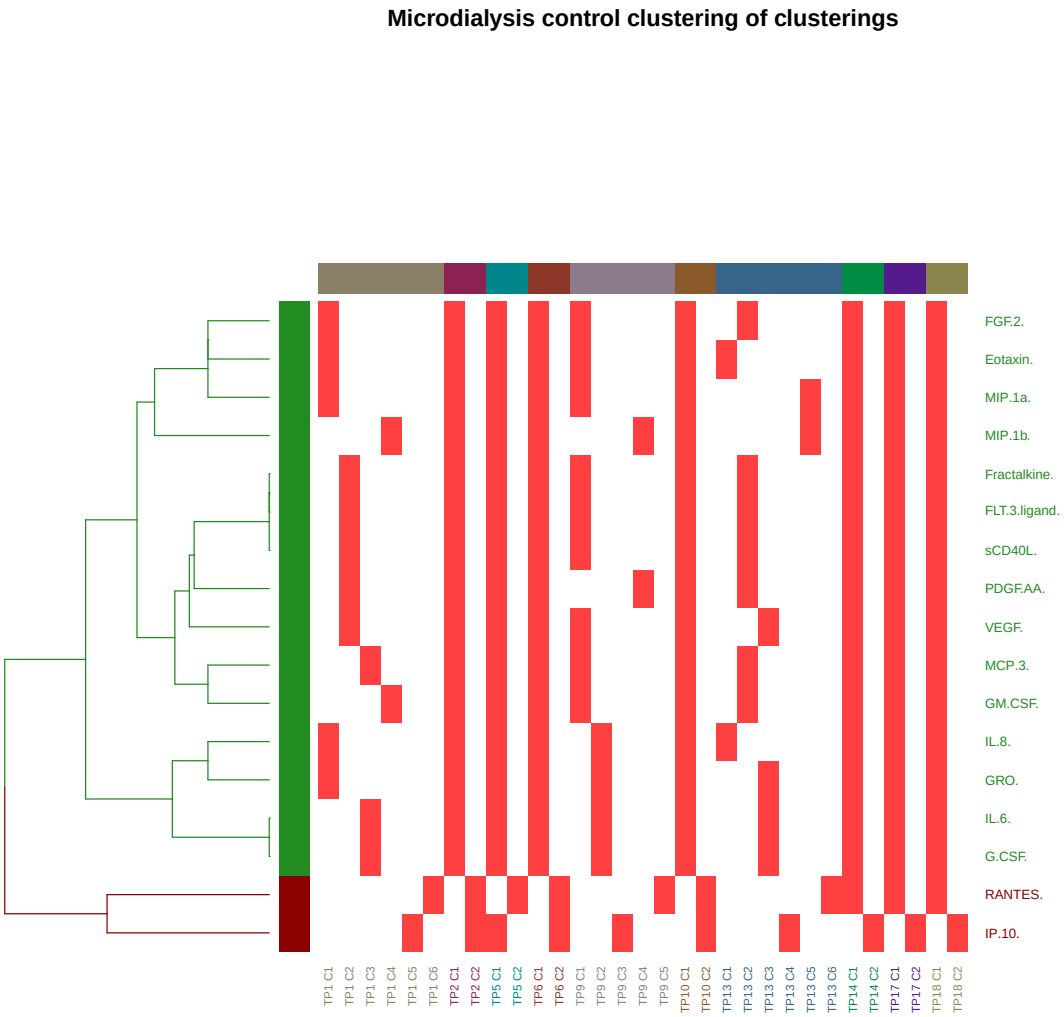


Fig. C.3 COCA for the microdialysis samples of the control group using TAS having imputed with $k = 5$. The labels of the columns indicate a cluster allocation for a given time point, e.g. time point 10 cluster 2 is denoted as TP10 C2. Each time point has been given its own colour for clarity.

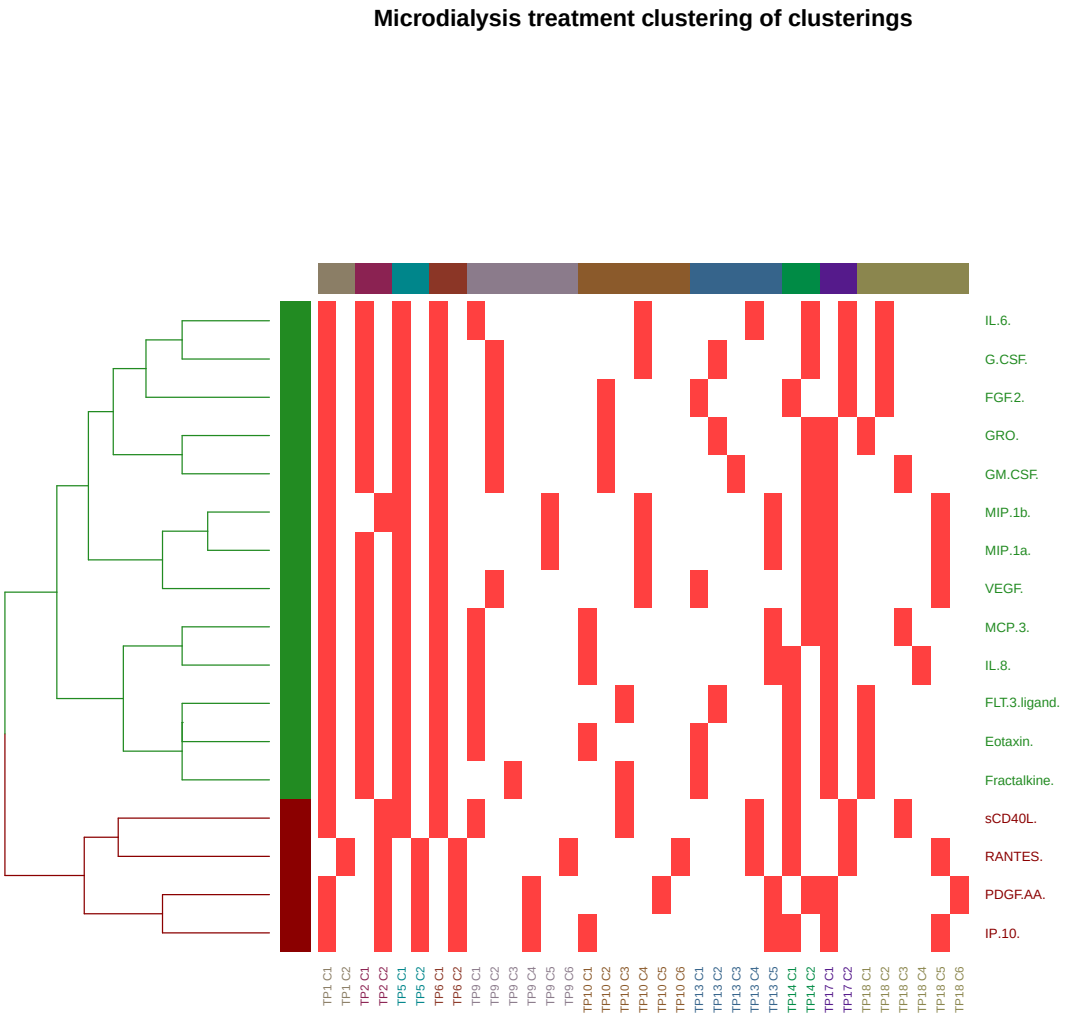


Fig. C.4 COCA for the microdialysis samples of the treatment group using TAS having imputed with $k = 5$. The labels of the columns indicate a cluster allocation for a given time point, e.g. time point 10 cluster 2 is denoted as TP10 C2. Each time point has been given its own colour for clarity.

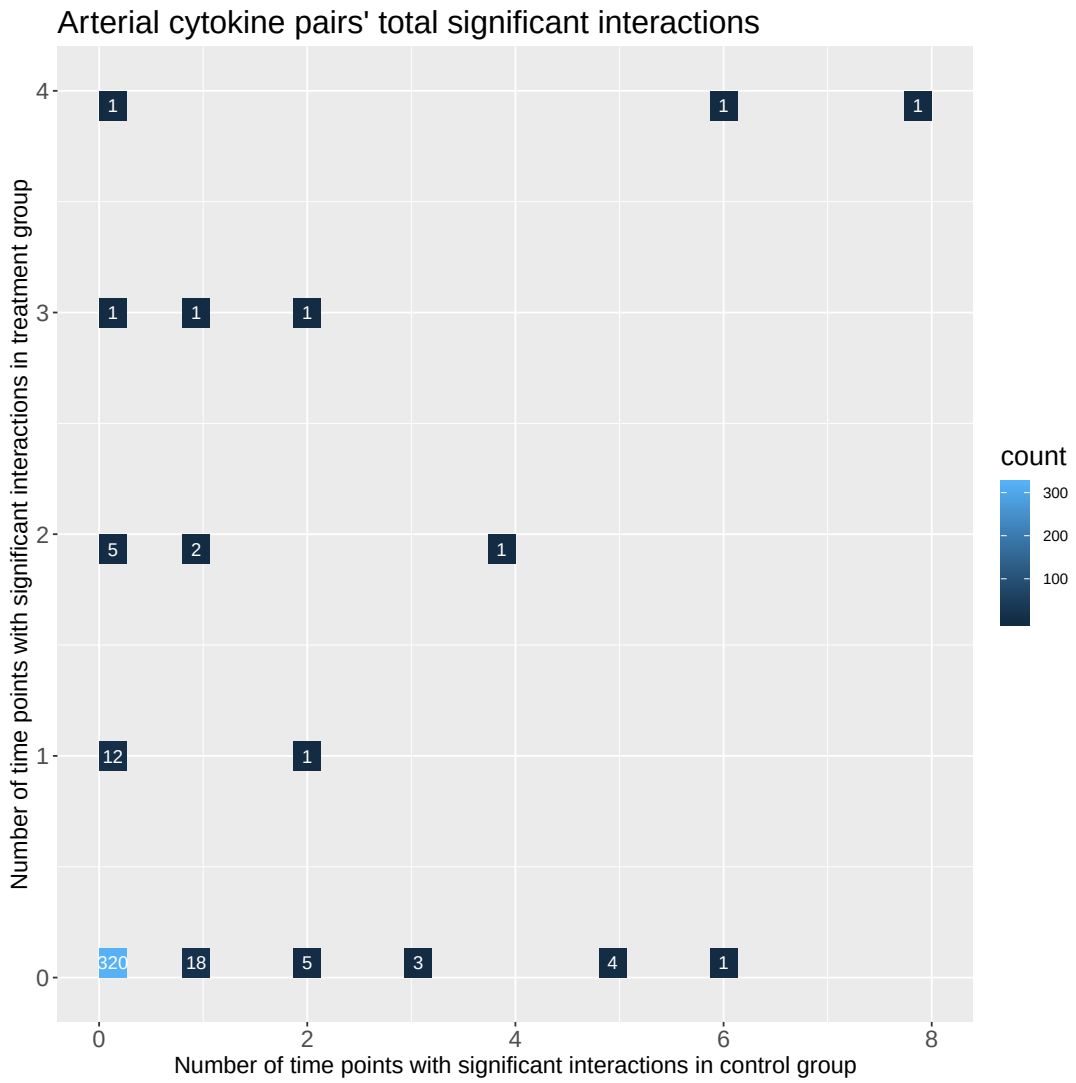


Fig. C.5 Network edge summaries for arterial samples using TAS having imputed with $k = 5$. The x-axis shows the number of times an edge appeared in the control group networks, whilst the y-axis does the same for the treatment group. The number inside the blue boxes shows how many pairs of cytokines had an edge appear that many times.

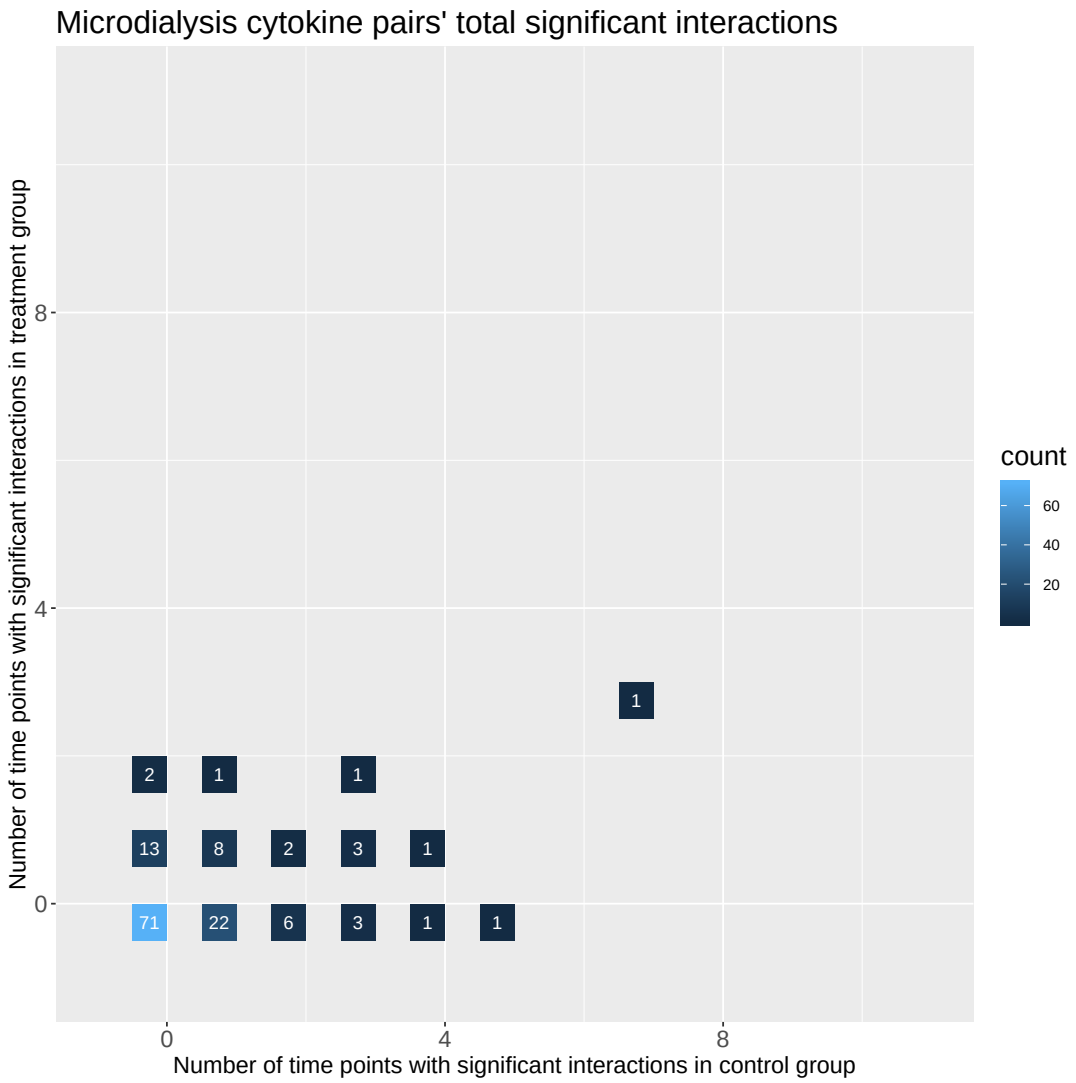


Fig. C.6 Network edge summaries for microdialysis samples using TAS having imputed with $k = 5$. The x-axis shows the number of times an edge appeared in the control group networks, whilst the y-axis does the same for the treatment group. The number inside the blue boxes shows how many pairs of cytokines had an edge appear that many times.

C.2.2 $k = 7$

Figures C.7, C.8, C.9, C.10,C.11, and C.12 correspond to the Figures 4.10, 4.11, 4.12, 4.13, 4.15, and 4.16 having run the imputation algorithm with $k = 7$.

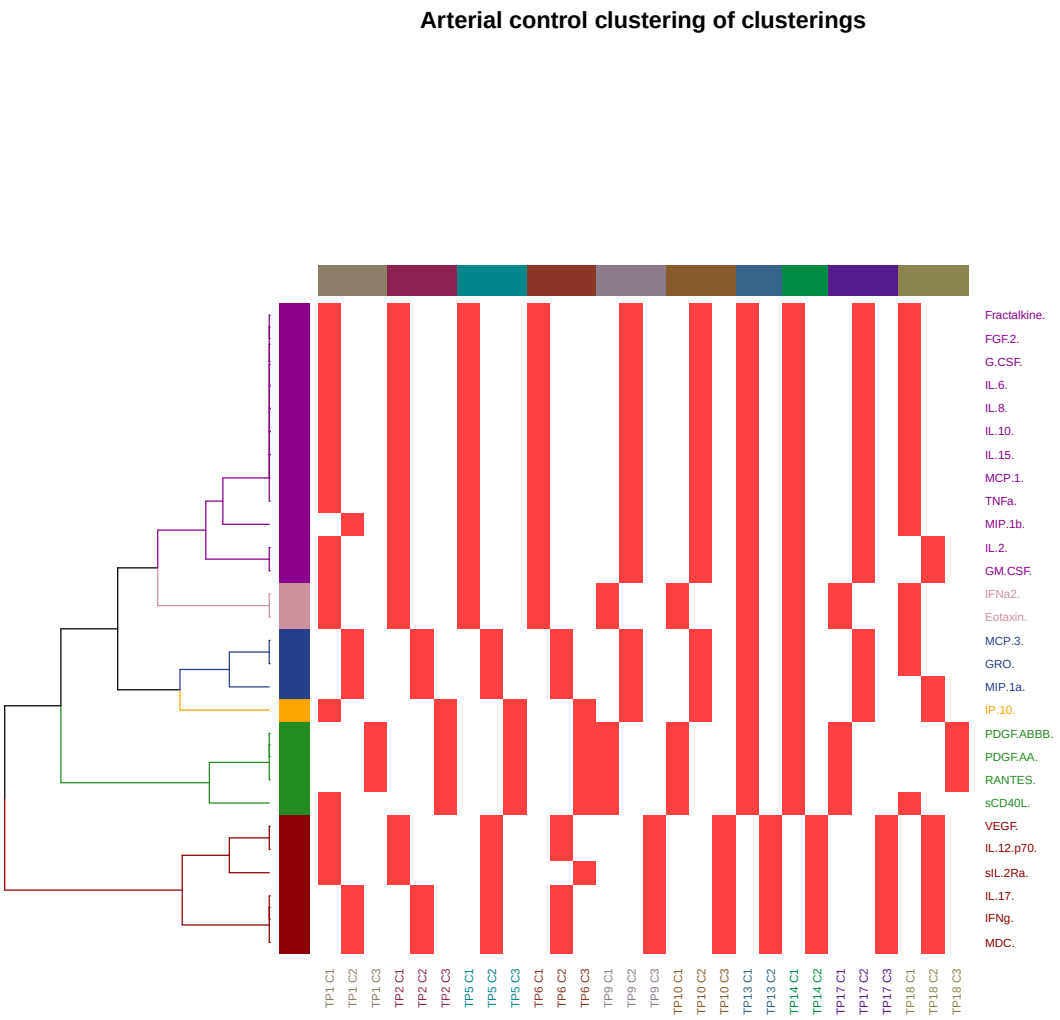


Fig. C.7 COCA for the arterial samples of the control group using TAS having imputed with $k = 7$. The labels of the columns indicate a cluster allocation for a given time point, e.g. time point 10 cluster 2 is denoted as TP10 C2. Each time point has been given its own colour for clarity.

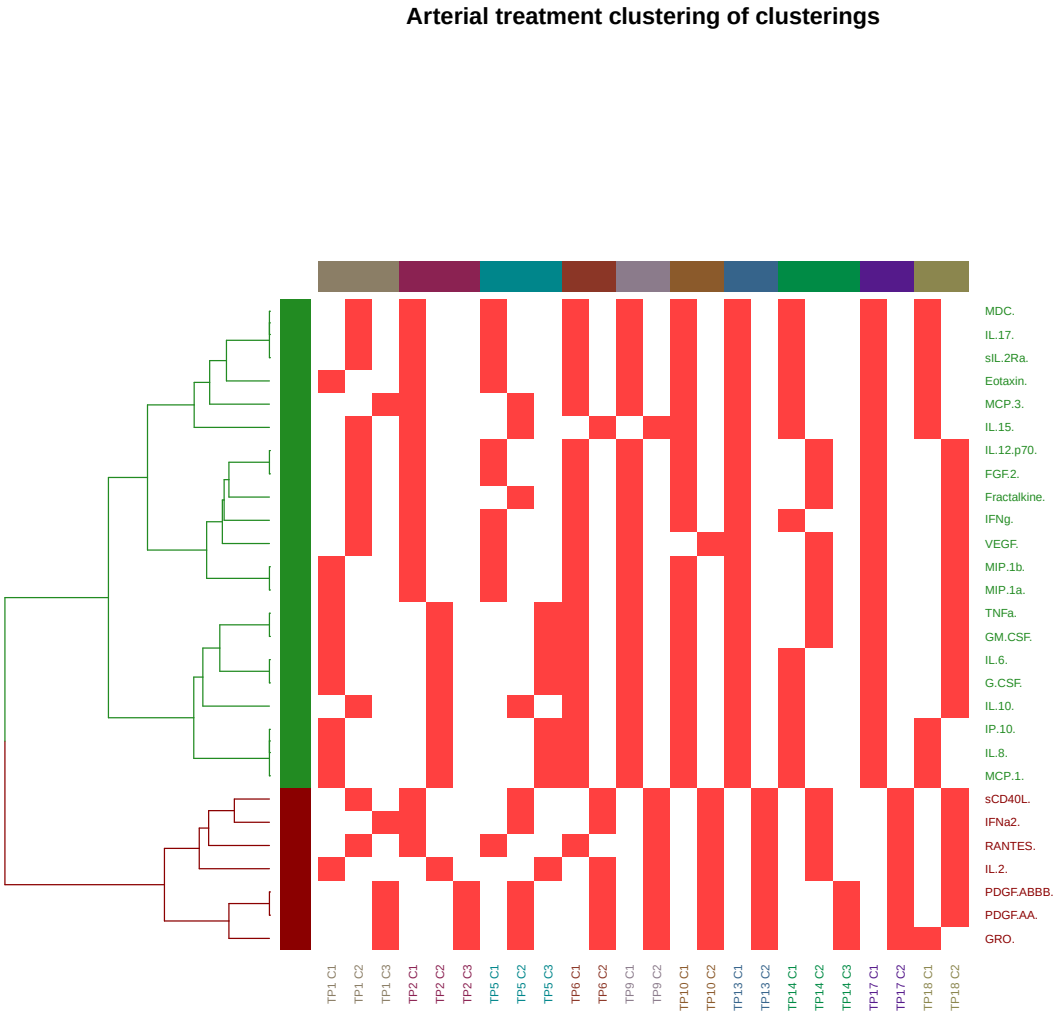


Fig. C.8 COCA for the arterial samples of the treatment group using TAS having imputed with $k = 7$. The labels of the columns indicate a cluster allocation for a given time point, e.g. time point 10 cluster 2 is denoted as TP10 C2. Each time point has been given its own colour for clarity.

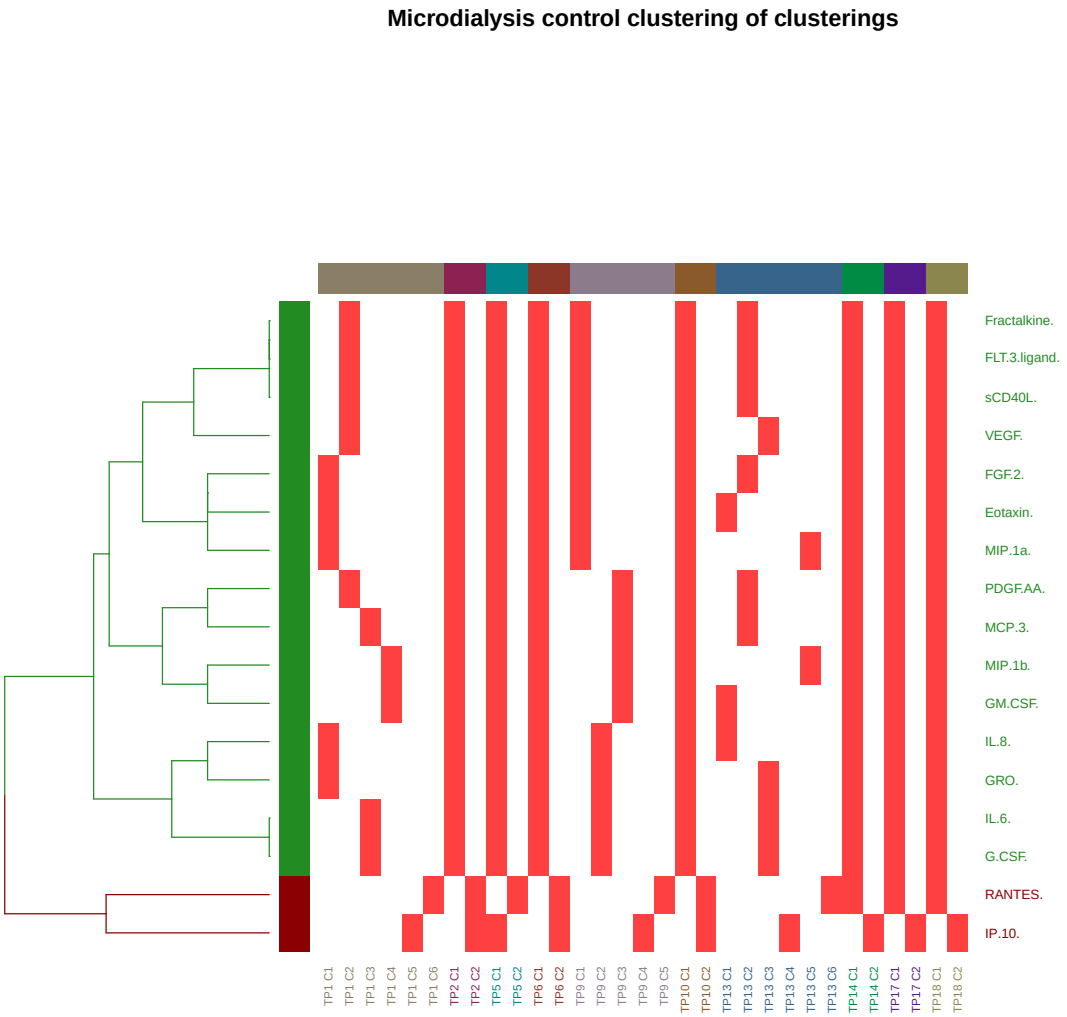


Fig. C.9 COCA for the microdialysis samples of the control group using TAS having imputed with $k = 7$. The labels of the columns indicate a cluster allocation for a given time point, e.g. time point 10 cluster 2 is denoted as TP10 C2. Each time point has been given its own colour for clarity.

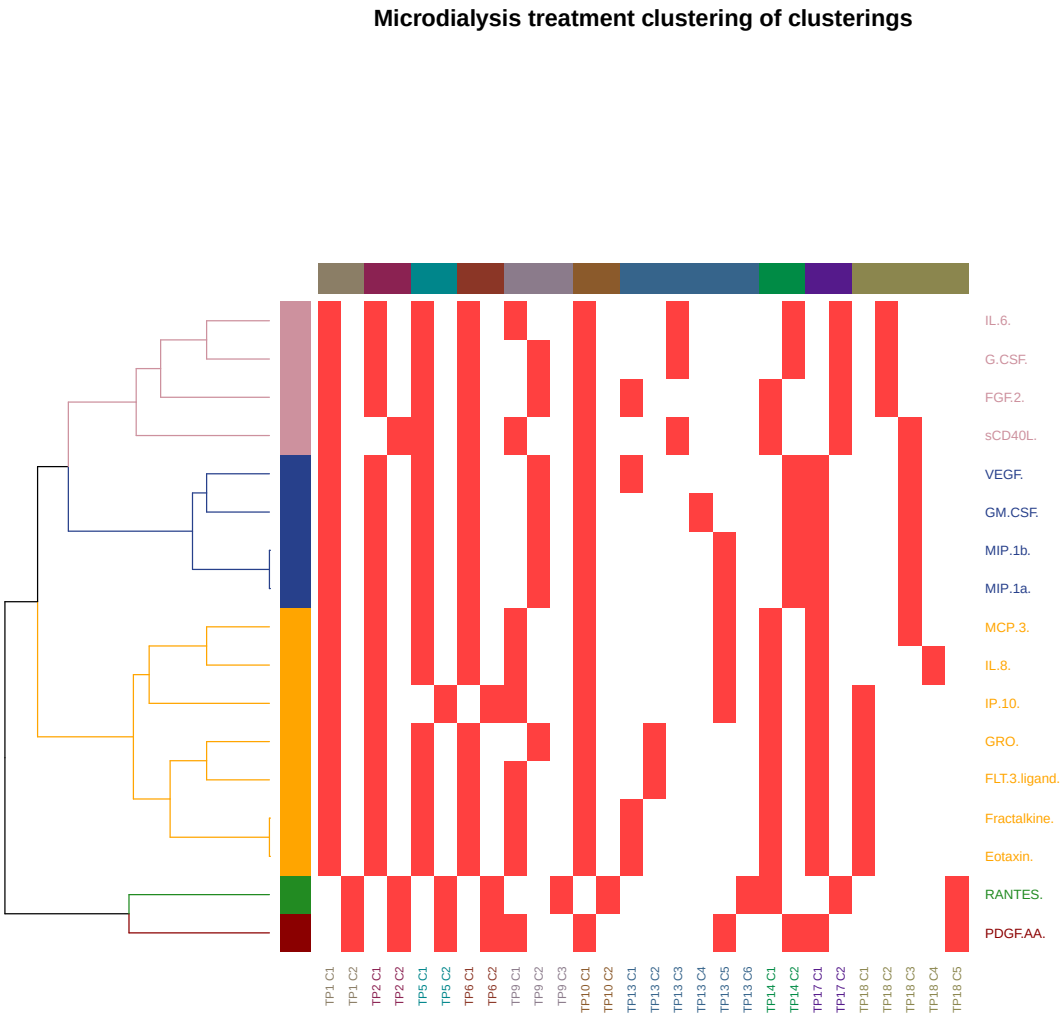


Fig. C.10 COCA for the microdialysis samples of the treatment group using TAS having imputed with $k = 7$. The labels of the columns indicate a cluster allocation for a given time point, e.g. time point 10 cluster 2 is denoted as TP10 C2. Each time point has been given its own colour for clarity.

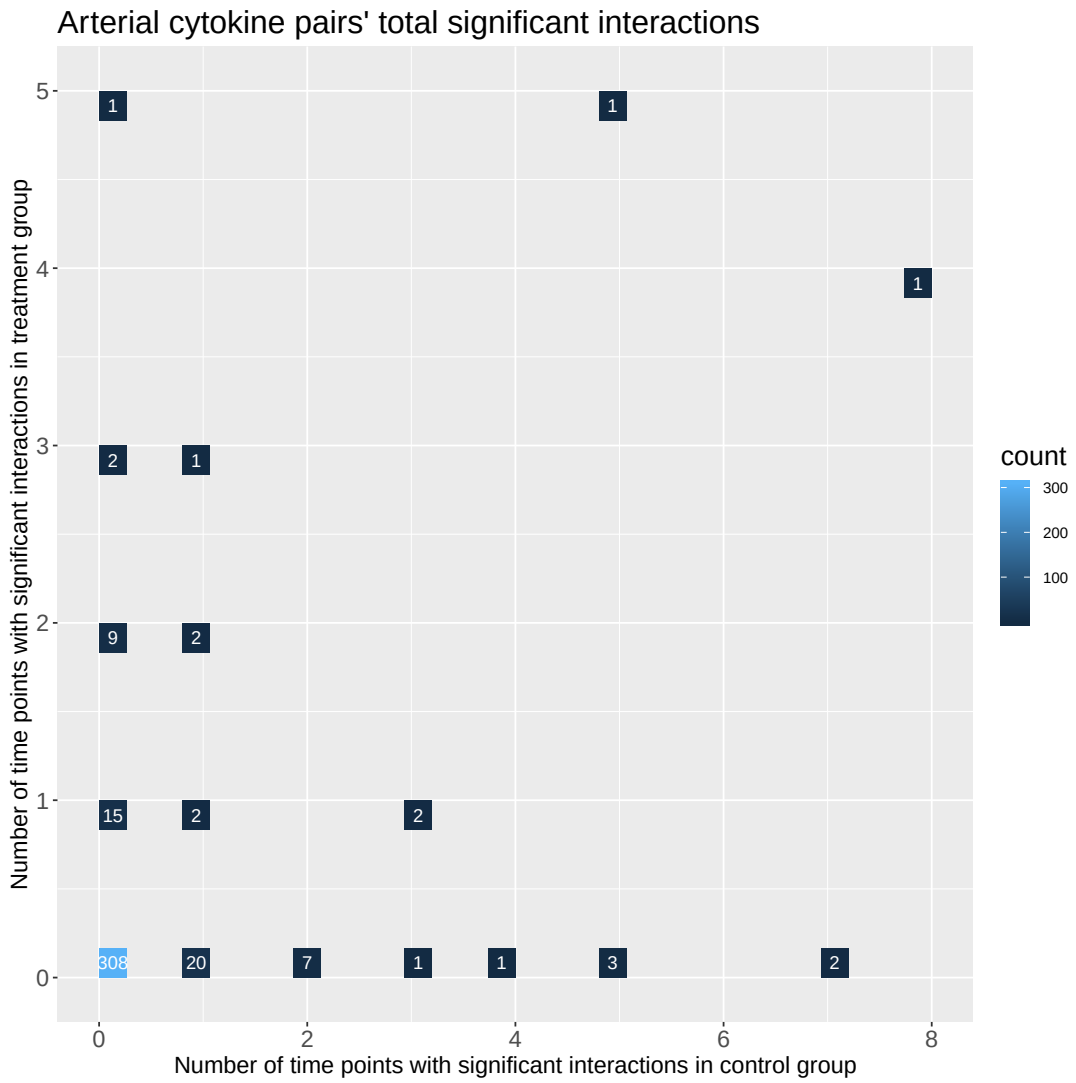


Fig. C.11 Network edge summaries for arterial samples using TAS having imputed with $k = 7$. The x-axis shows the number of times an edge appeared in the control group networks, whilst the y-axis does the same for the treatment group. The number inside the blue boxes shows how many pairs of cytokines had an edge appear that many times.

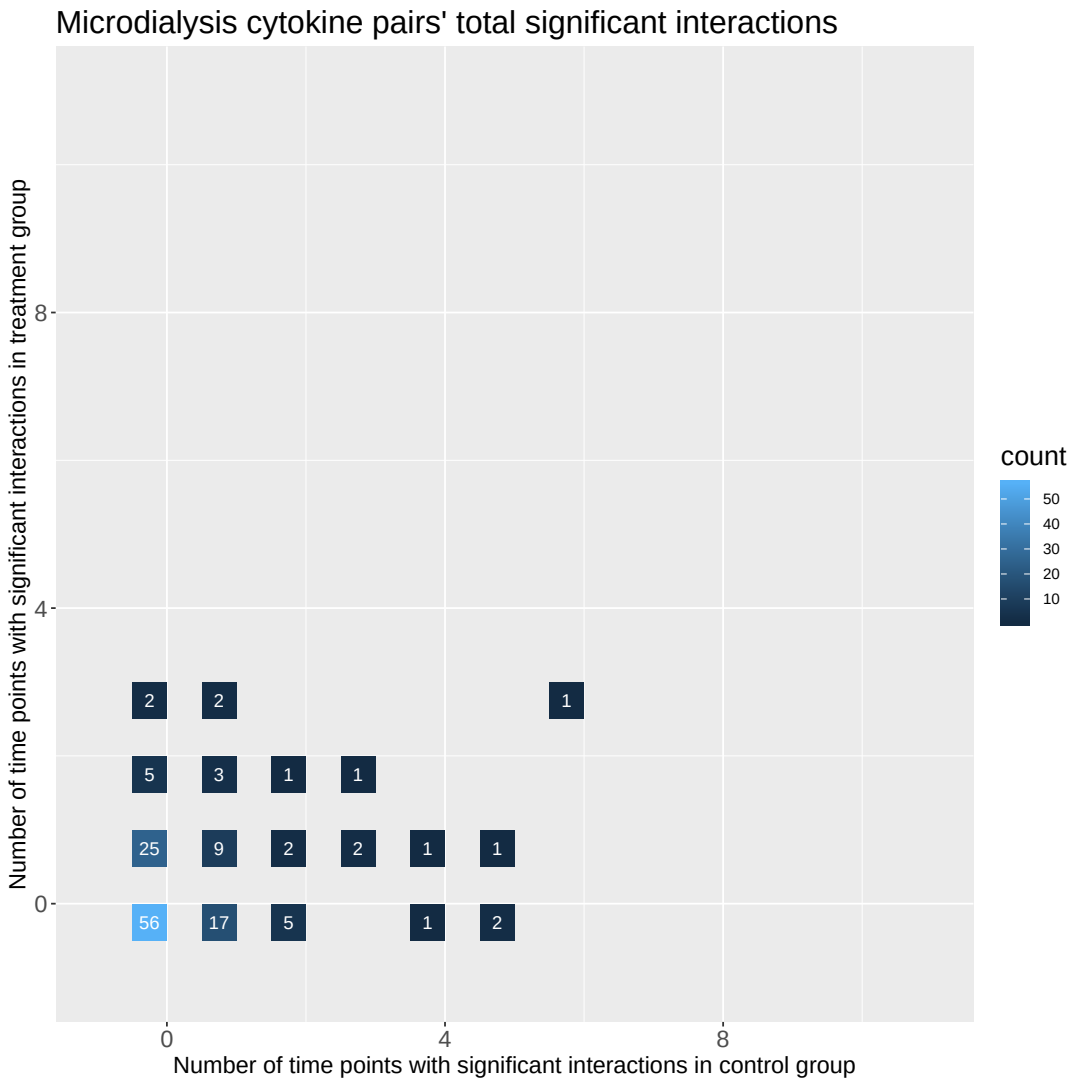


Fig. C.12 Network edge summaries for microdialysis samples using TAS having imputed with $k = 7$. The x-axis shows the number of times an edge appeared in the control group networks, whilst the y-axis does the same for the treatment group. The number inside the blue boxes shows how many pairs of cytokines had an edge appear that many times.

Appendix D

`TAS` package documentation

In this Appendix we provide the documentation for the `TAS` software package.

Package 'TAS'

September 27, 2019

Type Package

Title Target-Averaged Linear Shrinkage (TAS)

Version 1.0

Date 2019-09-27

Author Harry Gray

Maintainer Harry Gray <h.w.gray@dundee.ac.uk>

Description High-dimensional covariance matrix estimation
using linear shrinkage with multiple target matrices Gray, H.,
Leday, G.G.R., Vallejos, C.A. and Richardson, S.,
(2018) <arXiv:1809.08024>.
Multiple targets can be useful when
there is uncertainty around the choice of target or if there is
external data that can be used to construct a target matrix.

License GPL-3 | file LICENSE

Imports Rcpp (>= 0.12.13), matrixStats

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.1.1

URL https://github.com/HGray384/TAS

BugReports https://github.com/HGray384/TAS/issues

NeedsCompilation yes

R topics documented:

TAS-package *Target-Averaged Linear Shrinkage estimation*

Description

Conjugate Bayesian covariance matrix estimation using linear shrinkage with multiple target matrices (Gray et al., 2018). Most useful in high-dimensional data settings, where the number of variables is greater than the number of samples.

Details

This package contains functions for covariance estimation using a conjugate Bayesian model. Whilst the main functionality of the package is for multiple target linear shrinkage estimation, we also provide functionality for the single target analogue (Hannart and Naveau, 2014; Gray et al., 2018).

These shrinkage methods perform best when an external dataset is used to create a target matrix/target matrices that is informative of the actual dataset under examination. An example of this utility is provided in Gray et al. (2018), in which high-dimensional protein covariance matrices for various cancer types are greatly informed by large sample covariance matrices from 'similar' cancer types.

Author(s)

Harry Gray

Maintainer: Harry Gray <h.w.gray@dundee.ac.uk>

References

Gray, H., Leday, G.G.R., Vallejos, C.A. and Richardson, S., 2018. Shrinkage estimation of large covariance matrices using multiple shrinkage targets. arXiv preprint.

Hannart, A. and Naveau, P., 2014. Estimating high dimensional covariance matrices: A new look at the Gaussian conjugate framework. Journal of Multivariate Analysis, 131, pp.149-162. doi.

addTarget *Add a new target to TAS without re-running taShrink()*

Description

Add a new target to TAS without re-running taShrink()

Usage

```
addTarget(X, TASoutput, NEWtarget)
```

Arguments

X	matrix – data matrix with variables in rows and observations in columns. This method performs best when there are more variables than observations.
TASoutput	list – output from the taShrink function.
NEWtarget	matrix – a new target to add to the shrinkage estimation. Must have the same dimensions as the other targets.

Value

list – the updated TAS output having added the new target matrix to the target set.

See Also

taShrink

Examples

```
set.seed(102)
X <- matrix(rnorm(50), 10, 5) # p=10, n=5, identity covariance
X <- t(scale(t(X), center=TRUE, scale=FALSE)) # mean 0
targets <- getTargetSet(X)[,,c(1, 4, 7)] # use unit variance targets
alpha <- seq(0.01, 0.99, 0.01)
tas <- taShrink(X, targets = targets[,,c(1, 3)], plots = FALSE)
tw1 <- targetWeights(tas)
barplot(tw1, names.arg = c("target1", "target2", "S"),
main = "Target-specific shrinkage weights",
col = c("red", "green", "purple"), space = 0,
xlab = "Target", ylab = "Weight")
tas2 <- addTarget(X, tas, targets[,,2])
tw2 <- targetWeights(tas2)
par(mfrow=c(1, 2))
barplot(tw1, names.arg = c("target1", "target2", "S"),
main = "Target-specific shrinkage weights",
col = c("red", "green", "purple"), space = 0,
xlab = "Target", ylab = "Weight")
barplot(tw2, names.arg = c("target1", "target2", "target3", "S"),
main = "Target-specific shrinkage weights",
col = c("red", "green", "blue", "purple"), space = 0,
xlab = "Target", ylab = "Weight")
par(mfrow=c(1, 1))
plot(alpha, tas2$logmarginals[1,], col = 'red', pch = 16,
ylab = "log marginal likelihoods", xlab = expression(alpha))
points(alpha, tas2$logmarginals[2,], col = 'green', pch = 16)
points(alpha, tas2$logmarginals[3,], col = 'blue', pch = 16)
legend('bottomright', c("target1", "target2", "target3"), pch = 16,
  col=c('red', 'green', 'blue'))
```

---

gcShrink — *Bayesian Gaussian conjugate (GC) single target linear shrinkage covariance estimator*

Description

Implements a Bayesian Gaussian conjugate (GC) single target linear shrinkage covariance estimator as in Gray et al. (2018) and Hannart and Naveau (2014). It is most useful when the observed data is high-dimensional (more variables than observations) and allows a user-specified target matrix.

Usage

```
gcShrink(X, target = "none", var = 2, cor = 1, alpha = seq(0.01,
  0.99, 0.01), plots = TRUE, weighted = FALSE, ext.data = FALSE)
```

Arguments

X	matrix – data matrix with variables in rows and observations in columns. This method performs best when there are more variables than observations.
target	character or matrix – if "none" then a default target specified by var and cor will be used for shrinkage. If matrix then this will be used as the target for shrinkage. The target must be a real symmetric positive definite matrix.
var	numeric – c(1, 2, 3) variance structure for the target matrix. 1 sets all variances equal to 1. 2 sets all variances equal to their sample mean. 3. sets all variances to their sample values.
cor	numeric – c(1, 2, 3) correlation structure for the target matrix. 1 sets the correlations to 0. 2 sets the correlations equal to their sample mean. 3 sets the correlations equals to an autocorrelation structure with parameter equal to the sample mean.
alpha	list – the grid of shrinkage intensities in (0, 1) to be used. Recommended to be an equidistant grid that covers the whole interval. A short comparison of estimation accuracy versus granularity is provided in ...
plots	logical – if TRUE then plots the log-marginal likelihood for each value of alpha with the value of alpha that maximises this highlighted.
weighted	logical – if TRUE then average over all values of alpha and their respective marginal likelihood value as in Gray et al (submitted). If FALSE then only use the value of alpha that maximises the log-marginal likelihood as in Hannart and Naveau (2014).
ext.data	matrix – an external data matrix used a surrogate to estimate the parameters in the default target set for X. Never recommended unless there is a belief that ext.data is informative of the covariances of X.

Value

list –

sigmahat matrix – the estimated covariance matrix.

optimalpha numeric – the value of alpha that maximises the log-marginal likelihood.

target matrix – the target matrix used for shrinkage.

logmarg numeric – the values of the log marginal likelihood for each (target, alpha) pair.

References

Gray, H., Leday, G.G., Vallejos, C.A. and Richardson, S., 2018. Shrinkage estimation of large covariance matrices using multiple shrinkage targets. arXiv preprint.

Hannart, A. and Naveau, P., 2014. Estimating high dimensional covariance matrices: A new look at the Gaussian conjugate framework. Journal of Multivariate Analysis, 131, pp.149-162. doi.

Examples

```
set.seed(102)
X <- matrix(rnorm(50), 10, 5) # p=10, n=5, identity covariance
X <- t(scale(t(X), center=TRUE, scale=FALSE)) # mean 0
t1 <- gcShrink(X, var=1, cor=1) # apply shrinkage and view likelihood for T1
t2 <- gcShrink(X, var=2, cor=2) # apply shrinkage and view likelihood for T2
norm(t1$sigmahat-diag(10), type="F") # calculate loss
```

```
norm(t2$sigmahat-diag(10), type="F") # calculate loss
# one target clearly better but how to choose this a priori?
```

`getTarget` *Construct a target matrix for single target linear shrinkage*

Description

Construct a popular target matrix from the linear shrinkage literature. These targets consist of a combination of variance and correlation structure. Possible variance structures are unit, sample mean, and sample. Possible correlation structures are zero, sample mean, and autocorrelation.

Usage

```
getTarget(X, varNumber = 2L, corNumber = 1L)
```

Arguments

`X`	`matrix` – data matrix with variables in rows and observations in columns.
`varNumber`	`numeric` – c(1, 2, 3) variance structure for the target matrix. 1 sets all variances equal to 1. 2 sets all variances equal to their sample mean (using `X`). 3. sets all variances to their sample values (using `X`).
`corNumber`	`numeric` – c(1, 2, 3) correlation structure for the target matrix. 1 sets the correlations to 0. 2 sets the correlations equal to their sample mean (using `X`). 3 sets the correlations equals to an autocorrelation structure with parameter equal to the sample mean (using `X`).

Value

`matrix` – target matrix for linear shrinkage estimation.

See Also

`gcShrink`

Examples

```
set.seed(102)
X <- matrix(rnorm(50), 10, 5) # p=10, n=5, identity covariance
X <- t(scale(t(X), center=TRUE, scale=FALSE)) # mean 0
getTarget(X, varNumber = 1, corNumber = 1) # unit variance, zero correlation
getTarget(X, varNumber = 2, corNumber = 1) # equal variance, zero correlation
getTarget(X, varNumber = 3, corNumber = 1) # sample variances, zero correlation
```

getTargetSet *Construct a set of target matrices for Target-Averaged linear shrinkage*

Description

Construct a set of popular target matrices from the linear shrinkage literature. These nine targets consist of the combinations of variance and correlation structures; variance structures are unit, sample mean, and sample; correlation structures are zero, sample mean, and autocorrelation.

Usage

```
getTargetSet(X)
```

Arguments

X	matrix – data matrix with variables in rows and observations in columns.

Value

array – a pxpx9 array of target matrices, where p is the number of variables of X.

See Also

taShrink

Examples

```
set.seed(102)
X <- matrix(rnorm(50), 10, 5) # p=10, n=5, identity covariance
X <- t(scale(t(X), center=TRUE, scale=FALSE)) # mean 0
ts <- getTargetSet(X) # an array of targets
# inspect the variances of the targets
vars <- apply(ts, 3, diag)
colnames(vars) <- paste("target", c(1:9), sep="")
vars
boxplot(vars, ylab = "variances")
# inspect the correlations of the targets
corrs <- apply(ts, 3, function(x){cov2cor(x)[lower.tri(x)]})
colnames(corrs) <- paste("target", c(1:9), sep="")
corrs
boxplot(corrs, ylab = "correlations")
```

logML *Log-marginal likelihood of a Gaussian-inverse Wishart conjugate model*

Description

Evaluate the log-marginal likelihood of a Gaussian-inverse Wishart distribution parametrised in terms of its prior mean matrix and its prior variance parameter. In the Bayesian linear shrinkage model, these parameters correspond to the target matrix and the shrinkage intensity (Hannart and Naveau, 2014).

Usage

```
logML(X, target, alpha)
```

Arguments

X	matrix –data matrix with variables in rows and observations in columns.
target	matrix – prior mean matrix parameter of the inverse-Wishart distribution.
alpha	numeric – prior variance parameter of the inverse-Wishart distribution.

Value

numeric – log-marginal likelihood evaluated at (target, alpha). If alpha is a vector is a vector then the function returns a vector evaluated at each element of alpha.

References

Alexis Hannart and Philippe Naveau (2014). Estimating high dimensional covariance matrices: A new look at the Gaussian conjugate framework. Journal of Multivariate Analysis. doi.

See Also

gcShrink, taShrink

Examples

```
set.seed(102)
X <- matrix(rnorm(50), 10, 5) # p=10, n=5, identity covariance
X <- t(scale(t(X), center=TRUE, scale=FALSE)) # mean 0
target <- getTarget(X)
alpha <- seq(0.01, 0.99, 0.01)
lml <- logML(X, target, alpha)
plot(alpha, lml, col = 'blue', pch = 16,
ylab = "log marginal likelihoods", xlab = expression(alpha))
lines(x = rep(alpha[which(lml==max(lml))], 2), y = c(min(lml), max(lml)), col='red')
```

---

targetWeights — *Extract the target-specific and sample covariance shrinkage weights from TAS output*

Description

Extract the target-specific and sample covariance shrinkage weights from TAS output

Usage

```
targetWeights(TASoutput)
```

Arguments

TASoutput	list – output from the taShrink function.

Value

`list` – the weights from each target and sample covariance matrix in TAS.

See Also

taShrink

Examples

```
set.seed(102)
X <- matrix(rnorm(50), 10, 5) # p=10, n=5, identity covariance
X <- t(scale(t(X), center=TRUE, scale=FALSE)) # mean 0
targets <- getTargetSet(X)[,,c(1, 4, 7)] # use unit variance targets
tas <- taShrink(X, targets = targets[,,c(1, 3)], plots = FALSE)
tw1 <- targetWeights(tas)
barplot(tw1, names.arg = c("target1", "target2", "S"),
main = "Target-specific shrinkage weights",
col = c("red", "green", "purple"), space = 0,
xlab = "Target", ylab = "Weight")
tas2 <- addTarget(X, tas, targets[,,2])
tw2 <- targetWeights(tas2)
par(mfrow=c(1, 2))
barplot(tw1, names.arg = c("target1", "target2", "S"),
main = "Target-specific shrinkage weights",
col = c("red", "green", "purple"), space = 0,
xlab = "Target", ylab = "Weight", ylim = c(0, 0.6))
barplot(tw2, names.arg = c("target1", "target2", "target3", "S"),
main = "Target-specific shrinkage weights",
col = c("red", "green", "blue", "purple"), space = 0,
xlab = "Target", ylab = "Weight", ylim = c(0, 0.6))
par(mfrow=c(1, 1))
```

---

taShrink *Bayesian Target-Averaged linear Shrinkage (TAS) covariance estimator*

Description

Implements a Bayesian target-averaged linear shrinkage covariance estimator as in Gray et al. (2018). It is most useful when the observed data is high-dimensional (more variables than observations) and there are other datasets that can be used to include as prior data-driven targets to shrink towards.

Usage

```
taShrink(X, targets = "default", without = 0, alpha = seq(0.01, 0.99,
  0.01), plots = TRUE, ext.data = FALSE)
```

Arguments

X	`matrix` – data matrix with variables in rows and observations in columns. This method performs best when there are more variables than observations.

targets	character or array – "default" creates a target set of common literature targets, or the user may specify an array of targets to use, e.g. ones that have been derived from external data. All targets must be real symmetric positive definite matrices.
without	list – if targets=="default" then this indicates which of the default targets should be excluded from shrinkage. This can be useful when exploring the shrinkage behaviour with a subset of targets (e.g. through simulation).
alpha	list – the grid of shrinkage intensities in (0, 1) to be used. Recommended to be an equidistant grid that covers the whole interval. A short comparison of estimation accuracy versus granularity is provided in ...
plots	logical – if TRUE then create a barplot of the target-specific shrinkage weights. Recommend option FALSE if using many iterations.
ext.data	matrix – an external data matrix used a surrogate to estimate the parameters in the default target set for X. Never recommended unless there is a belief that ext.data is informative of the covariances of X.

Value

list –

sigmahat matrix – the estimated covariance matrix.

targets array – the targets used for shrinkage.

weights matrix – the weight of each (target, alpha) pair such that sum(weights)=1. The weights are calculated by normalising the log-marginal likelihood values below.

logmarginals matrix – the values of the log marginal likelihood for each (target, alpha) pair.

alpha list – the values of shrinkage intensities used.

References

Gray, H., Leday, G.G.R., Vallejos, C.A. and Richardson, S., 2018. Shrinkage estimation of large covariance matrices using multiple shrinkage targets. arXiv preprint.

Examples

```
set.seed(101)
X <- matrix(rnorm(50), 10, 5) # p=10, n=5, identity covariance
X <- t(scale(t(X), center=TRUE, scale=FALSE)) # mean 0
tas <- taShrink(X, plots = FALSE) # apply shrinkage and view target weight bar plot
barplot(targetWeights(tas), names.arg = c(1:9, "S"),
main = "Target-specific shrinkage weights",
col = rainbow(dim(tas$targets)[3]+1), space = 0,
xlab = "Target", ylab = "Weight")
abs(tas$sigmahat - diag(10)) # inspect absolute differences
norm(tas$sigmahat-diag(10), type="F") # calculate loss
# compare this to each single target
norm(gcShrink(X, var=1, cor=1)$sigmahat-diag(10), type="F")
norm(gcShrink(X, var=2, cor=1)$sigmahat-diag(10), type="F")
norm(gcShrink(X, var=3, cor=1)$sigmahat-diag(10), type="F")
norm(gcShrink(X, var=1, cor=2)$sigmahat-diag(10), type="F")
norm(gcShrink(X, var=2, cor=2)$sigmahat-diag(10), type="F")
norm(gcShrink(X, var=3, cor=2)$sigmahat-diag(10), type="F")
norm(gcShrink(X, var=1, cor=3)$sigmahat-diag(10), type="F")
norm(gcShrink(X, var=2, cor=3)$sigmahat-diag(10), type="F")
norm(gcShrink(X, var=3, cor=3)$sigmahat-diag(10), type="F")
```

Index

Appendix E

`pcaNet` package documentation

In this Appendix we provide the documentation for the `pcaNet` software package.

Package 'pcaNet'

October 4, 2019

Type Package

Title Probabilistic principal components analysis - covariance estimation and network reconstruction

Version 1.0

Date 2019-09-28

Author Paul DW Kirk and Harry Gray

Maintainer <paul.kirk@mrc-bsu.cam.ac.uk> <h.w.gray@dundee.ac.uk>

Description Various implementations of algorithms for probabilistic PCA, with an emphasis on covariance matrix estimation and network reconstruction in the presence of missing values.

License GPL (>= 2) | file LICENSE

Depends R (>= 3.3.0)

biocViews pcaMethods

Imports Rcpp (>= 0.12.9), pcaMethods (>= 1.70.0), fdrtool, plotrix, igraph, pheatmap, methods, mvtnorm, RColorBrewer

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.1.1

NeedsCompilation yes

R topics documented:

`pcaNet-package` *Probabilistic principal components analysis - covariance estimation and network reconstruction*

Description

Various implementations of algorithms for probabilistic PCA, with an emphasis on covariance matrix estimation and network reconstruction in the presence of missing values.

Details

Algorithms for PPCA have been ported from the `PCAMV` MATLAB toolbox (Ilin and Raiko, 2010) and extended from the `pcaMethods` (Stacklies et. al., 2007) R-package to focus on covariance matrix estimation and network reconstruction in the presence of missing values. Full PCA functionality with `pcaMethods` is retained in `pcaNet` due to the use of the pcaRes class.

The inverse of the covariance matrix from PPCA can be computed efficiently, and this functionality is provided in `ppca2Covinv`. Using the false discovery rate method from Strimmer (2008), the estimated partial correlations can be tested to construct a network. Whilst default behaviour for this is available, the full output of the testing is also provided, so that users may further explore the statistics using `fdrtool`. Functionality for visualising the covariance matrix is provided, as well as for the reconstructed network using `igraph` (Csardi and Nepusz, 2006).

Author(s)

Paul DW Kirk and Harry Gray

Maintainers: <paul.kirk@mrc-bsu.cam.ac.uk> <h.w.gray@dundee.ac.uk>

References

Oba, S., Sato, M.A., Takemasa, I., Monden, M., Matsubara, K.I. and Ishii, S., 2003. doi.

Stacklies, W., Redestig, H., Scholz, M., Walther, D. and Selbig, J., 2007. doi.

Ilin, A. and Raiko, T., 2010. link

Porta, J.M., Verbeek, J.J. and Kroese, B.J., 2005. link

Strimmer, K., 2008. link.

Strimmer, K., 2008. doi.

Csardi, G. and Nepusz, T., 2006. link.

See Also

`pcaMethods`

bpcaNet *Bayesian PCA updates*

Description

Perform parameter updates for PPCA using the Variational Bayes framework from Oba (2003). Not recommended to use standalone, rather it is called from within bpcapM and its wrapper pcapM.

Usage

```
bpcaNet(myMat, covy, N, D, hidden, numberOfNonNAvaluesInEachCol,
  nomissIndex, missIndex, nMissing, nPcs = 2L, threshold = 1e-04,
  maxIterations = 200L)
```

Arguments

myMat	matrix – data matrix with observations in rows and variables in columns. (Note that this is the transpose of X in pca_full.)
covy	matrix – the unbiased sample covariance of the data matrix.
N	numeric – the number of observations.
D	numeric – the number of variables.
hidden	numeric – indices of missing values in 1:length(myMat).
numberOfNonNAvaluesInEachCol	numeric – number of observed values in each column of the data (i.e. variables).
nomissIndex	numeric – indices of rows (observations) without any missing values.
missIndex	numeric – indices of rows (observations) with missing values.
nMissing	numeric – total number of missing values.
nPcs	numeric – number of components/latent variables to use.
threshold	numeric – threshold for convergence, applied to the precision parameter tau. Updates for which the change in tau are below this threshold value stop the algorithm.
maxIterations	numeric – the maximum number of iterations to be completed.

Value

A list of 5 elements:

W matrix – the estimated loadings.

ss numeric – the estimated model variance.

C matrix – the estimated covariance matrix.

scores matrix – the estimated scores.

m numeric – the estimated mean vector.

References

Oba, S., Sato, M.A., Takemasa, I., Monden, M., Matsubara, K.I. and Ishii, S., 2003. doi.

Stacklies, W., Redestig, H., Scholz, M., Walther, D. and Selbig, J., 2007. doi.

See Also

bpcapM, pcapM

Examples

```
set.seed(102)
  N <- 20
  D <- 20
  nPcs <- 2
  maxIterations <- 1000
  X <- matrix(rnorm(50), D, N)
  X <- scale(X, center=TRUE, scale=FALSE) # mean 0
  covX <- cov(X)
  IX <- sample(1:D, 10)
  JX <- sample(1:N, 10)
  nMissing <- length(IX)+length(JX)
  X[JX, IX] <- 0
  hidden <- which(X==0)
  numberOfNonNAvaluesInEachCol <- colSums(X!=0)
  nomissIndex <- which(rowSums(X!=0)==N)
  missIndex <- which(rowSums(X!=0)!=N)
  threshold <- 1e-4
  bpcaNetOutput <- bpcaNet(myMat=X, covy=covX, N=N, D=D, hidden=hidden,
    numberOfNonNAvaluesInEachCol=numberOfNonNAvaluesInEachCol,
    nomissIndex=nomissIndex, missIndex=missIndex, nMissing=nMissing,
    nPcs=nPcs, threshold=threshold, maxIterations=maxIterations)
```

---

bpcapM *Bayesian PCA (pcaMethods version)*

Description

Implements a Bayesian PCA missing value estimator, as in pcaMethods. Use of Rcpp makes this version faster and the emphasised output is the covariance matrix `Sigma`, which can be used for network reconstruction.

Usage

```
bpcapM(myMat, nPcs = NA, threshold = 1e-04, maxIterations = 100,
  loglike = TRUE, verbose = TRUE)
```

Arguments

`myMat`	`matrix` – Pre-processed matrix (centered, scaled) with variables in columns and observations in rows. The data may contain missing values, denoted as `NA`.
`nPcs`	`numeric` – Number of components used for re-estimation. Choosing few components may decrease the estimation precision.
`threshold`	`numeric` – Convergence threshold. If the increase in precision of an update falls below this then the algorithm is stopped.
`maxIterations`	`numeric` – Maximum number of estimation steps.
`loglike`	`logical` – should the log-likelihood of the estimated parameters be returned? See Details.
`verbose`	`logical` – verbose intermediary algorithm output.

Details

Details about the probabilistic model underlying BPCA are found in Oba et. al 2003. The algorithm uses an expectation maximation approach together with a Bayesian model to approximate the principal axes (eigenvectors of the covariance matrix in PCA). The estimation is done iteratively, the algorithm terminates if either the maximum number of iterations is reached or if the estimated increase in precision falls below $1e^{-4}$.

Value

A `list` of 4 elements:

W `matrix` – the estimated loadings.

sigmaSq `numeric` – the estimated isotropic variance.

Sigma `matrix` – the estimated covariance matrix.

pcaMethodsRes `class` – see pcaRes.

References

Oba, S., Sato, M.A., Takemasa, I., Monden, M., Matsubara, K.I. and Ishii, S., 2003. doi.

Stacklies, W., Redestig, H., Scholz, M., Walther, D. and Selbig, J., 2007. doi.

See Also

pcapM

Examples

```
# simulate a dataset from a zero mean factor model X = Wz + epsilon
# start off by generating a random binary connectivity matrix
n.factors <- 5
n.genes <- 200
# with dense connectivity
# set.seed(20)
conn.mat <- matrix(rbinom(n = n.genes*n.factors,
                          size = 1, prob = 0.7), c(n.genes, n.factors))

# now generate a loadings matrix from this connectivity
loading.gen <- function(x){
  ifelse(x==0, 0, rnorm(1, 0, 1))
}

W <- apply(conn.mat, c(1, 2), loading.gen)

# generate factor matrix
n.samples <- 100
z <- replicate(n.samples, rnorm(n.factors, 0, 1))

# generate a noise matrix
sigma.sq <- 0.1
epsilon <- replicate(n.samples, rnorm(n.genes, 0, sqrt(sigma.sq)))

# by the ppca equations this gives us the data matrix
X <- W%*%z + epsilon
WWt <- tcrossprod(W)
```

```
Sigma <- WWt + diag(sigma.sq, n.genes)

# select 10% of entries to make missing values
missFrac <- 0.1
inds <- sample(x = 1:length(X),
               size = ceiling(length(X)*missFrac),
               replace = FALSE)

# replace them with NAs in the dataset
missing.dataset <- X
missing.dataset[inds] <- NA

# run bpca
bp <- bpcapM(t(missing.dataset), nPcs = 5)
names(bp)

# sigmasq estimation
abs(bp$sigmaSq-sigma.sq)

# X reconstruction
recon.X <- bp$pcaMethodsRes@loadings%*%t(bp$pcaMethodsRes@scores)
norm(recon.X-X, type="F")^2/(length(X))

# covariance estimation
norm(bp$Sigma-Sigma, type="F")^2/(length(X))
```

`compute_loglikeimp` *Compute the log-likelihood of the observed data given PCA parameter estimates*

Description

The log-likelihood of the data for probabilistic PCA is known to be multivariate Gaussian. Using this, one can check the log-likelihood value of the observed data values given the parameter estimates from the PCA model. This can be useful to compare different models.

Usage

```
compute_loglikeimp(dat, A, S, covmat, meanvec, verbose = TRUE)
```

Arguments

`dat`	`matrix` – the data matrix with variables in rows and observations in columns.
`A`	`matrix` – estimated loadings matrix with observed variables in rows and latent variables in columns.
`S`	`matrix` – estimated factor scores matrix with latent variables in rows and observations in columns.
`covmat`	`matrix` – the estimated covariance matrix.
`meanvec`	`numeric` – the estimated mean vector.
`verbose`	`logical` – whether extra output should be displayed.

Value

the log-likelihood value

Examples

```
p <- 20
n <- 20
set.seed(10045)
  verbose <- 1
  bias <- 1
  rotate2pca <- 1
  ncomp <- 2
  maxiters <- 1000
  opts <- list(init='random',
   maxiters=as.numeric(1000),
   niter_broadprior=as.numeric(100),
   earlystop=as.numeric(0)
  )
  use_prior = 1
  use_postvar = 1
  X <- matrix(rnorm(p*n), p, n)
  miss.inds <- sample(1:(p*n), round(p*n/10))
  X[miss.inds] <- NaN
  Xsaved <- X
  M <- !is.nan(X)
  X[X==0]      <- .Machine$double.eps
  X[is.nan(X)] <- 0

  notmiss <- which(X!=0, arr.ind = TRUE)
  IX      <- notmiss[,1]
  JX      <- notmiss[,2]

  Nobs_i = rowSums(M)
  ndata   <- length(IX)
  # C++ indexing
  IX <- IX -1
  JX <- JX -1

  initialisedParms <- initParms(p, n, ncomp, verbose = verbose)
  A   <- initialisedParms$A
  S   <- initialisedParms$S
  Mu  <- initialisedParms$Mu
  V   <- initialisedParms$V
  Av  <- initialisedParms$Av
  Sv  <- initialisedParms$Sv
  Muv <- initialisedParms$Muv
  Va  <- 1000*rep(1,ncomp)
  Vmu <- 1000
  Mu <- rowSums(X) / Nobs_i
  computedRMS <- compute_rms(X, A, S, M, ndata, verbose = verbose)
  errMx       <- computedRMS$errMx
  rms         <- computedRMS$rms
  hpVa <- 0.001
  hpVb <- 0.001
  hpV  <- 0.001
  Isv <- rep(0, 2)
```

```
# data centering
X <- subtractMu(Mu, X, M, p, n, bias, verbose = verbose)
ppcaOutput <- pca_updates(X=X, V=V, A=A, Va=Va, Av = Av, S = S, Sv = Sv,
Mu = Mu, Muv = Muv, Vmu = Vmu,
hpVa = hpVa, hpVb = hpVb, hpV = hpV, ndata = ndata, Nobs_i = Nobs_i,
Isv = Isv, M = M, IX = IX, JX = JX, rms = rms, errMx = errMx,
bias = bias, rotate2pca = rotate2pca, niter_broadprior = opts$niter_broadprior,
use_prior = use_prior, use_postvar = use_postvar,
maxiters = maxiters, verbose = verbose)
# initialised model log-likelihood
compute_loglikeimp(dat=Xsaved, A=A, S=S, covmat=tcrossprod(A)+diag(p),
meanvec=Mu, verbose=TRUE)
# estimated model log-likelihood
compute_loglikeimp(dat=Xsaved, A=ppcaOutput$W, S=t(ppcaOutput$scores), covmat=ppcaOutput$C,
meanvec=ppcaOutput$m, verbose=TRUE)
```

`compute_loglikeobs` *Compute the log-likelihood of the observed data given PCA parameter estimates*

Description

The log-likelihood of the data for probabilistic PCA is known to be multivariate Gaussian. Using this, one can check the log-likelihood value of the observed data values given the parameter estimates from the PCA model. This can be useful to compare different models.

Usage

```
compute_loglikeobs(dat, covmat, meanvec, verbose = TRUE)
```

Arguments

`dat`	`matrix` – the data matrix with variables in rows and observations in columns.
`covmat`	`matrix` – the estimated covariance matrix.
`meanvec`	`numeric` – the estimated mean vector.
`verbose`	`logical` – whether extra output should be displayed.

Value

the log-likelihood value

Examples

```
p <- 20
n <- 7
set.seed(10045)
X <- matrix(rnorm(p*n), p, n)
miss.inds <- sample(1:(p*n), (p*n)/4)
X[miss.inds] <- NA
M <- !is.na(X)
Nobs_i <- rowSums(M)
Mu <- rowSums(X, na.rm = TRUE) / Nobs_i
Mu2 <- rep(0, p)
```

```
covmat <- diag(p)
# using sample mean
compute_loglikeobs(dat=X, covmat=covmat, meanvec=Mu, verbose=TRUE)
# using zero mean
compute_loglikeobs(dat=X, covmat=covmat, meanvec=Mu2, verbose=TRUE)
```

`compute_rms` *Compute the root mean-squared error of a PCA projection*

Description

Root mean-squared error is the square root of the element-wise error's mean. This is a useful quantity to display during parameter estimation in `pca_updates` since it is a measure of how well the PCA projection is fitting the data.

Usage

```
compute_rms(X, A, S, M, ndata, verbose = TRUE)
```

Arguments

`X`	`matrix` – the data matrix with variables in rows and observations in columns.
`A`	`matrix` – initialised loadings matrix with observed variables in rows and latent variables in columns.
`S`	`matrix` – initialised factor scores matrix with latent variables in rows and observations in columns.
`M`	`matrix` – logical matrix whose values indicate whether the corresponding entry in `X` is observed.
`ndata`	`numerical` – the total number of observed values.
`verbose`	`logical` – whether extra output should be displayed.

Value

A `list` of length 2:

errMx `matrix` – matrix of element-wise differences (errors) between the observed data and the PCA projection.

rms `numerical` – root mean-squared error of the PCA projection.

Examples

```
p <- 20
n <- 7
set.seed(10045)
X <- matrix(rnorm(p*n), p, n)
miss.inds <- sample(1:(p*n), (p*n)/4)
X[miss.inds] <- NA
M <- !is.na(X)
Nobs_i <- rowSums(M)
Mu <- rowSums(X, na.rm = TRUE) / Nobs_i
update_bias <- TRUE
```

```
Xcent <- subtractMu(Mu=Mu, X=X, M=M, p=p, n=n, update_bias=update_bias, verbose=TRUE)
init.model <- initParms(p=p, n=n, ncomp=2, verbose = TRUE)
compute_rms(X=X, A=init.model$A, S=init.model$S, M=M, ndata=sum(Nobs_i), verbose=TRUE)
```

`initParms` *Initialise model parameters for* `pca_updates`

Description

Internal function within `pca_full` that initialises most model parameters. WARNING: does not initialise all parameters by itself correctly (since this depends on context) and so care should be taken when using as a standalone function.

Usage

```
initParms(p, n, ncomp, verbose = TRUE)
```

Arguments

`p`	`numeric` – the number of variables
`n`	`numeric` – the number of observations
`ncomp`	`numeric` – the number of components/latent variables
`verbose`	`logical` – whether extra output should be displayed

Details

Random initialisations are set for the loadings and scores matrices. The mean vector is initialised to `c()` and set outside this function. Diagonal matrices are set for the elements of `Av` and `Sv`. `V` is initialised to 1 and `Muv` is initialised to a vector of 1s.

Value

A `list` of length 7:

A `matrix` – initialised loadings matrix with observed variables in rows and latent variables in columns.

S `matrix` – initialised factor scores matrix with latent variables in rows and observations in columns.

Mu `numeric` – initialised mean vector.

V `numeric` – scalar value corresponding to the initialised variance of the error parameter.

Av `array` – initialised covariance matrices of the rows of A.

Sv `array` – initialised covariance matrices of the rows of S.

Muv `numeric` – the initialisation of the prior variance of Mu.

Examples

```
init.model <- initParms(p=10, n=10, ncomp=2, verbose = TRUE)
init.model$A
init.model$Av
```

orthMat *Calculate an orthonormal basis*

Description

A copied (unexported) function from pcaMethods. ONB = orth(mat) is an orthonormal basis for the range of matrix mat. That is, ONB' * ONB = I, the columns of ONB span the same space as the columns of mat, and the number of columns of ONB is the rank of mat.

Usage

```
orthMat(mat, skipInac = FALSE)
```

Arguments

mat	matrix – matrix to calculate the orthonormal basis of
skipInac	logical – do not include components with precision below .Machine$double.eps if TRUE

Value

orthonormal basis for the range of mat

Author(s)

Wolfram Stacklies

References

Stacklies, W., Redestig, H., Scholz, M., Walther, D. and Selbig, J., 2007. doi.

See Also

orth

Examples

```
set.seed(102)
X <- matrix(rnorm(10), 5, 2)
norm(X[,1], type="2")
norm(X[,2], type="2")
t(X[,1])%*%X[,2]
Xorth <- orthMat(X)
# now unit norms
norm(Xorth[,1], type="2")
norm(Xorth[,2], type="2")
# and zero dot product
t(Xorth[,1])%*%Xorth[,2]
```

pcapM *A wrapper for pcaMethods function implementations*

Description

Implements the equivalent of pca. This function preprocesses the data as specified by the user, then calls ppcapM or bpcapM, and finally handles this output to return a list. One element of the output is a pcaRes object.

Usage

```
pcapM(myMat, nPcs = 2, method = "ppca", seed = NA,
  threshold = 1e-04, maxIterations = 1000, center = TRUE,
  scale = c("none", "pareto", "vector", "uv"), loglike = TRUE,
  verbose = TRUE)
```

Arguments

myMat	matrix – Data matrix with variables in columns and observations in rows. The data may contain missing values, denoted as NA.
nPcs	numeric – Number of components used for re-estimation. Choosing few components may decrease the estimation precision.
method	c("ppca","bpca") – frequentist or Bayesian estimation of model parameters.
seed	numeric – the random number seed used, useful to specify when comparing algorithms.
threshold	numeric – Convergence threshold. If the increase in precision of an update falls below this then the algorithm is stopped.
maxIterations	numeric – Maximum number of estimation steps.
center	logical – should the data be centered?
scale	c("none","pareto","vector","uv") – which method of scaling should be used? See pca.
loglike	logical – should the log-likelihood of the estimated parameters be returned? See Details.
verbose	logical – verbose intermediary algorithm output.

Details

See ppcapM and bpcapM for the algorithm specifics. loglike indicates whether log-likelihood values for the resulting estimates should be computed. This can be useful to compare different algorithms.

Value

A list of 5 or 7 elements, depending on the value of loglike:

W matrix – the estimated loadings.

sigmaSq numeric – the estimated isotropic variance.

Sigma matrix – the estimated covariance matrix.

m `numeric` – the estimated mean vector.

logLikeObs `numeric` – the log-likelihood value of the observed data given the estimated parameters.

logLikeImp `numeric` – the log-likelihood value of the imputed data given the estimated parameters.

pcaMethodsRes `class` – see pcaRes.

Examples

```
# simulate a dataset from a zero mean factor model X = Wz + epsilon
# start off by generating a random binary connectivity matrix
n.factors <- 5
n.genes <- 200
# with dense connectivity
# set.seed(20)
conn.mat <- matrix(rbinom(n = n.genes*n.factors,
                          size = 1, prob = 0.7), c(n.genes, n.factors))

# now generate a loadings matrix from this connectivity
loading.gen <- function(x){
  ifelse(x==0, 0, rnorm(1, 0, 1))
}

W <- apply(conn.mat, c(1, 2), loading.gen)

# generate factor matrix
n.samples <- 100
z <- replicate(n.samples, rnorm(n.factors, 0, 1))

# generate a noise matrix
sigma.sq <- 0.1
epsilon <- replicate(n.samples, rnorm(n.genes, 0, sqrt(sigma.sq)))

# by the ppca equations this gives us the data matrix
X <- W%*%z + epsilon
WWt <- tcrossprod(W)
Sigma <- WWt + diag(sigma.sq, n.genes)

# select 10% of entries to make missing values
missFrac <- 0.1
inds <- sample(x = 1:length(X),
               size = ceiling(length(X)*missFrac),
               replace = FALSE)

# replace them with NAs in the dataset
missing.dataset <- X
missing.dataset[inds] <- NA

# run ppca
ppm <- pcapM(t(missing.dataset), nPcs=5, method="bpca", seed=2009,
maxIterations=1000, center=TRUE, loglike=TRUE, verbose=TRUE)
```

pca_full *A wrapper for PCAMV (MATLAB) function implementations*

Description

Implements the PPCA algorithms from See Ilin and Raiko (2010), previously only available in MATLAB. One element of the outputs is a pcaRes object, providing an interface between PCAMV and pcaMethods.

Usage

```
pca_full(X, ncomp = NA, algorithm = "vb", maxiters = 1000,
  bias = TRUE, rotate2pca = TRUE, loglike = TRUE, verbose = TRUE)
```

Arguments

X	matrix – Data matrix with observations in columns and variables in rows. The data may contain missing values, denoted as NA, or NaN.
ncomp	numeric – Number of components used for re-estimation. Choosing few components may decrease the estimation precision. Setting to NA results in ncomp = min(n, p) -1, which will be slow for large data.
algorithm	c("ppca","map","vb") – the algorithm to be used for estimation, see Details.
maxiters	numeric – Maximum number of estimation steps.
bias	logical – should the mean be estimated?
rotate2pca	logical – should the solution be rotated to a PCA basis? See Details.
loglike	logical – should the log-likelihood of the estimated parameters be returned? See Details.
verbose	logical – verbose intermediary algorithm output.

Details

The algorithm argument provides the option of performing either 'ppca' for PPCA, 'vb' for BPCA using a variational approximation, or 'map' for a variational approximation ignoring posterior uncertainty (for faster computation). See Ilin and Raiko (2010) for the full models. Setting rotate2pca will perform a post-estimation rotation of the scores and loadings matrices so that they satisfy the PCA conditions of orthonormality, see See Ilin and Raiko (2010) for the derivations. loglike indicates whether log-likelihood values for the resulting estimates should be computed. This can be useful to compare different algorithms.

Value

A list of 6 or 8 elements, depending on the value of loglike:

W matrix – the estimated loadings.

sigmaSq numeric – the estimated isotropic variance.

Sigma matrix – the estimated covariance matrix.

m numeric – the estimated mean vector.

logLikeObs numeric – the log-likelihood value of the observed data given the estimated parameters.

logLikeImp `numeric` – the log-likelihood value of the imputed data given the estimated parameters.

m `numeric` – the number of iterations taken to converge.

pcaMethodsRes `class` – see pcaRes.

References

Ilin, A. and Raiko, T., 2010. link

Examples

```
# simulate a dataset from a zero mean factor model X = Wz + epsilon
# start off by generating a random binary connectivity matrix
n.factors <- 5
n.genes <- 200
# with dense connectivity
# set.seed(20)
conn.mat <- matrix(rbinom(n = n.genes*n.factors,
                          size = 1, prob = 0.7), c(n.genes, n.factors))

# now generate a loadings matrix from this connectivity
loading.gen <- function(x){
  ifelse(x==0, 0, rnorm(1, 0, 1))
}

W <- apply(conn.mat, c(1, 2), loading.gen)

# generate factor matrix
n.samples <- 100
z <- replicate(n.samples, rnorm(n.factors, 0, 1))

# generate a noise matrix
sigma.sq <- 0.1
epsilon <- replicate(n.samples, rnorm(n.genes, 0, sqrt(sigma.sq)))

# by the ppca equations this gives us the data matrix
X <- W%*%z + epsilon
WWt <- tcrossprod(W)
Sigma <- WWt + diag(sigma.sq, n.genes)

# select 10% of entries to make missing values
missFrac <- 0.1
inds <- sample(x = 1:length(X),
               size = ceiling(length(X)*missFrac),
               replace = FALSE)

# replace them with NAs in the dataset
missing.dataset <- X
missing.dataset[inds] <- NA

# run ppca
ppf <- pca_full(missing.dataset, ncomp=5, algorithm="vb", maxiters=5,
bias=TRUE, rotate2pca=FALSE, loglike=TRUE, verbose=TRUE)
```

pca_updates *PPCA updates*

Description

Perform the parameter updates for PPCA using either Expectation- Maximisation or Variational Bayes as in Ilin and Raiko (2010). Recommended to not use standalone, rather this function is called within pca_full.

Usage

```
pca_updates(X, V, A, Av, Va, S, Sv, Mu, Muv, Vmu, hpVa, hpVb, hpV, ndata,
  Nobs_i, Isv, M, IX, JX, rms, errMx, bias = 1L, rotate2pca = 1L,
  niter_broadprior = 100L, use_prior = 1L, use_postvar = 1L,
  maxiters = 1000L, verbose = 1L)
```

Arguments

X	matrix – data matrix with variables in rows and observations in columns.
V	numeric – scalar value corresponding to the initialised variance of the error parameter.
A	matrix – initialised loadings matrix with observed variables in rows and latent variables in columns
Av	array – initialised covariance matrices of the rows of A.
Va	numeric – the hyperparameter of the prior variance of the rows of A.
S	matrix – initialised factor scores matrix with latent variables in rows and observations in columns.
Sv	array – initialised covariance matrices of the rows of S.
Mu	numeric – vector corresponding to the initialised mean of the observed variables.
Muv	numeric – the initialisation of the prior variance of Mu.
Vmu	numeric – the hyperparameter of the prior variance of Mu
hpVa	numeric – hyperparameter for the prior of the variance of Vmu and Va.
hpVb	numeric – hyperparameter for the prior of the variance of Vmu and Va.
hpV	numeric – hyperparameter for the prior of V.
ndata	numeric – number of observed values.
Nobs_i	numeric – number of observed values in each row of the data.
Isv	numeric – indices j for Svj that are identical. Not currently used.
M	matrix – logical values indicating which elements of the data are observed and missing.
IX	numeric – row indices of missing values
JX	numeric – column indices of missing values
rms	numeric – scalar indicating the initial rms
errMx	matrix – initial error matrix whose elements correspond to difference between the observed data and its model prediction.

bias	logical – value indicating whether the mean vector should be estimated or not.
rotate2pca	logical – value indicating whether to rotate the pca solution during learning.
niter_broadprior	numeric – number of iterations before the prior parameters begin to be updated.
use_prior	logical – whether or not a prior is assumed for the model parameters.
use_postvar	logical – whether the posterior variance should be computed and taken into account.
maxiters	numeric – the maximum number of iterations to be completed.
verbose	logical – whether extra output, such as the iteration number and cost function value, should be displayed.

Value

A list of 6 elements:

scores matrix – the estimated scores.

m numeric – the estimated mean vector.

ss numeric – the estimated model variance.

W matrix – the estimated loadings.

C matrix – the estimated covariance matrix.

numIter numeric – the number of iterations.

References

Ilin, A. and Raiko, T., 2010. link.

See Also

pca_full

Examples

```
set.seed(102)
n <- 20
p <- 20
verbose <- 1
bias <- 1
rotate2pca <- 1
ncomp <- 2
maxiters <- 1000
opts <- list(init='random',
  maxiters=as.numeric(1000),
  niter_broadprior=as.numeric(100),
  earlystop=as.numeric(0)
)
use_prior = 1
use_postvar = 1
X <- matrix(rnorm(50), p, n)
X <- t(scale(t(X), center=TRUE, scale=FALSE))
IX <- sample(1:p, 10)
JX <- sample(1:n, 10)
X[IX, JX] <- 0
```

```
M <- X!=0
Nobs_i = rowSums(M)
ndata   <- length(IX)
# C++ indexing
IX <- IX -1
JX <- JX -1

initialisedParms <- initParms(p, n, ncomp, verbose = verbose)
A   <- initialisedParms$A
S   <- initialisedParms$S
Mu  <- initialisedParms$Mu
V   <- initialisedParms$V
Av  <- initialisedParms$Av
Sv  <- initialisedParms$Sv
Muv <- initialisedParms$Muv
Va  <- 1000*rep(1,ncomp)
Vmu <- 1000
if (is.null(Mu)){
  if (bias){
    Mu <- rowSums(X) / Nobs_i
}else{
  Mu = rep(0, p)
  }
}
computedRMS <- compute_rms(X, A, S, M, ndata, verbose = verbose)
errMx       <- computedRMS$errMx
rms         <- computedRMS$rms
hpVa <- 0.001
hpVb <- 0.001
hpV  <- 0.001
Isv <- rep(0, 2)
# data centering
X <- subtractMu(Mu, X, M, p, n, bias, verbose = verbose)
ppcaOutput <- pca_updates(X=X, V=V, A=A, Va=Va, Av = Av, S = S, Sv = Sv,
Mu = Mu, Muv = Muv, Vmu = Vmu,
hpVa = hpVa, hpVb = hpVb, hpV = hpV, ndata = ndata, Nobs_i = Nobs_i,
Isv = Isv, M = M, IX = IX, JX = JX, rms = rms, errMx = errMx,
bias = bias, rotate2pca = rotate2pca, niter_broadprior = opts$niter_broadprior,
use_prior = use_prior, use_postvar = use_postvar,
maxiters = maxiters, verbose = verbose)
```

`ppca2Covinv` *Inverse covariance matrix computation from PPCA*

Description

Efficient inversion of the covariance matrix estimated from PPCA.

Usage

```
ppca2Covinv(ppcaOutput)
```

Arguments

ppcaOutput	list – the output object from running any of the PPCA functions in this package.

Details

The computation exploits the Woodbury identity so that a kxk matrix (where k is often less than 10) is inverted instead of the potentially large pxp matrix. The closed-form expression for the inverse depends upon parameters that are estimated in the PPCA algorithm.

Value

matrix – the inverse of the covariance matrix.

Examples

```
# simulate a dataset from a zero mean factor model X = Wz + epsilon
# start off by generating a random binary connectivity matrix
n.factors <- 5
n.genes <- 200
# with dense connectivity
# set.seed(20)
conn.mat <- matrix(rbinom(n = n.genes*n.factors,
                          size = 1, prob = 0.7), c(n.genes, n.factors))

# now generate a loadings matrix from this connectivity
loading.gen <- function(x){
  ifelse(x==0, 0, rnorm(1, 0, 1))
}

W <- apply(conn.mat, c(1, 2), loading.gen)

# generate factor matrix
n.samples <- 100
z <- replicate(n.samples, rnorm(n.factors, 0, 1))

# generate a noise matrix
sigma.sq <- 0.1
epsilon <- replicate(n.samples, rnorm(n.genes, 0, sqrt(sigma.sq)))

# by the ppca equations this gives us the data matrix
X <- W%*%z + epsilon
WWt <- tcrossprod(W)
Sigma <- WWt + diag(sigma.sq, n.genes)

# select 10% of entries to make missing values
missFrac <- 0.1
inds <- sample(x = 1:length(X),
               size = ceiling(length(X)*missFrac),
               replace = FALSE)

# replace them with NAs in the dataset
missing.dataset <- X
missing.dataset[inds] <- NA

# run ppca
```

```
ppf <- pca_full(missing.dataset, ncomp=5, algorithm="vb", maxiters=5,
bias=TRUE, rotate2pca=FALSE, loglike=TRUE, verbose=TRUE)

# compute the inverse
covinv <- ppca2Covinv(ppf)
system.time(ppca2Covinv(ppf))

covinv2 <- solve(ppf$Sigma)
system.time(solve(ppf$Sigma))
```

---

ppca2Covplot *Covariance matrix visualisation*

Description

Heatmap visualisation of the covariance matrix estimated within PPCA.

Usage

```
ppca2Covplot(ppcaOutput)
```

Arguments

ppcaOutput	list – the output object from running any of the PPCA functions in this package.

Value

plot of the estimated covariance matrix

Examples

```
#' # simulate a dataset from a zero mean factor model X = Wz + epsilon
# start off by generating a random binary connectivity matrix
n.factors <- 5
n.genes <- 200
# with dense connectivity
# set.seed(20)
conn.mat <- matrix(rbinom(n = n.genes*n.factors,
                          size = 1, prob = 0.7), c(n.genes, n.factors))

# now generate a loadings matrix from this connectivity
loading.gen <- function(x){
  ifelse(x==0, 0, rnorm(1, 0, 1))
}

W <- apply(conn.mat, c(1, 2), loading.gen)

# generate factor matrix
n.samples <- 100
z <- replicate(n.samples, rnorm(n.factors, 0, 1))

# generate a noise matrix
sigma.sq <- 0.1
```

```
epsilon <- replicate(n.samples, rnorm(n.genes, 0, sqrt(sigma.sq)))

# by the ppca equations this gives us the data matrix
X <- W%*%z + epsilon
WWt <- tcrossprod(W)
Sigma <- WWt + diag(sigma.sq, n.genes)

# select 10% of entries to make missing values
missFrac <- 0.1
inds <- sample(x = 1:length(X),
               size = ceiling(length(X)*missFrac),
               replace = FALSE)

# replace them with NAs in the dataset
missing.dataset <- X
missing.dataset[inds] <- NA

# run ppca
ppf <- pca_full(missing.dataset, ncomp=5, algorithm="vb", maxiters=5,
bias=TRUE, rotate2pca=FALSE, loglike=TRUE, verbose=TRUE)

# plot the matrix
ppca2Covplot(ppf)
```

---

ppca2Net *Network reconstruction from PPCA*

Description

Constructs a conditional independence network of the observed variables from the data using the implicitly estimated covariance matrix within PPCA.

Usage

```
ppca2Net(ppcaOutput, plot = TRUE, verbose = TRUE, vertex.size = 10,
  edge.width = 2, vertex.label.cex = 0.4, vertex.color = "cyan",
  vertex.label.color = "black", edge.color = "pink",
  vertex.label.family = "Helvetica", vertex.label = NULL)
```

Arguments

ppcaOutput	list – the output object from running any of the PPCA functions in this package.
plot	logical – visualise the resulting network.
verbose	logical – verbose intermediary output.
vertex.size	see igraph.plotting
edge.width	see igraph.plotting
vertex.label.cex	see igraph.plotting
vertex.color	see igraph.plotting

vertex.label.color
: see igraph.plotting

edge.color
: see igraph.plotting

vertex.label.family
: see igraph.plotting

vertex.label
: see igraph.plotting

Details

Covariance estimation is done as a preliminary step for this function. The function then inverts this matrix, which can be done very efficiently, to obtain the precision matrix. Then the precision matrix is scaled to unit variance (diagonal) to obtain partial correlation estimates in the off-diagonal entries, which is a measure of conditional independence. A two component mixture model is then fit to the distribution of partial correlations using fdrtool. The partial correlations that are not part of the 'null' component are then selected as true edges of the network, effectively setting the null values to 0. The function then visualises the resulting network using plot.igraph. The user can extract the fdr.stats element of this output to view the full output of fdrtool, from which the magnitude and significance of each partial correlation can be seen (and customised thresholding can be performed). The graph element of the output is an 'igraph' class, and so can be used to easily make alternative visualisations or compute graph statistics.

Value

A list of 2 elements:

graph 'igraph' – Contains the network information.

fdr.stats list – the full output of an internal call to fdrtool. Can be useful to inspect the statistics upon which the network was reconstructed.

References

Strimmer, K., 2008. link.

Strimmer, K., 2008. doi.

Csardi, G. and Nepusz, T., 2006. link.

See Also

igraph, fdrtool

Examples

```
#' # simulate a dataset from a zero mean factor model X = Wz + epsilon
# start off by generating a random binary connectivity matrix
n.factors <- 5
n.genes <- 200
# with dense connectivity
# set.seed(20)
conn.mat <- matrix(rbinom(n = n.genes*n.factors,
                          size = 1, prob = 0.7), c(n.genes, n.factors))

# now generate a loadings matrix from this connectivity
loading.gen <- function(x){
  ifelse(x==0, 0, rnorm(1, 0, 1))
```

```
}

W <- apply(conn.mat, c(1, 2), loading.gen)

# generate factor matrix
n.samples <- 100
z <- replicate(n.samples, rnorm(n.factors, 0, 1))

# generate a noise matrix
sigma.sq <- 0.1
epsilon <- replicate(n.samples, rnorm(n.genes, 0, sqrt(sigma.sq)))

# by the ppca equations this gives us the data matrix
X <- W%*%z + epsilon
WWt <- tcrossprod(W)
Sigma <- WWt + diag(sigma.sq, n.genes)

# select 10% of entries to make missing values
missFrac <- 0.1
inds <- sample(x = 1:length(X),
               size = ceiling(length(X)*missFrac),
               replace = FALSE)

# replace them with NAs in the dataset
missing.dataset <- X
missing.dataset[inds] <- NA

# run ppca
ppf <- pca_full(missing.dataset, ncomp=5, algorithm="vb", maxiters=5,
bias=TRUE, rotate2pca=FALSE, loglike=TRUE, verbose=TRUE)

# compute the network
pcanet <- ppca2Net(ppf, plot=TRUE)
```

---

ppcaNet *Probabilistic PCA updates*

Description

Perform parameter updates for PPCA using the Expectation-Maximisation framework from Porta (2005) and also in the R-package pcaMethods (Stacklies, 2007). Not recommended to use standalone, rather it is called from within ppcapM and its wrapper pcapM.

Usage

```
ppcaNet(myMat, N, D, W, hidden, nMissing, nPcs = 2L, threshold = 1e-05,
  maxIterations = 1000L)
```

Arguments

myMat	matrix – data matrix with observations in rows and variables in columns. (Note that this is the transpose of X in pca_full.)
N	numeric – the number of observations.

D	numeric – the number of variables.
W	matrix – initialised loadings matrix with observed variables in rows and latent variables in columns.
hidden	numeric – indices of missing values in 1:length(myMat).
nMissing	numeric – total number of missing values.
nPcs	numeric – number of components/latent variables to use.
threshold	numeric – threshold for convergence, applied to the precision parameter tau. Updates for which the change in tau are below this threshold value stop the algorithm.
maxIterations	numeric – the maximum number of iterations to be completed.

Value

A list of 4 elements:

W matrix – the estimated loadings.

ss numeric – the estimated model variance.

C matrix – the estimated covariance matrix.

myMat matrix – the data matrix with missing values replaced by their estimated projections.

References

Porta, J.M., Verbeek, J.J. and Kroese, B.J., 2005. link

Stacklies, W., Redestig, H., Scholz, M., Walther, D. and Selbig, J., 2007. doi.

See Also

ppcapM, pcapM

Examples

```
set.seed(102)
N <- 20
D <- 20
nPcs <- 2
maxIterations <- 1000
X <- matrix(rnorm(50), D, N)
X <- scale(X, center=TRUE, scale=FALSE) # mean 0
covX <- cov(X)
IX <- sample(1:D, 10)
JX <- sample(1:N, 10)
nMissing <- length(IX)+length(JX)
X[JX, IX] <- 0
hidden <- which(X==0)
threshold <- 1e-4
r <- sample(N)
W <- t(X[r[1:nPcs], ,drop = FALSE])
W <- matrix(rnorm(W), nrow(W), ncol(W), dimnames = labels(W) )
ppcaNetOutput <- ppcaNet(X, N, D, W, hidden, nMissing, nPcs, threshold, maxIterations)
```

ppcapM *Probabilistic PCA (pcaMethods version)*

Description

Implements a probabilistic PCA missing value estimator, as in pcaMethods. Use of Rcpp makes this version faster and the emphasised output is the covariance matrix `Sigma`, which can be used for network reconstruction.

Usage

```
ppcapM(myMat, nPcs = 2, seed = NA, threshold = 1e-04,
  maxIterations = 1000, loglike = TRUE, verbose = TRUE)
```

Arguments

`myMat`	`matrix` – Pre-processed matrix (centered, scaled) with variables in columns and observations in rows. The data may contain missing values, denoted as `NA`.
`nPcs`	`numeric` – Number of components used for re-estimation. Choosing few components may decrease the estimation precision.
`seed`	`numeric` – the random number seed used, useful to specify when comparing algorithms.
`threshold`	`numeric` – Convergence threshold. If the increase in precision of an update falls below this then the algorithm is stopped.
`maxIterations`	`numeric` – Maximum number of estimation steps.
`loglike`	`logical` – should the log-likelihood of the estimated parameters be returned? See Details.
`verbose`	`logical` – verbose intermediary algorithm output.

Details

Details about the probabilistic model underlying PPCA are found in Bishop 1999. The algorithm (Porta, 2005) uses an expectation maximisation approach together with a probabilistic model to approximate the principal axes (eigenvectors of the covariance matrix in PCA). The estimation is done iteratively, the algorithm terminates if either the maximum number of iterations is reached or if the estimated increase in precision falls below $1e^{-4}$.

Value

A `list` of 4 elements:

W `matrix` – the estimated loadings.

sigmaSq `numeric` – the estimated isotropic variance.

Sigma `matrix` – the estimated covariance matrix.

pcaMethodsRes `class` – see `pcaRes`.

References

Porta, J.M., Verbeek, J.J. and Kroese, B.J., 2005. link

Stacklies, W., Redestig, H., Scholz, M., Walther, D. and Selbig, J., 2007. doi.

Examples

```
# simulate a dataset from a zero mean factor model X = Wz + epsilon
# start off by generating a random binary connectivity matrix
n.factors <- 5
n.genes <- 200
# with dense connectivity
# set.seed(20)
conn.mat <- matrix(rbinom(n = n.genes*n.factors,
                          size = 1, prob = 0.7), c(n.genes, n.factors))

# now generate a loadings matrix from this connectivity
loading.gen <- function(x){
  ifelse(x==0, 0, rnorm(1, 0, 1))
}

W <- apply(conn.mat, c(1, 2), loading.gen)

# generate factor matrix
n.samples <- 100
z <- replicate(n.samples, rnorm(n.factors, 0, 1))

# generate a noise matrix
sigma.sq <- 0.1
epsilon <- replicate(n.samples, rnorm(n.genes, 0, sqrt(sigma.sq)))

# by the ppca equations this gives us the data matrix
X <- W%*%z + epsilon
WWt <- tcrossprod(W)
Sigma <- WWt + diag(sigma.sq, n.genes)

# select 10% of entries to make missing values
missFrac <- 0.1
inds <- sample(x = 1:length(X),
               size = ceiling(length(X)*missFrac),
               replace = FALSE)

# replace them with NAs in the dataset
missing.dataset <- X
missing.dataset[inds] <- NA

# run ppca
pp <- ppcapM(t(missing.dataset), nPcs = 5)
names(pp)

# sigmasq estimation
abs(pp$sigmaSq-sigma.sq)

# X reconstruction
recon.X <- pp$pcaMethodsRes@loadings%*%t(pp$pcaMethodsRes@scores)
norm(recon.X-X, type="F")^2/(length(X))

# covariance estimation
norm(pp$Sigma-Sigma, type="F")^2/(length(X))
```

ppcaQ2 *Cross-validation for PCA*

Description

Internal cross-validation can be used for estimating the level of structure in a data set and to optimise the choice of number of principal components.

Usage

```
ppcaQ2(obj, originalData = obj$pcaMethodsRes@completeObs, fold = 5,
  nruncv = 1, type = c("krzanowski", "impute"),
  verbose = interactive(), variables = 1:(obj$pcaMethodsRes@nVar), ...)
```

Arguments

obj	A pcaRes object (result from previous PCA analysis.)
originalData	The matrix (or ExpressionSet) that used to obtain the pcaRes object.
fold	The number of groups to divide the data in.
nruncv	The number of times to repeat the whole cross-validation
type	krzanowski or imputation type cross-validation
verbose	boolean If TRUE Q2 outputs a primitive progress bar.
variables	indices of the variables to use during cross-validation calculation. Other variables are kept as they are and do not contribute to the total sum-of-squares.
...	Further arguments passed to the pca function called within Q2.

Details

A wrapper for the Q2 function from pcaMethods, which calculates Q^2 for a PCA model. This is the cross-validated version of R^2 and can be interpreted as the ratio of variance that can be predicted independently by the PCA model. Poor (low) Q^2 indicates that the PCA model only describes noise and that the model is unrelated to the true data structure. The definition of Q^2 is:

$$Q^2 = 1 - \frac{\sum_i^p \sum_j^n (X - \hat{X})^2}{\sum_i^p \sum_j^n X^2}$$

for the matrix X which has n rows and p columns. For a given number of PC's X is estimated as $\hat{X} = TP'$ (T are scores and P are loadings). Although this defines the leave-one-out cross-validation this is not what is performed if fold is less than the number of rows and/or columns. In 'impute' type CV, diagonal rows of elements in the matrix are deleted and the re-estimated. In 'krzanowski' type CV, rows are sequentially left out to build fold PCA models which give the loadings. Then, columns are sequentially left out to build fold models for scores. By combining scores and loadings from different models, we can estimate completely left out values. The two types may seem similar but can give very different results, krzanowski typically yields more stable and reliable result for estimating data structure whereas impute is better for evaluating missing value imputation performance. Note that since Krzanowski CV operates on a reduced matrix, it is not possible estimate Q2 for all components and the result vector may therefore be shorter than nPcs(object).

Value

A matrix or vector with Q^2 estimates.

Author(s)

Henning Redestig, Ondrej Mikula

References

Krzanowski, WJ. Cross-validation in principal component analysis. Biometrics. 1987(43):3,575-584

See Also

Q2

Examples

```
# analogously to pcaMethods...
data(iris)
x <- iris[,1:4]
pcIr <- pcapM(as.matrix(x), nPcs=3, method="ppca", seed=104, scale="none")
q2 <- ppcaQ2(pcIr)
barplot(q2, main="Krzanowski CV", xlab="Number of PCs",
 ylab=expression(Q^2))
```

---

subtractMu — *Subtract the row means from a matrix of data with missing values*

Description

internal function within pca_full to subtract the row means from a matrix of data using only the observed values. Offers little utility standalone.

Usage

```
subtractMu(Mu, X, M, p, n, update_bias, verbose = TRUE)
```

Arguments

Mu	numeric – the sample mean of the observed variables.
X	matrix – the data matrix with variables in rows and observations in columns.
M	matrix – logical matrix whose values indicate whether the corresponding entry in X is observed.
p	numeric – the number of variables.
n	numeric – the number of observations.
update_bias	logical – whether the mean should be subtracted. or not.
verbose	logical – whether extra output should be displayed.

Value

X `matrix` – centered data matrix.

Examples

```
p <- 20
n <- 7
set.seed(10045)
X <- matrix(rnorm(p*n), p, n)
miss.inds <- sample(1:(p*n), (p*n)/4)
X[miss.inds] <- NA
M <- !is.na(X)
Nobs_i <- rowSums(M)
Mu <- rowSums(X, na.rm = TRUE) / Nobs_i
update_bias <- TRUE
Xcent <- subtractMu(Mu=Mu, X=X, M=M, p=p, n=n, update_bias=update_bias, verbose=TRUE)
X-Xcent
Mu # all observed values in each column equal to Mu
```

Index